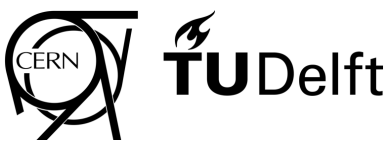
Generalised Automatic Harmonic Operation in the CERN Proton Synchrotron Booster

Double MSc Graduation Thesis (AE & AP)

by

Anibal Luciano Pastinante 5010276

Course:	AE5822/AP3903: Master's Thesis
	Dr. Simon Albright (CERN)
Supervisors:	Dr. Alessandra Menicucci (TU Delft - AE)
	Dr. Ernst-Jan Buis (TU Delft - AP)
Institution:	CERN & Delft University of Technology
Place:	CERN, Prevessin Site
Delivery date:	Monday 22nd September, 2025



Abstract

This thesis addresses profile-based optimization of longitudinal beam dynamics in the CERN Proton Synchrotron Booster (PSB) at two levels of complexity. In the 1D case (double-harmonic), the task is to infer and correct the second-harmonic phase Φ_2 to achieve optimal bunch lengthening throughout the acceleration cycle. The operational diversity of longitudinal profiles in this regime required a supervised learning dataset to cover representative conditions. In the 5D case (triple-harmonic), the goal is to jointly optimize $V_{\text{total}}, r_2, r_3, \Phi_2, \Phi_3$ to obtain a flat-topped, lengthened bunch under stability constraints. Prior Bayesian Optimization (BO) identified a Kullback–Leibler (KL) divergence on the central charge region as a usable objective, removing the need for supervised labels in 5D.

For the double-harmonic optimization, a Convolutional Neural Network (CNN) with Convolutional Block Attention Modules (CBAM) was developed to map measured profiles directly to Φ_2 corrections as a single-shot regressor using a pre-made dataset of simulated profiles. Training leveraged cosine loss, intensity-agnostic max-normalization, stratified training-validation splits, and realistic profile augmentation. A double-layered hyperparameter optimization was performed with the BO framework Optuna using Tree of Parzen Estimators (TPE), with feature importance via Random Forest-based fANOVA and MDI. When applied iteratively, the single-shot regressor lacked an internal notion of convergence, so a decaying-corrections scheme tempered updates. In simulation and on-machine validation in the PSB, CBAM produced strong approximations to operator-tuned phases within sub–super-cycle latency, at times exceeding manual phasing performance in 10 iterations (5 minutes in the PSB). However, its sensitivity to initialization conditions, and limited corrective amplitude under persistent noise motivated a Reinforcement Learning (RL) alternative with better compatibility to the unlabelled 5D case.

A recurrent Long Short-Term Memory–Twin Delayed Deep Deterministic policy gradient (LSTM–TD3) framework was introduced to enable profile-based, continuous iterative control without supervised targets. Methodological advances included a learnable soft-threshold gate on profiles (actor and critic), Prioritized Experience Replay (PER), action bounding, and twin critics trained with the Huber loss instead of Mean Squared Error to hedge against Q-network overestimation bias amplified by PER. To our knowledge, this is the only open-source LSTM–TD3 implementation that combines this loss choice with parallelized environments and PER.

For the Φ_2 phasing problem, the RL agent used a simple phase-centric reward focusing on convergence and beam-loss prevention. The optimised agent achieved the best simulated performance when benchmarked against CBAM, producing better Φ_2 phasing in simulation in fewer iterations with enhanced robustness against different impedances and initial conditions. Due to time constraints, only an earlier, unoptimised variant was validated in the PSB; its behaviour (including over-corrections over the cycle and dependence on initialization) matched verification expectations and was used to extrapolate the optimised agent's expected performance on machine. The results highlighted the value of the gating and Huber loss redesign, with hyperparameter optimization identifying TD3 training parameters as the most influential.

In triple-harmonic optimization, the same recurrent off-policy agent was trained with observations comprising the normalized profile, normalized radio-frequency (RF) parameters ($V_{\text{total}}, r_2, r_3, \Phi_2, \Phi_3$), and a normalized magnetic-field ramp rate. Reward shaping follows a stability-then-shape paradigm through bucket-area sufficiency followed by a KL objective. TD3 training parameters were extrapolated from the 1D case. In simulation, the agent converged in under 20 corrections to effective phases and amplitudes, achieving approximate triple-peak matching within tolerance and demonstrating that RL can solve the 5D optimization without supervised labels while learning safety-centred strategies. Again, the agent showed robustness to different initialization conditions and impedances.

The project delivered embedded RF parameter control over the cycle, a data acquisition and preprocessing pipeline, operational scripts with monitoring, and publicly released training code with supporting documentation. Although optimised agents (1D and 5D) could not be validated on machine within the available time, verification results and safety mechanisms indicate readiness for testing and high likelihood of sufficiently high performance. Future work includes PSB validation of the optimised agents, further latency reduction, online PSB learning for the triple-harmonic agent, and testing for transferability to other accelerators.

Preface

"Physics is like trying to understand the universe's most elaborate joke, just when you think you get the punchline, you realize the setup was two hundred pages long.'

— A RELIEVED AND HOPEFULLY GRADUATED STUDENT

After a year at CERN, witnessing the intricate dance of particles through kilometers of beam lines and the elegant complexity of the world's most sophisticated machines, this work represents the culmination of countless hours spent trying to find an automatized solution to perhaps one of the most trial-and-error problems in it. Many times I have questioned both my sanity throughout this process and how complex can such a rudimentary problem can become when one tries to find an elegant solution to everything. However, thanks to the magnificent people around me, I was able to keep the slightest bit of lucidity to see what a beautiful problem this was and how privileged I am to have been able to work on this together with the brilliant minds at CERN who guided me on this journey. This represents my life's greatest work (to this date).

This one goes out to my supervisor at CERN, Dr. Simon Albright who had more patience with me than I could have ever imagined, with me spouting my ML algorithms at every weekly meeting and switching my entire methodology at a moment's notice. To my supervisors at TU Delft, Dr. Ernst-Jan Buis and Dr. Alessandra Menicucci, who again held strong throughout all the meetings and made the truest of efforts in supporting me throughout this process in a mixture of topics that does not lie in their expertise. To my office mate and partner in ML, Joel Axel Wulff and also to Mariangela Marchi, both of whom helped me realize that I was not going (entirely) crazy on many occasions. To Heiko Damerau, Alexandre Lasheen, and Simon Lauber, who helped me throughout with any question I had and motivated me to keep going.

To my professors and colleagues at the university who shaped my understanding and helped me form an inquisitive mind, some of you deserve more credit than I could ever express. To my friends who kept me grounded with a cold beer and letting me explain what it was I was doing when the equations got too abstract. I hope seeing your name on this list lets you know how thankful I am of having you in my life: Antonio Lopez Rivera (I genuinely don't know how much of this thesis is mine and how much is yours), Jaap Van Brussel, Ulf Torsten Kemmsies, Neil Richez, Jorge Gonzalez Navarro, Serban Blaga, Pablo Medina, Marcos Vazquez, Oliver Ross, Sergio Marin, Yotam Granov, Luuk Bartels, Mateo Giacomelli, Marcelo Santos, Neo Cass, Pedro Menegazzo, Manuel Arocena, Alessandro Crespi, Alessio Simoni and so many others.

Finally, and most importantly, to the ones who made all of this possible. I do not know who I would be without my family who supported me through every late night and early morning, I am forever indebted to you. Gracias Lau, Leo, Jose, Agus, Nacho, Facu, Luli, Mar, Tia Mariana, Tia Marina, Tia Coqui, Tio Renzo, Guadi, Barbara, Tio Dario. Son todo para mi. Mama, Papa todo esto es para ustedes.

Anibal Luciano Pastinante
Delft, September 2025

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

Abbreviations

Physics & Accelerators

Abbreviation	Definition
AD	Antiproton Decelerator
BCMS	Batch Compression, Merging and Splitting
BLonD	Beam Longitudinal Dynamics (code)
BLM	Bunch Lengthening Mode
BSM	Bunch Shortening Mode
CCC	CERN Control Center
CERN	European Organization for Nuclear Research (translated)
EM	Electromagnetic
EoM	Equations of Motion
FWHM	Full-Width Half-Maximum
ISOLDE	Isotope Mass Separator On-Line
JAPC	Java API for Parameter Control
LEIR	Low Energy Ion Ring
LHC	Large Hadron Collider
LINAC	Linear Accelerator
LLRF	Low-Level Radio-Frequency
nTOF	Neutron Time-of-Flight
PS	Proton Synchrotron
PSB	Proton Synchrotron Booster
PyJAPC	Python-Java API for Parameter Control
PyDA	Python Device Access
RF	Radio-Frequency
SFTPRO	SPS Fixed Target Proton
SPS	Super Proton Synchrotron
WCM	Wall Current Monitor

Machine Learning & Computing

Abbreviation	Definition
ADAM	Adaptive Moment Estimation
API	Application Programming Interface
BO	Bayesian Optimization
CAE	Convolutional Auto-Encoder
CBAM	Convolutional Block Attention Modules
CNN	Convolutional Neural Networks
DDPG	Deep Deterministic Policy Gradient
fANOVA	Functional Analysis of Variance
FIFO	First-In-First-Out (Buffer)
GP	Gaussian Process
GPU	Graphical Processing Unit
Grad-CAM	Gradient-weighted Class Activation Map
GUI	Graphical User Interface
HPC	High Performance Computing (cluster)
LSTM	Long-Short Term Memory

Continued on next page

Table 2 – continued from previous page

Abbreviation	Definition
MDP	Markov Decision Process
MDI	Mean Decrease in Impurity
MSE	Mean Squared Error
PER	Prioritized Experience Replay
PI	Probability of Improvement
POMDP	Partially Observable Markov Decision Processes
RL	Reinforcement Learning
RNG	Random Number Generator
SPS	Super Proton Synchrotron
TCN	Temporal Convolutional Networks
TD3	Twin Delayed Deep Deterministic Policy Gradient
TD	Temporal Difference (error)
TPE	Tree-structured Parzen Estimators
t-SNE	t-distributed Stochastic Neighbours Embedding

Symbols

Physics

Symbol	Definition	Typical Unit
$\mathcal{B}$	Magnetic Field	[T]
D	Dispersion Function	[m]
E	Energy	[eV]
E_s	Synchronous Particle Energy	[eV]
$\mathcal{E}$	Electric Field	[V/m]
FF	Filling Factor	[%]
h	Harmonic Number	[-]
I	Beam Intensity	$[\frac{\text{particles}}{\text{bunch}}]$
K	Kinetic Energy	[eV·s]
N_p	Number of Particles	[-]
N_m	Number of Macro-Particles	[-]
p	Momentum	[eV/c]
$U(\phi)$	RF Potential	[eV·s]
Q	Quality Factor	[-]
Q_s	Synchrotron Tune	[turns]
R	Mean Radius of Accelerator	[m]
R_{sh}	Shunt Impedance	[Ω]
s	Path Length	[m]
T_a	Transit Time Factor	[-]
t	Time	[s]
V_{sc}	Space-Charge Voltage	[V]
V_{rf}	RF Voltage	[V]
V_{total}	Total Voltage of all h_n	
v	Velocity	[m/s]
$W_{\parallel}$	Longitudinal Wakefield	[V/C]
$Z_{\parallel}$ Longitudinal Impedance		[Ω/C]
α_c	Momentum Compaction Factor	[-]
$\alpha_n, n = 0, 1, 2...$	Courant-Snyder Parameter α	[-]
β	Relativistic Beta	[-]
δ	Off-Momentum Variable	[-]
ϵ	Emittance	[eV·s]

Continued on next page

Table 3 – continued from previous page

Symbol	Definition	Typical Unit
η	Phase-slip Factor	[-]
$\mathcal{E}$	Electric Field	[V]
γ	Lorentz Gamma	[-]
γ_T	Transition γ	[-]
$\lambda_{t/\phi}$	Longitudinal Charge Density	[C/(t/ϕ)]
ω	Angular Frequency	[rad/s]
ω_s	Synchrotron Frequency	[rad/s]
Φ_h	RF System Phase Angle for Harmonic h	[rad or $\circ$] ϕ
Phase Angle	[rad or $\circ$]	
θ	Azimuthal Orbital Angle	[rad or $\circ$]

Machine Learning

Symbol	Definition	Typical Unit
B	Batch size	[number of samples]
$\mathcal{H}$	History Buffer	[-]
h_{len}	Length of History Buffer	[number of samples in history]
$J(\pi)$	Expected Cumulative Reward	
$\mathcal{L}$	Loss Function	[-]
n_{opt}	Number of Computation Points over the Cycle	[-]
n_{env}	Number of Parallelized Environments	[-]
Q	Critic Network	[-]
$\mathcal{R}$	Reward Function	[-]
r_{att}	Reduction Ratio for the Channel Attention Module	[-]
r_l	Learning Rate	[-]
r_{replay}	Replay Ratio	[-]
T_b	Decision Tree b	[-]
y	Objective Value of Hyperparameter Optimization	[-]
α	Spatial Attention Weights	[-]
$\Delta\Phi^*$	Optimal Phase Correction	[rad]
μ	Channel Attention Weights	[-]
π	Actor Policy	[-]
ϕ or θ	Model Weights (Parameters)	[-]

1 Introduction

At the heart of Europe's particle physics research infrastructure lies CERN, where the accelerator complex has continuously evolved to push the boundaries of scientific discovery. These machines are constantly improved to deliver higher intensities and brighter beams to experiments, creating new opportunities for groundbreaking research, but also presenting unprecedented technical challenges. Among the most demanding applications is the Isotope Mass Separator On-Line (ISOLDE) facility, which requires ultra-high-intensity (i.e. a high number of particles per bunch) proton beams to produce radioactive isotopes for nuclear physics research, and with an upgrade to ISOLDE on the horizon, even higher intensities are desired [10]. However, as beam intensities reach new heights, the limiting factor increasingly becomes space-charge effects that can destroy beam quality and prevent successful acceleration to higher energies. The term space-charge is used to describe the effect of electromagnetic forces between charged particles within a beam, causing, among other phenomena, transverse beam resonance effects, which ultimately lead to beam loss.

1.1. The CERN Accelerator Complex and Its Evolution

Understanding the context of space-charge challenges in modern accelerators requires appreciation of the sophisticated multi-stage acceleration process employed at CERN. The accelerator complex at CERN, shown in Figure 1.1, is an accelerator chain where each stage builds upon the previous to achieve higher particle energies.

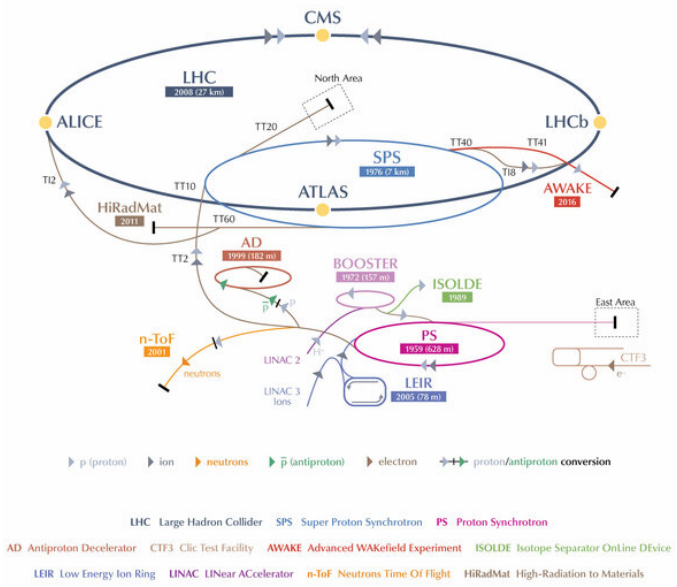


Figure 1.1: CERN Accelerator Complex [37]

The proton acceleration chain begins with LINAC (Linear Accelerator) 4, which serves as the first stage by accelerating protons from rest to 160 MeV before injection into the Proton Synchrotron Booster (PSB). This injection energy represents a significant increase from the original 50 MeV design, implemented as part of the LHC Injectors Upgrade (LIU) project specifically to combat space-charge limitations [20]. The PSB, comprising four superposed rings sharing magnetic cores but equipped with independent RF systems, accelerates beams up to 2 GeV of energy. The next stage is the Proton Synchrotron (PS), which achieves a final energy of 26 GeV before transferring beams to the Super Proton Synchrotron (SPS) with its 450 GeV capability. Finally,

the Large Hadron Collider (LHC) takes protons at this energy and accelerates them to approximately 6.8 TeV per beam, enabling collisions at a total centre-of-mass energy of 13.6 TeV.

Crucially, this complex serves not only the LHC but also a diverse array of experimental facilities through strategic beam extraction points. As illustrated in Figure 1.1, beams can be extracted through transfer lines at various stages for different experimental programmes. For instance, the PSB can extract at 1.4 GeV to supply the ISOLDE experiment (shown in green), which produces radioactive ion beams for nuclear physics research. The PS similarly serves the neutron Time-of-Flight (nTOF) facility, the Antiproton Decelerator (AD), and other fixed-target experiments.

The accelerator complex's evolution towards higher intensities reflects the physics community's demand for more precise measurements and access to rarer phenomena. For the PSB, space-charge has always been the limiting factor for high-intensity beam production, as at these high charge densities space-charge-induced phenomena become even stronger. As a result, understanding these limitations and developing solutions for them has become critical not only for current operations but also for future upgrades.

As a result, the current work focuses specifically on the PSB, where the high intensity accelerated at low particle energies makes the space-charge effects most severe. Each of the four PSB rings is equipped with sophisticated wideband RF systems utilising Finemet-loaded cavities [8]. The Finemet material provides exceptionally broadband impedance characteristics, enabling the cavities to support a wide frequency range required as the proton's relativistic velocity $\beta = v/c$ changes dramatically from approximately 53% to 94% the speed of light during which the proton's revolution frequency nearly doubles [3, 5].

This multi-harmonic capability is essential for the advanced longitudinal beam manipulation techniques that have become increasingly critical as intensity demands grow. The flexibility to operate with arbitrary voltage divisions between harmonics, without requiring cavity retuning, provides the foundation for bunch shaping strategies that form the core of this research.

1.2. Space-Charge Mitigation in Particle Accelerators

Space-charge effects represent one of the most fundamental limitations in low-energy, high-intensity particle accelerators. These effects arise from the electromagnetic forces between charged particles within a beam that change the oscillation frequencies of particles in the transverse plane to unwanted values, leading to resonant excitation. This can ultimately lead to beam loss or blow-up. The severity of space-charge effects scales approximately as $\lambda_t/\beta\gamma^2$ (see Chapter 2), where λ_t is the longitudinal charge density (also called longitudinal line density) and β and γ are the relativistic factors of the particles [3, 16].

To combat these detrimental effects, the CERN Proton Synchrotron Booster (PSB) has undergone significant upgrades to increase the injection energy provided by the linear accelerator (LINAC). The injection energy has been increased from the original 50 MeV to the current 160 MeV following the LHC Injectors Upgrade (LIU) project [8, 10]. This energy increase reduces space-charge effects by increasing the $\beta\gamma^2$ factor from $\beta\gamma^2_{50MeV} \approx 0.35$ to $\beta\gamma^2_{160MeV} \approx 0.71$, essentially allowing twice the beam intensity while still maintaining the same space-charge effects [10, 3, 16].

However, there are practical limitations to how much energy can be gained in a LINAC due to cost, the fact that particles only go through the LINAC once, and technical complexity. Therefore, alternative approaches must be pursued to further mitigate space-charge effects. This has led to research into other solutions, such as varying the magnetic field ramp profiles to increase particle energy as rapidly as possible, thereby reducing the time spent in the high space-charge regime where these effects are most detrimental. The degree to which the ramp can be modified is constrained by main-magnet circuit cooling and stability considerations, which also prompts the search for other solutions [3].

As a result, the challenge extends beyond simple parameter optimisation to encompass complex multi-harmonic Radio-Frequency (RF) manipulation strategies. Since β and γ are set by the invariant injection energy, a primary alternative strategy involves minimising the longitudinal charge density λ_t while simultaneously attempting to increase the total number of particles—objectives that are inherently competing and require optimisation. This can be achieved by adding higher-harmonic voltages with controlled amplitudes and relative phases to reshape the RF potential well and flatten the longitudinal charge density as much as possible while

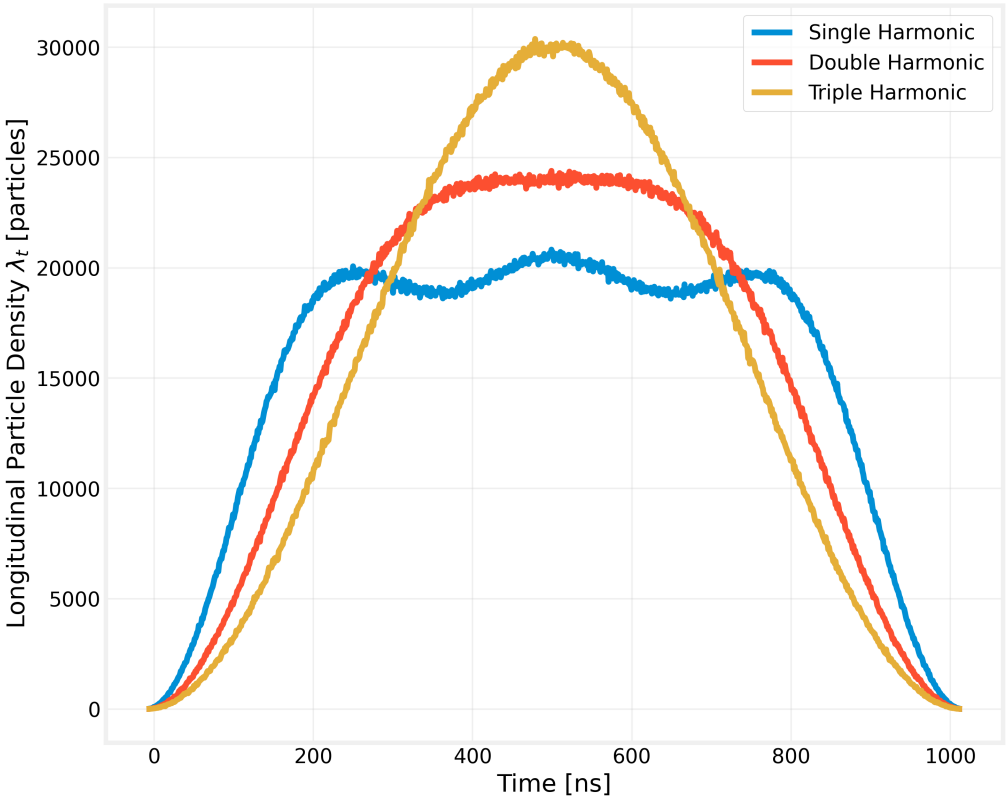


Figure 1.2: Impact of multi-harmonic operation on simulated longitudinal bunch profiles with the same intensity

still maintaining the same intensity, as shown in Figure 1.2. This approach is used operationally in the PSB to prepare the beam for the ISOLDE experiment and also LHC beams to reduce the transverse cross-section of the beam, but requires careful tuning of multiple amplitudes and phases and is sensitive to beam intensity changes, path differences in the wiring of the RF system, and the energy gained per turn [24, 50].

1.2.1. Longitudinal Bunch Profile Optimisation

As a result, the quest to minimise space-charge effects while maximising beam intensity has led to a focus on flattening the longitudinal bunch profile as much as possible. This approach recognises that reducing peak particle density through profile shaping can significantly mitigate space-charge effects without sacrificing total beam intensity. This strategy is currently employed not only in the PSB, albeit through time-intensive manual optimisation processes, but also in other accelerators facing similar or more severe space-charge limitations, such as the Rapid Cycling Synchrotron (RCS) at the Japan Proton Accelerator Research Complex (J-PARC) [42].

In the PSB, longitudinal bunch shaping is achieved through sophisticated multi-harmonic RF systems utilising wideband Finemet cavities operating at multiple harmonics simultaneously, with an arbitrary voltage division between harmonics without the need to tune the cavity to a specific harmonic [3, 5]. This is due to the material properties of the Finemet, and is ideal for the PSB as the proton's revolution frequency changes significantly during acceleration as the speed of the proton changes from $\sim$ 53% of the speed of light to $\sim$ 94% during acceleration.

As mentioned earlier, the current optimisation process for triple-harmonic operation requires manual adjustment of five critical parameters: three voltage amplitudes (V_1,V_2,V_3) and two relative phases (Φ_2, Φ_3) for the harmonic RF systems. Meanwhile, double-harmonic operation requires the adjustment of 3 parameters (no V_3 and Φ_3). This labour-intensive procedure can take 10-20 minutes for double-RF optimisation and 30-40 minutes for triple-RF optimisation. This represents a significant operational burden and limits the accelerator's ability to adapt to varying beam conditions without the need to re-optimise the parameters when the beam conditions change, making the automation of this optimisation very attractive.

Indeed, the PSB represents an ideal testing ground for developing and validating new automated approaches to this optimisation challenge. The facility's well-characterised beam dynamics, comprehensive bunch profile measurements, and the Finemet cavity's flexibility make it an excellent platform for implementing automatic control systems. The potential benefits extend far beyond operational efficiency, as automated optimisation could enable real-time adaptation to beam loading effects, machine drift, and varying cycle requirements.

1.3. Problem Statement and Solution Roadmap

The current manual optimization approach for longitudinal bunch profile control in the PSB faces several critical limitations. First, the optimization process is extremely time-consuming, limiting the frequency with which parameters can be adjusted within reason (considering the effort implied on RF experts who seek to optimize other aspects of the accelerator). Second, the manual approach cannot adapt in real-time to dynamic effects such as RF system drift, changes in the control loops, or changes in the beam intensity. Third, the labor-intensive nature of the process prevents exploration of optimal settings for different operational scenarios, potentially leaving performance improvements unrealized.

These limitations become increasingly problematic as the demand for higher intensity beams continues to grow, particularly for ISOLDE operations where beam quality and intensity directly impacts the production rates of exotic isotopes, with higher intensities yielding rarer isotopes [10]. The need for a real-time, automated solution that can track optimal harmonic phase settings under varying beam-loading conditions and machine drift has become critical for the facility's future operational success, while also reducing the workload on the accelerator operators.

1.3.1. Operational Requirements Beyond Space-Charge Mitigation

Although triple-harmonic operation is a partial solution to space-charge mitigation through longitudinal profile flattening [10], the PSB's operational diversity demands a more comprehensive approach to automated RF control. Double-harmonic operation, utilising harmonics $h = 1$ and $h = 2$, forms the foundation of numerous beam production schemes that extend beyond only space-charge reduction.

For example, the neutron Time-of-Flight (nTOF) beam production presents entirely different operational requirements [38, 2]. The nTOF facility requires ultra-short, high-intensity proton pulses (7 ns root-mean-square (RMS) width after PS extraction) delivered to a lead spallation target to generate neutrons for nuclear physics experiments [38]. For nTOF operations, the optimisation objective in the PSB is initially space-charge mitigation due to the high intensity, but then by the end of the cycle, a jump in the fundamental harmonic voltage ($h = 1$) is used to improve the transfer of the beam to the PS.

However, this changes for the AD beam production, where the AD requires a high-intensity proton beam to be delivered to a target to generate antiprotons for nuclear physics experiments [34]. For AD operations, there are different extraction schemes per ring, which are not related to space-charge mitigation. To illustrate, rings 1, 2 and 4 extract at the fundamental harmonic ($h = 1$), while ring 3 extracts at the second harmonic ($h = 2$) to produce 2 bunches. This is done in order to capture the 5 bunches in the PS at the sixth harmonic ($h = 6$) and then perform manipulations and improve antiproton production [34].

1.3.2. Solution Approach and Implementation Roadmap

Given this operational complexity, a progressive implementation approach is essential. Double-harmonic phase optimisation must be performed first before attempting to tackle the more complex triple-harmonic systems. This progression is driven by several critical factors:

Operational Prevalence: Double-harmonic operation represents the backbone of PSB operations, being used across the majority of beam types including Batch Compression, Merging and Splitting (BCMS) and nTOF, and controlled longitudinal emittance blow-up procedures [5, 2].

Parameter Space Complexity: Double-harmonic operation optimisation involves only one primary parameter: the relative phase between the fundamental and second harmonic, ϕ_2. Only the phase must be optimised as the RF experts still have to define the amplitudes of the voltages depending on the purpose of the beam. This makes it an ideal proving ground for automatic optimisation techniques. Success in this reduced parameter space provides essential validation before tackling the five-dimensional optimisation space of triple-harmonic

operation (three amplitudes and two relative phases).

Generalisability Requirements: An automated system must demonstrate robustness across diverse operational scenarios, from space-charge mitigation (bunch lengthening) to beam compression (nTOF), to bunch splitting (AD). Double-harmonic systems provide the broadest operational test bed for validating generalisation capabilities of the proposed optimiser, rather than only focusing on the space-charge mitigation.

Accuracy Constraints: The sensitivity of double-harmonic operation to the phase imposes stricter accuracy requirements than what is required for sufficient space-charge mitigation. Demonstrating sub-degree accuracy and sub-super-cycle latency in double-harmonic control establishes the foundation for the more complex triple-harmonic implementations.

The implementation roadmap therefore follows a logical progression: establish automatic control for double-harmonic operations across all operational scenarios, validate generalization capabilities and accuracy, then extend the methodology to encompass triple-harmonic systems for space-charge mitigation.

1.4. Research Questions

As a consequence of the complex operational environment and diverse beam production requirements in the PSB, this thesis addresses the following fundamental research question:

Primary Research Question: How can an automated tool for the phase (and amplitude) computation of the second and third RF harmonics in the CERN Proton Synchrotron Booster provide robust, real-time optimization across diverse operational scenarios while ensuring operational safety and maintainability?

Furthermore, to address key aspects concerning the feasibility, performance, safety, maintainability and scalability of the developed solution, the following sub-questions are used to guide the research and development process:

Sub-questions:

1. **Technical Feasibility and Performance:**

R.Q.1: Can machine learning approaches accurately predict optimal harmonic parameters from bunch profile measurements across different beam types (ISOLDE, nTOF, BCMS, etc.) with the precision required for operational deployment?

R.Q.2: How do automated optimization techniques perform when confronted with the diverse operational parameter space?

R.Q.3: What is the minimum response time achievable for real-time harmonic control while maintaining the required accuracy standards for operational deployment?

2. **Operational Robustness and Generalization:**

R.Q.4: How effectively can automated approaches adapt to different PSB operational cycles under varying beam intensities, injection conditions, and machine states?

R.Q.5: What level of robustness can be achieved against common operational variations such as beam loading effects, path-delay miscalibrations, and transient machine conditions?

R.Q.6: How do automated optimization methods perform in dynamic operational scenarios compared to traditional manual optimization approaches?

3. **Safety and Fail-Safe Operation:**

R.Q.7: What are the failure modes of automated approaches when deployed in operational accelerator environments, and how can they be detected and mitigated?

R.Q.8: How can the system ensure safe operation when encountering out-of-distribution scenarios or unex-

pected beam conditions not seen during development?

R.Q.9: What safeguards and fallback mechanisms are required to prevent beam loss or equipment damage during automated operation?

4. **Integration and Maintainability:**

R.Q.10: How can an automated system be integrated with existing PSB control infrastructure while maintaining operational transparency and operator oversight?

R.Q.11: What strategies ensure system maintainability and adaptability to future accelerator upgrades or changes in operational requirements?

R.Q.12: How can automated approaches be validated and benchmarked against expert operator performance across the full range of operational scenarios?

5. **Scalability and Technology Transfer:**

R.Q.13: Can the developed methodology scale from single-parameter optimization to full multi-dimensional harmonic control while maintaining performance and safety standards?

R.Q.14: How transferable are automated optimization approaches to other accelerator facilities with similar space-charge limitations, such as the Low Energy Ion Ring (LEIR)?

1.5. Project Goals

The primary objective of this research is to develop, validate, and deploy an automated control system for longitudinal bunch profile optimization in the CERN PSB that meets the stringent requirements of operational accelerator environments. The project goals are structured in progressive phases that are in line with answering the previously stated questions:

Core Technical Development Goals

P.G.1: Develop robust automated optimization methods capable of determining optimal RF system parameters from bunch profile measurements with sufficient accuracy across all operational beam types.

P.G.2: Implement adaptive optimization algorithms for continuous multi-dimensional RF parameter control, demonstrating stability and adaptation to time-varying beam conditions.

P.G.3: Achieve real-time operational performance with response times suitable for operational deployment (latency $<$ 1 super-cycle, approximately 30 seconds) while maintaining accuracy requirements.

P.G.4: Establish scalable optimization framework progressing systematically from single-parameter control to full multi-harmonic amplitude and phase optimization for comprehensive space-charge mitigation.

Validation and Verification Goals

P.G.5: Comprehensive simulation validation using physics-accurate models across the complete operational parameter space, including collective effects, beam loading, and realistic machine imperfections.

P.G.6: Machine validation and benchmarking against expert operator performance using actual PSB operational data across multiple beam types and operational conditions.

P.G.7: Robustness demonstration under realistic operational variations including intensity variations, injection transients, and changes in machine configuration.

P.G.8: Safety validation through comprehensive testing of failure modes, edge cases, and recovery mechanisms to ensure beam preservation under all conditions.

Integration and Deployment Goals

P.G.9: **Control system integration** with full operational deployment capability, including operator interfaces, monitoring systems, and manual override capabilities through existing infrastructure.

P.G.10: **Comprehensive safety protocol implementation** with safeguards including beam loss detection, parameter bounds checking, automatic fallback to manual control, and emergency stop capabilities.

P.G.11: **Operational transparency and interpretability** providing operators with clear insight into automated decision-making processes and confidence metrics for operational acceptance.

P.G.12: **Documentation and training framework** enabling technology transfer to operational teams and ensuring long-term system maintainability.

Technology Transfer and Scalability Goals

P.G.13: **Cross-facility applicability assessment** evaluating approach transferability to other CERN accelerators (LEIR), demonstrating broad methodology applicability.

P.G.14: **Future-proofing and adaptability** ensuring the optimization framework can accommodate future accelerator upgrades, new beam types, and evolving operational requirements without fundamental redesign.

P.G.15: **Performance benchmarking establishment** creating evaluation criteria and performance metrics for automated accelerator control.

1.6. Scope and Approach

This research follows a systematic, safety-first progression from controlled simulation environments to full operational deployment. The approach emphasizes validation at each stage and incorporates operational requirements from the earliest development phases.

The methodology begins with comprehensive physics modelling to ensure accurate representation of PSB longitudinal dynamics, including space-charge, beam loading, and impedance effects. Additionally, the progression from single-parameter control to full multi-harmonic optimization provides incremental validation opportunities while building operational confidence.

The verification and validation pathway emphasizes three key stages: simulation verification under idealized conditions, simulation validation with realistic machine imperfections, and finally machine validation using actual operational data. Each stage includes comprehensive safety testing and failure mode analysis.

The scope explicitly encompasses operational deployment considerations including control system integration, fail-safe protocols, operator training, and long-term maintainability. The potential for technology transfer is evaluated through benchmarking and the assessment of the generalizability of the approach.

1.7. Thesis Structure

This thesis is organized to provide comprehensive treatment of automated harmonic control for operational accelerator applications, structured as follows:

- **Chapter 2: Beam Dynamics Background** establishes the theoretical foundation: particle accelerator fundamentals, longitudinal beam dynamics, collective effects and impedance, and the simulation framework used in this work. It also studies the PSB-specific longitudinal profiles, BlonD code components, and caveats in bunch-profile measurement that impact simulation accuracy.

- **Chapter 3: Requirements Analysis & Definition** derives system requirements from stakeholder needs, operational beam types, technical constraints and risk analysis. It defines the full set of system requirements, the verification & validation methodology, the verification criteria for the requirements, and the requirements traceability to the proposed solution.

- **Chapter 4: Machine Learning Background** details dataset creation methodology and constraints (simulated vs. experimental, temporal vs. static), the selected AI frameworks, Convolutional Neural Networks (CNN) with attention, stratified sampling strategies, reinforcement learning (RL) and memory-augmented RL, possibilities of augmented feature extraction, and hyperparameter optimization with functional Analysis of Variance (fANOVA).

- **Chapter 5: Double-Harmonic Operation: Φ_2 Optimization** presents the complementary solutions:a CNN with attention mechanisms, and a memory-augmented RL agent. Both solutions are presented and discussed along with their training considerations, possibilities of autoencoder-assisted feature extraction, hyperparameter optimization, model and verification analyses, data acquisition & preprocessing, and a validation procedure with PSB results.

- **Chapter 6: Triple-Harmonic Operation: 5D Optimization** extends to full five-dimensional triple-harmonic control with the memory-augmented RL agent, including training, verification analysis in simulation for space-charge mitigation, and a PSB implementation roadmap.

- **Chapter 7: Conclusions & Recommendations** evaluates outcomes against the research questions, project goals, and requirements. It also provides recommendations for future work and states the overall conclusions.

The thesis concludes with comprehensive assessment of the methodology's impact on accelerator operations, lessons learned from operational deployment, and recommendations for future automated control applications in accelerator systems.

2

Beam Dynamics Background

This chapter establishes the principles of acceleration and outlines longitudinal beam dynamics. Additionally, synchrotron motion under multi-harmonic operation is discussed together with the theoretical principles for double-harmonic particle motion and its different modes under constant RF system parameters. Naturally, this is followed by the theoretical principles of triple-harmonic particle motion, also under constant parameters. To demonstrate these concepts, idealised tracking code results are presented to showcase how this motion looks under double- and triple-harmonic operation. This sets the stage to introduce the real-world effects particle accelerators face, such as beam loading and space-charge, including how they relate to the problem statement. Then, an analysis of the profiles produced by PSB operational cycles and their statistical features is performed to identify possible optimisation routes. Finally, this chapter ends by presenting the simulation code and model for the PSB, and its real-world fidelity is discussed.

2.1. Accelerator Fundamentals

Particle acceleration relies on the manipulation of electromagnetic (EM) fields to control the motion and energy of charged particles. In synchrotrons, both magnetic and electric fields play critical roles, with magnetic fields guiding particles along a desired path, while electric fields provide the energy increase. This section outlines the fundamentals of this process, emphasising the interplay between these fields and the role of RF cavities, without needing to specify the type of accelerator.

2.1.1. Lorentz Force: Steering and Acceleration

The behaviour of charged particles in electromagnetic fields is governed by the Lorentz force law, which states that a charged particle with charge q and velocity $\vec{v}$ experiences a force given by [19]:

$$\vec{F} = q(\vec{\mathcal{E}} + \vec{v} \times \vec{\mathcal{B}}), \tag{2.1}$$

where $\vec{\mathcal{E}}$ represents the electric field and $\vec{\mathcal{B}}$ represents the magnetic field.

In the presence of a magnetic field $\vec{\mathcal{B}}$, the term $q\vec{v} \times \vec{\mathcal{B}}$ results in a force that is always perpendicular to the particle's velocity $\vec{v}$. This means that while static magnetic fields can change the direction of the particle's motion, they cannot alter its speed or kinetic energy. As a result, magnetic fields alone are not capable of accelerating particles in the direction of their motion; they only provide the force needed to bend particle trajectories.

For acceleration, where an increase in particle speed or energy is desired, a parallel force component is necessary. This is achieved through the application of electric fields, where the force $q\vec{\mathcal{E}}$ directly influences the particle along the direction of the electric field. When the electric field aligns with the particle's velocity, it performs work on the particle, thereby increasing its kinetic energy [19].

In particle accelerators, RF cavities are used to confine oscillating electric fields that provide this acceleration. These RF fields are synchronized with the particle beam's motion, ensuring that particles gain energy at each passage through the cavity. This will be discussed more in detail later on in Section 2.2.

2.1.2. Role of RF Cavities in Particle Accelerators

RF cavities are essentially volumes of empty space where the boundary is conductive and whose geometries are specialised for creating the resonant electric field to accelerate particles when exposed to an RF wave. This RF wave resonates inside the cavity due to the conductive walls, generating an electric field that resonates because of the application of Maxwell's equations considering the specific boundary conditions. The exact properties of

the electric field depend on the frequency of the RF wave, its amplitude and the material characteristics of the cavity walls: their conductivity, their inductance, and other material characteristics such as the smoothness of the walls. These factors will determine how electromagnetic waves reflect within the cavity and establish standing-wave patterns, leading to the formation of resonant modes. There are also cavities that can generate travelling waves that *travel* with the particles they accelerate, so they always see the same accelerating field as they travel through it, which are used in accelerators like the SPS [19]. However, in the PSB only standing-wave cavities are employed, so the discussion will be limited to these.

The cavities usually operate at single frequencies, tuned to match the beam's motion, allowing them to provide energy consistently as the particles circulate through the accelerator. As mentioned previously, the PSB has broadband cavities, meaning they can operate at a range of frequencies. The oscillating electric fields in these cavities impart small energy increments to the particles on each pass, gradually increasing their momentum.

The efficiency of energy transfer from the cavity to the beam depends on maintaining a precise phase relationship between the oscillating RF field and the circulating particles. This ensures that the particles are accelerated at the right moment, optimising the energy gain per turn and maintaining beam stability [30]. There are other particularities of the geometry of the cavities that need to be considered in order to excite only the desired modes, but this is out of scope for this project.

2.1.3. RF Cavities as RLC Circuits and Impedance

Each resonant mode of an RF cavity can be modelled as an equivalent RLC (Resistor–Inductor–Capacitor) circuit to understand its electromagnetic properties and energy transfer characteristics [19]. This circuit is shown in Figure 2.1. This model reduces the interactions of the beam with the cavity and its behaviour to resonance and *impedance*. Key parameters in this model include:

- **Resonance Angular Frequency (ω_r)**: The frequency at which the cavity naturally oscillates, given by:
$$\omega_r = \frac{1}{\sqrt{LC}}, \tag{2.2}$$
where L is the inductance and C is the capacitance.

- **Quality Factor (Q)**: A measure of the cavity's ability to store energy relative to energy lost per cycle, defined as:
$$Q = 2\pi \frac{\text{energy stored}}{\text{energy dissipated}} = \frac{\omega_r L}{R}, \tag{2.3}$$
where R represents the resistive losses in the cavity (i.e. the resistor R in Figure 2.1).

- **Shunt Impedance (R_{sh})**: Indicates how efficiently the cavity transfers power to the beam, defined as:
$$R_{sh} = \frac{|V|^2}{2P_{in}}, \tag{2.4}$$
where V is the peak voltage achieved in the cavity and P is the power delivered to the cavity. This means that for a higher shunt impedance, for a given input power, a higher accelerating voltage V can be achieved.

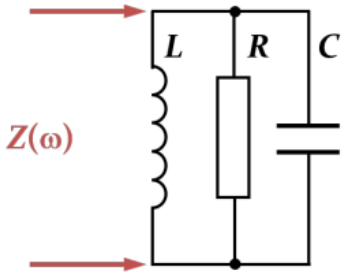


Figure 2.1: Cavity modelled as an equivalent LRC circuit [19]

The impedance, Z, of the cavity, which describes its response to the beam current, is a complex function:

$$Z(\omega) = \left(\frac{1}{R} + \frac{1}{i\omega L} + i\omega C \right)^{-1}, \tag{2.5}$$

where ω is the angular frequency of the input signal. Additionally, because of this representation, it can be said that when an impedance has the form $Z = -iC$, it is called a *capacitive* impedance, while it is called *inductive* if it has the form $Z = iL$ (with C and L being constant). This concept of impedance relates to the shunt impedance, as the power input into the cavity is related to the voltage and impedance at resonance as follows:

$$P_{in} = \frac{V^2}{Z(\omega)_{\text{resonance}}} = V^2 \left(\frac{1}{R} + \frac{1}{i\omega L} + i\omega C \right) = P_{\text{loss}} + 2\omega i \underbrace{(W_e - W_m)}_{=0 \text{ at resonance}} \tag{2.6}$$

where $P_{\text{loss}} = V^2/R_{sh}$. However, this theory, and specifically Equation 2.5, describes the impedance for *resonant* modes of cavities, but as stated in Chapter 1, the Finemet cavities in the PSB are broad-band cavities. This means they support a broad range of frequencies and, thus, cannot be modelled by this resonant-cavity description, but they still have an impedance, which will be shown in Section 2.3. It is also possible to model other real-world effects related to the accelerator geometry, or the interaction of the particles with themselves (known as *collective effects*) as impedances that add upon the impedance of the cavity as will also be shown in Section 2.3. Now that the basics of longitudinal particle acceleration have been outlined, it is possible to deepen the analysis into the dynamics of particles during acceleration.

2.2. Longitudinal Beam Dynamics

In order to be able to assess the performance of the final optimiser, it is necessary to have the relevant background on longitudinal beam dynamics. To do so, it is even more imperative to establish how the RF cavities synchronise with the beam passage to provide energy to the particles circulating within it. First, an introduction to synchrotrons is given and how they confine the beam using the EM fundamentals described previously. Then, the dynamics of the particles in the longitudinal direction are characterised through the synchrotron equations of motion (EoM) which describe the evolution of the particles. This is followed by analysing synchrotron EoMs under small-amplitude approximations to gain more insight into the type of motion the particles undergo and the regions of stability in phase space. Finally, all these concepts are integrated by analysing the results of a simplified tracking code which uses the full, non-linear EoM to describe the evolution of the particles.

2.2.1. Synchrotrons

Now the discussion turns to particle acceleration in synchrotrons. The synchrotron is a circular accelerator which uses EM fields to provide an energy gain to a group of particles, known as a beam, and also confine this beam in a closed orbit. The electric field generated in the RF cavities is responsible for accelerating (gaining kinetic energy) the particle through the Coulomb force ideally parallel to the direction of propagation. The magnetic field provides a centripetal force, through the Lorentz force on charged particles, to keep the beam in a quasi-circular orbit. The magnetic fields are provided by electromagnets whose beam-directing properties depend on the geometry and quantity of magnetic poles. The discussion of the magnets is limited to only the bending of particle trajectories within the plane of the accelerator (i.e. no consideration of focusing, defocusing, nor chromatic aberration correction of the beam; for a full treatment of these effects see [19, 30]). As is expected from Equation 2.1, the fact that a particle's transverse control, through magnetic fields, depends on its longitudinal dynamic properties (i.e. its speed), there should be some coupling between the longitudinal beam dynamics and the transverse one. Nevertheless, it is sufficient to consider a decoupled system without much loss of accuracy [30].

The bending of trajectories within the plane of the accelerator can be analysed by equating the centripetal force with the magnetic component of the Lorentz force:

$$m\frac{v_\parallel^2}{\rho} = qv_\parallel \mathcal{B} \longrightarrow p = mv = q\mathcal{B}\rho \tag{2.7}$$

where ρ is the bending radius of the dipoles (not the radius of the synchrotron), and $\mathcal{B}$ is the magnetic field

strength pointing into the plane of the accelerator for positively charged particles and out of the plane for negatively charged particles. Thus, the magnetic field strength is related to the momentum of the particle and, by taking the time derivative of Equation 2.7, the rate of change of the magnetic field is related to the rate of change of momentum, meaning the accelerating force. The important thing to remember here is that the rate of change of the magnetic field will determine the rate of change of the particle's energy through the relativistic energy equation (E_k is the kinetic energy and E_0 is the particle's rest energy) [19]:

$$E = \sqrt{p^2c^2 + m_0^2c^4} = E_k + E_0 \longrightarrow \dot{E} \propto \dot{p} = q\dot{\mathcal{B}}\rho \tag{2.8}$$

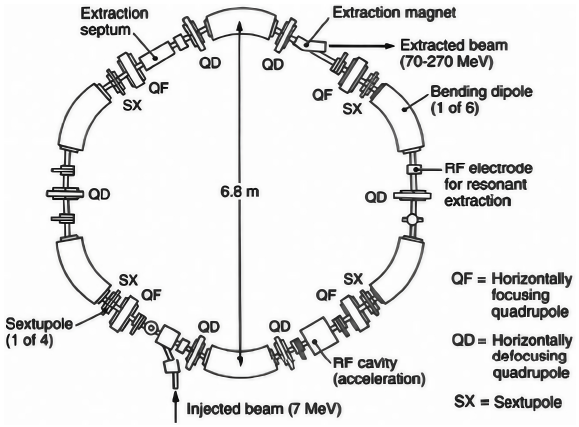


Figure 2.2: Simplified schematic of a synchrotron [49]

In reality, synchrotrons are not perfectly circular machines ($\rho \neq R_0 = C_{\text{ring}}/2\pi$), but are comprised of straight sections and curved sections. Straight sections are where the beam is accelerated, focused transversely (using higher-order magnets known as quadrupoles and sextupoles), extracted, among other things that interact with the beam (collimators, beam dumps, detectors, etc.). Meanwhile, in curved sections the beam is directed using the dipole magnets as shown in Figure 2.2, and possibly focused as well. Furthermore, for the entirety of this work, the Frenet–Serret coordinate system is used, which is defined in Figure 2.3. Since the majority of the work focuses on longitudinal beam dynamics, the main direction concerning the discussion is $\hat{s}$, which is defined as the direction perpendicular to the radius of the orbit $\vec{r}(s)$ and in the plane of the orbit, where s is the position along the orbit.

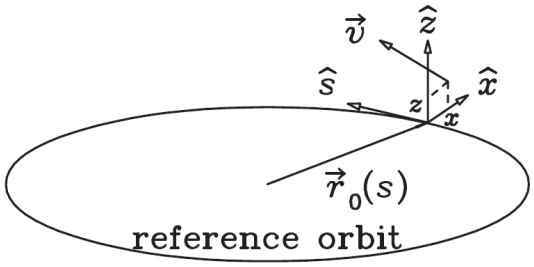


Figure 2.3: The Frenet-Serret coordinate system used in accelerator physics [30]

Moreover, given that the Lorentz force will, by definition, always be perpendicular to the velocity and that the electric field is parallel to the path taken by the particles and is only present at the RF cavities, the energy gain can be written as (see [50]):

$$(\Delta E)_{\text{gain}} = \oint_{\text{path}} \boldsymbol{F} \cdot \boldsymbol{ds} = q \oint_{S_{\text{ct}}} \mathcal{E} \cdot \boldsymbol{ds} + q \int_0^{T_0} \boldsymbol{v} \times \boldsymbol{B} \cdot \boldsymbol{v} dt = q \int_{\text{RF cavities}} \mathcal{E} \cdot \boldsymbol{ds}, \tag{2.9}$$

where $(\Delta E)_{\text{gain}}$ is the energy gain per turn for a given particle and T_0 is the revolution period. Note that,

ultimately, the only contribution to the energy gain comes from the electric field in the RF cavities and how well aligned the resonating electric field is with the direction of propagation of the particles (denoted by the dot product). This equation can be developed further by specifying the time-dependence of the electric field and assuming only one RF cavity:

$$\mathcal{E}(t) = \hat{\mathcal{E}} \sin\left(\omega_{\mathrm{rf}} t + \phi_{\mathrm{rf}}\right) \tag{2.10}$$

where ω_{rf} and ϕ_{rf} are the RF angular frequency and phase, respectively. This can now undergo a variable change from time to the distance s covered by the particle, i.e. $t = s/v$, giving rise to the following energy gain for a particle after transiting the length of the cavity l_{gap}:

$$(\Delta E)_{\mathrm{gain}} = q \int_{-\frac{l_{\mathrm{gap}}}{2}}^{\frac{l_{\mathrm{gap}}}{2}} \mathcal{E}(s) ds = q\hat{\mathcal{E}} \int_{-\frac{l_{\mathrm{gap}}}{2}}^{\frac{l_{\mathrm{gap}}}{2}} \sin\left(\frac{\omega_{\mathrm{rf}}}{v} s + \phi_{\mathrm{rf}}\right) ds \tag{2.11}$$

As mentioned before, in the PSB a standing wave is created in the cavity, so the particle sees different phases along its passage through the cavity. Developing the equation further yields (using the trigonometric identity $\sin(a+b) = \sin a \cos b + \sin b \cos a$ and the fact that $\int_{-x}^{x} \sin b db = 0$):

$$(\Delta E)_{\mathrm{gain}} = q\hat{\mathcal{E}} \int_{-\frac{l_{\mathrm{gap}}}{2}}^{\frac{l_{\mathrm{gap}}}{2}} \cos\left(\frac{\omega_{\mathrm{rf}}}{v} s\right) \sin\phi_{\mathrm{rf}} ds = \frac{2q\hat{\mathcal{E}} v \sin\phi_{\mathrm{rf}}}{\omega_{\mathrm{rf}}} \sin\left(\frac{\omega_{\mathrm{rf}} l_{\mathrm{gap}}}{2v}\right), \tag{2.12}$$

which can also be rewritten by defining the transit time factor

$$T_{\mathrm{a}} = \frac{\sin\left(\frac{\omega_{\mathrm{rf}} l_{\mathrm{gap}}}{2v}\right)}{\frac{\omega_{\mathrm{rf}} l_{\mathrm{gap}}}{2v}} \tag{2.13}$$

that represents the energy gain lost due to the finite length of the cavity, and the RF voltage $V_{\mathrm{rf}} = \frac{2\mathcal{E}v}{\omega_{rf}}$:

$$(\Delta E)_{\mathrm{gain}} = q V_{\mathrm{rf}} T_{\mathrm{a}} \sin\phi_{\mathrm{rf}}, \tag{2.14}$$

Nevertheless, $\omega_{\mathrm{rf}}\, l_{\mathrm{gap}}\, /(2v) \propto l_{\mathrm{gap}}\, / C_{\mathrm{ring}}$, so if $l_{\mathrm{gap}} \ll C_{\mathrm{ring}}$, which is the case for the PSB, then $T_a \approx 1$. This comprises the fundamental physics of particle acceleration, but this still needs to be translated to the beam dynamics world, which implies equations of motion (EoM) are required.

2.2.2. Longitudinal Equations of Motion

The term equations of motion refers to mathematical equations that describe the behaviour of physical systems in terms of their motion over time. In the following derivations, the equations of motion of the single and multi-harmonic system are defined through the use of a so-called "synchronous" particle (this does not need to be the case), which is an idealized virtual particle that has the desired momentum at all times throughout the acceleration ramp (see [50, 30] for full treatment). However, it is the magnetic field that defines the momentum of the synchronous particle over time through Equation 2.7 and its time derivative, while taking into account the geometry of the synchrotron. The following derivations are modified and expanded on from the derivations presented in literature (see [30, 15]).

Single-Harmonic System

From this point onwards, any parameter which has an s as its subscript is referring to the **synchronous particle**, with the exception of the revolution frequency, which for the synchronous particle is w_0. It is immensely beneficial to define these synchronous parameters in the derivation of the EoM because the synchronous particle synchronizes with the RF wave used to accelerate it with frequency $\omega_{rf} = h\omega_0$ where h is an integer representing the harmonic of the revolution frequency. This means that as the particle is passing through the

cavity, it will always synchronize with a particular RF phase angle $\phi_{rf} = \phi_s$ every turn, the value of which is computed depending on the voltage and energy gain desired for that turn (which is derived from the magnetic field). Thus, the energy change at turn n is (from Equation 2.14):

$$(\Delta E_s)_n = qV_{\text{rf}} \sin \phi_{s_n} = qV(\phi_s) \tag{2.15}$$

where now $q = e$ for a proton, and $V(\phi_s)$ is the accelerating voltage felt by the synchronous particle. In order to find the rate of change of energy for the particles, it is necessary to define the parameters of the non-synchronous, real particles in the accelerator:

$$\begin{cases} \omega = \omega_0 + \Delta\omega \text{ (ang. rev. frequency)}, \quad \phi = \phi_{\text{s}} + \Delta\phi \text{ (rf phase)}, \quad E = E_s + \Delta E \text{ (energy)} \\ \theta = \theta_{\text{s}} + \Delta\theta \text{ (azimuthal orbital angle)}, \quad p = p_s + \Delta p \text{ (momentum)}, \end{cases} \tag{2.16}$$

Using these parameters facilitates defining the EoM of the particles relative to the longitudinal motion and energy increase of the synchronous particle, which is more intuitive when defining what the particles are doing relative to our "ideal" particle. Again, note that the revolution frequency is defined as ω_0 rather than ω_s in order to avoid confusion for reasons which will become clear later on. The first step in the derivation is establishing the relationship between $\Delta\phi$ and $\Delta\theta$, which are related through the harmonic as θ oscillates with an angular frequency ω, while ϕ oscillates with $h\omega$:

$$\Delta\phi = \phi - \phi_{\text{s}} = -h\Delta\theta \tag{2.17}$$

In addition, one can realize that the frequency offset, $\Delta\omega$, defines the rate of change of the orbital angle offset, $\Delta\theta$, as θ_s oscillates with the synchronous angular frequency, giving:

$$\Delta\omega = \frac{d}{dt}\Delta\theta = -\frac{1}{h}\frac{d}{dt}\Delta\phi = -\frac{1}{h}\frac{d\phi}{dt}. \tag{2.18}$$

This equation already provides (partially) the time evolution of one of the two *phase-space* coordinates, ϕ, but the rate of change of the energy difference relative to that of the synchronous particle, $(\dot{\Delta E})$, must also be defined. In longitudinal beam dynamics, phase-space is a conceptual coordinate system used to describe the state of a bunch of particles in the variables of ΔE and ϕ (or any other equivalent coordinates). For this, one can look at the acceleration rate given by the energy gain *per turn* when transiting the RF cavity and multiplying by $\frac{\omega}{2\pi}$ to make the coordinate transformation from per turn to *per second*: $(\dot{\Delta E}) = \frac{\omega}{2\pi} eV_{\text{rf}} \sin\phi$. Then, by subtracting the energy gain rate of the synchronous particle in Equation 2.15 divided by its angular frequency, $\frac{\Delta \dot{E}_0}{\omega_0}$, and putting the particle's angular frequency within the time derivative, one can solve for $\frac{d}{dt}\left(\frac{\Delta E}{\omega_0}\right)$ by using the following relation (and the relations in Equation 2.16):

$$\frac{1}{\omega}\dot{E} - \frac{1}{\omega_0}\dot{E}_s = \frac{1}{\omega_0}\Delta\dot{E} - \dot{E}\frac{\Delta\omega}{\omega_0^2} \approx \frac{1}{\omega_0}\Delta\dot{E} + \left[\dot{E}\frac{\Delta\left(1/\omega_0\right)}{\Delta E}\right]\Delta E + \cdots = \frac{d}{dt}\left(\frac{\Delta E}{\omega_0}\right)$$

Where we have expanded $\frac{1}{\omega}\dot{E} = \frac{1}{\omega_0+\Delta\omega}\dot{E}$ using a Taylor series. This gives the following EoM for the energy-offset:

$$\frac{d}{dt}\left(\frac{\Delta E}{\omega_0}\right) = \frac{1}{2\pi} eV_{\text{rf}} \left(\sin\phi - \sin\phi_{\text{s}}\right) = \frac{1}{2\pi} e(V(\phi) - V(\phi_s)) \tag{2.19}$$

where now $V(\phi)$ is the accelerating voltage felt by the real particles. As for the time evolution of the phase angle, ϕ, there are more relations and concepts that need to be introduced. A key concept is the *off-momentum* variable, which indicates the ratio between momentum difference of the particle and the synchronous particle's momentum, $\delta = \frac{\Delta p}{p_0}$ (it can also be related to the energy difference $\delta = \frac{\omega_0}{\beta^2 E}\frac{\Delta E}{\omega_0}$). This variable defines the

so-called dispersion function $D(s)$, which relates how this off-momentum of the particles alters the path of the circular orbit of the particle. Thus, these relations are defined below:

$$\frac{\Delta\omega}{\omega_0} = \frac{\frac{2\pi R}{\beta c} - \frac{2\pi R_0}{\beta_0 c}}{\frac{2\pi R_0}{\beta_0 c}} = \frac{\beta R_0}{\beta_0 R} - 1 \tag{2.20}$$

$$R = R_0(1 + D(s)) = R_0\left(1 + \alpha_0\delta + \alpha_1\delta^2 + \alpha_2\delta^3 + \cdots\right) \tag{2.21}$$

where R_0 is the mean radius of a circular accelerator, R is the radius of the particle's circular orbit which is altered off-momentum of the particles (these perturbations are in the $\hat{x}$ direction in the Frenet–Serret coordinate system in Figure 2.3). Moreover, the α_n that define the dispersion function are called the *Courant–Snyder parameters* α_n, where n represents the number of times that the initial α_0 has been propagated by the transverse transfer matrix, but this will not be defined as it relates to transverse beam dynamics and escapes the scope of the thesis (see [19, 30] for more information). These α_n can be related to another accelerator-specific parameter called the momentum compaction factor α_c:

$$\alpha_c = \frac{1}{R_0}\frac{dR}{d\delta} = \alpha_0 + 2\alpha_1\delta + 3\alpha_2\delta^2 + \cdots \tag{2.22}$$

Intuitively, by looking at the definition of α_c, one can assert that if $\alpha_0 > 0$, which is the dominant factor in the formula (δ in the order of 10^{-2} or smaller [30]), it means that particles with higher momentum than the synchronous particle will travel a longer orbit path than that of the synchronous particle, and vice-versa for $\alpha_c < 0$. Using the definition of the off-momentum variable, it is possible to express β and γ in terms of δ (after some substitutions) [30]:

$$\delta = \frac{\Delta p}{p_0} = \frac{\beta\gamma}{\beta_0\gamma_0} - 1 \longrightarrow \begin{aligned} \frac{\gamma}{\gamma_0} &= \frac{\beta_0}{\beta}(1+\delta) = \frac{\frac{1}{\sqrt{1-\beta^2}}}{\frac{1}{\sqrt{1-\beta_0^2}}} = \sqrt{1 + 2\beta_0^2\delta + \beta_0^2\delta^2} \\ \frac{\beta}{\beta_0} &= \frac{1+\delta}{\sqrt{1 + 2\beta_0^2\delta + \beta_0^2\delta^2}} = 1 + \frac{1}{\gamma_0^2}\delta - \frac{3\beta_0^2}{2\gamma_0^2}\delta^2 + \frac{\beta_0^2\left(5\beta_0^2 - 1\right)}{2\gamma_0^2}\delta^3 + \cdots \end{aligned}$$

Replacing these expressions in Equation 2.20 and 2.22, integrating Equation 2.22 and replacing in Equation 2.20 ultimately allows expressing the relation in the following way:

$$\frac{\Delta\omega}{\omega_0} = -\eta(\delta)\delta \tag{2.23}$$

where η is the phase-slip factor and it can be expanded in the off-momentum variable as follows:

$$\eta(\delta) = \eta_0 + \eta_1\delta + \eta_2\delta^2 + \cdots \longrightarrow \begin{cases} \eta_0 = \left(\alpha_0 - \frac{1}{\gamma_0^2}\right), \\ \eta_1 = \frac{3\beta_0^2}{2\gamma_0^2} + \alpha_1 - \alpha_0\eta_0, \\ \eta_2 = -\frac{\beta_0^2\left(5\beta_0^2-1\right)}{2\gamma_0^2} + \alpha_2 - 2\alpha_0\alpha_1 + \frac{\alpha_1}{\gamma_0^2} + \alpha_0^2\eta_0 - \frac{3\beta_0^2\alpha_0}{2\gamma_0^2}. \end{cases}$$

Thus, in the first order approximation, only η_0 is considered and then the difference in angular frequency is:

$$\Delta\omega = -\eta_0\omega_0\delta = \left(\frac{1}{\gamma^2} - \alpha_c\right)\omega_0\delta \tag{2.24}$$

Looking at the equation, it is beneficial to relate α_c to a so-called transition Lorentz gamma γ_T to rewrite $\Delta\omega$ to :

$$\alpha_c = \frac{1}{\gamma_T} \longrightarrow \Delta\omega = \left(\frac{1}{\gamma^2} - \frac{1}{\gamma_T}\right)\omega_0\delta \tag{2.25}$$

Much like for the momentum compaction factor, this means that below the transition gamma, a higher energy particle with $\delta > 0$ will have a higher revolution frequency, and vice-versa with $\delta < 0$. Using this completes the EoM for both energy difference and phase (substituting δ):

$$\begin{aligned} \frac{d}{dt}\left(\frac{\Delta E}{\omega_0}\right) &= \frac{1}{2\pi} eV_{\mathrm{rf}} \left(\sin\phi - \sin\phi_{\mathrm{s}}\right) = \frac{1}{2\pi} e(V(\phi) - V(\phi_s)) \\ \frac{d\phi}{dt} &= \frac{h\omega_0^2\eta}{\beta^2 E}\left(\frac{\Delta E}{\omega_0}\right) \end{aligned} \tag{2.26}$$

This could have also been derived through the synchrotron Hamiltonian, or also in a discrete manner on a turn by turn basis, which allows for simple and sequential tracking of particles (see [50] for full discrete derivation). This will be discussed in more detail in Section 2.2.3 because this is how particles are simulated computationally. Ultimately, if this approach were followed, the following tracking equations would be obtained:

$$\begin{aligned} \Delta E_{n+1} &= \Delta E_n + eV_{\mathrm{rf}} \left(\sin\phi_n - \sin\phi_{\mathrm{s}}\right) \\ \phi_{n+1} &= \phi_n + \frac{2\pi h\eta}{\beta^2 E}\Delta E_{n+1}. \end{aligned} \tag{2.27}$$

where again n is referring to the turn number. Note that this is obtained by simply discretizing the dt to be the revolution period of the synchronous particle $dt = T_0$, which cancels out with the angular frequency term on the r.h.s. of Equation 2.19.

These equations can be linearized in order to familiarize oneself with the type of motion that the particles will undergo. To do so, consider again the non-discretized EoMs in Equation 2.26, and first linearize the ΔE EoM for small variations in phase around the synchronous phase ϕ_s using a Taylor series and keeping only the linear term:

$$\frac{d}{dt}\left(\frac{\Delta E}{\omega_0}\right) \approx \frac{1}{2\pi} eV_{\mathrm{rf}} \cos\phi_s \left(\phi - \phi_s\right) \tag{2.28}$$

Then take the time derivative of the ϕ EoM to replace the linearized ΔE EoM within it to retrieve the linearized EoM for the phase:

$$\frac{d^2}{dt^2}\left(\phi - \phi_{\mathrm{s}}\right) = \frac{h\omega_0^2 eV\eta_0 \cos\phi_{\mathrm{s}}}{2\pi\beta^2 E}\left(\phi - \phi_{\mathrm{s}}\right) \tag{2.29}$$

This equation is depicting a harmonic oscillator, with a frequency $\omega_{\mathrm{s0}} = \omega_0\sqrt{\frac{heV\eta_0\cos\phi_{\mathrm{s}}}{2\pi\beta^2 E}}$, which is known as the small-amplitude *synchrotron frequency* of the linearised system. Additionally, if one were to divide it by the revolution frequency ω_0, then one retrieves the *synchrotron tune* Q_s, whose inverse represents the number of turns needed to complete one full revolution in phase space:

$$Q_{\mathrm{s0}} = \sqrt{\frac{heV\eta_0\cos\phi_{\mathrm{s}}}{2\pi\beta^2 E}} \tag{2.30}$$

Looking at Equation 2.27 and 2.29, one can appreciate the physical meaning of the phase-slip factor η (or its first order approximation η_0), as when it is positive, then a particle with a positive ΔE will increase in phase, meaning that the particles will oscillate in the clockwise direction in phase-space. Furthermore, it is well known that these type of differential equations are solved by a linear combination of exponentials $\exp \pm w_{s0} t$ with real or imaginary arguments depending on whether the frequency is imaginary or real, respectively. Thus, considering the system of particles as harmonic oscillators, the condition for the phase coordinate to have a bounded evolution is that ${\omega_{s0}}^2 < 0$ in order to have an imaginary frequency, which means that $\eta \cos \phi_s < 0$. This essentially restricts the value of ϕ_s depending on whether the accelerator operates above or below the transition gamma γ_T (see Equation 2.25). Below transition for $\gamma < \gamma_T$ (meaning that $\eta < 0$ from Equation 2.25), then for stability (i.e. for the motion to remain bounded in phase-space) one must have $0 < \phi_s < \frac{\pi}{2}$ and above transition $\frac{\pi}{2} < \phi_s < \pi$. If the motion becomes unbounded, and the particles deviate sufficiently to become lost as the magnets are not synchronized with their momentum through Equation 2.7, and their trajectories are bent either too much or too little and the particles are lost to the beam pipe. Also, it is important to keep in mind that **the PSB operates completely below transition so** $\eta < 0$. Meanwhile, other accelerators, like the PS, cross this transition boundary: initially $\gamma < \gamma_T$ and from a certain point in the cycle $\gamma \geq \gamma_T$ implying the voltage phasing in the cavities must change as well in order to keep the beam stable [15].

Nevertheless, this linear analysis can only provide information on the low amplitude oscillations of the particles in phase-space about ϕ_s. To understand the how particles evolve within a larger region in phase-space, one must consider the synchrotron Hamiltonian:

$$H = \frac{1}{2} \frac{h \eta \omega_0^2}{\beta^2 E} \left(\frac{\Delta E}{\omega_0} \right)^2 + \frac{e V_{\mathrm{rf}}}{2\pi} \left[\cos \phi - \cos \phi_{\mathrm{s}} + (\phi - \phi_{\mathrm{s}}) \sin \phi_{\mathrm{s}} \right] = K + U(\phi) \tag{2.31}$$

where we have now replaced $\phi_s = \phi_{s0}$ which indicates the synchronous phase but for a single-harmonic system. Additionally, K and $U(\phi)$ are the kinetic and potential energies of the particle, where the potential energy (also called RF Potential) can be computed by integrating the r.h.s. of the energy difference EoM in Equation 2.19 with respect to ϕ and ϕ_s, as follows (in general for any voltage $V(\phi)$) [19]:

$$U(\phi) = \frac{e}{2\pi} \int_{\phi}^{\phi_s} [V(\phi') - V(\phi_s)] d\phi' \tag{2.32}$$

The Hamiltonian in Equation 2.31 does not describe the full picture given that *transverse* dynamics also couples with the *longitudinal* dynamics, but as mentioned before, it is still sufficient to consider the decoupled system [30]. Regardless, returning to the topic of stability, the potential well formed by $U(\phi)$ determines the region of bounded orbits in phase-space (if the potential well is unperturbed), as is usual in Hamiltonian systems. For example, in the single-harmonic case, the potential is shown in the middle plot of Figure 2.4 for multiple values of ϕ_s, meaning at different acceleration magnitudes. The voltage waveform that generates this potential well is shown on the left plot of Figure 2.4. It is possible to distinguish a minimum and 2 maxima that define the potential well. Actually, in the energy difference EoM in Equation 2.19, for given initial points, neither the phase nor the energy difference will change. These points are called *fixed points* and they can be unstable or stable, depending on the whether the RF potential energy has a maximum or minimum at that particular phase, respectively. For the single-harmonic system, the stable fixed point is located at $(\Delta E = 0, \phi = \phi_s)$, while the unstable fixed point is located at $(\Delta E = 0, \phi = \pi - \phi_s)$.

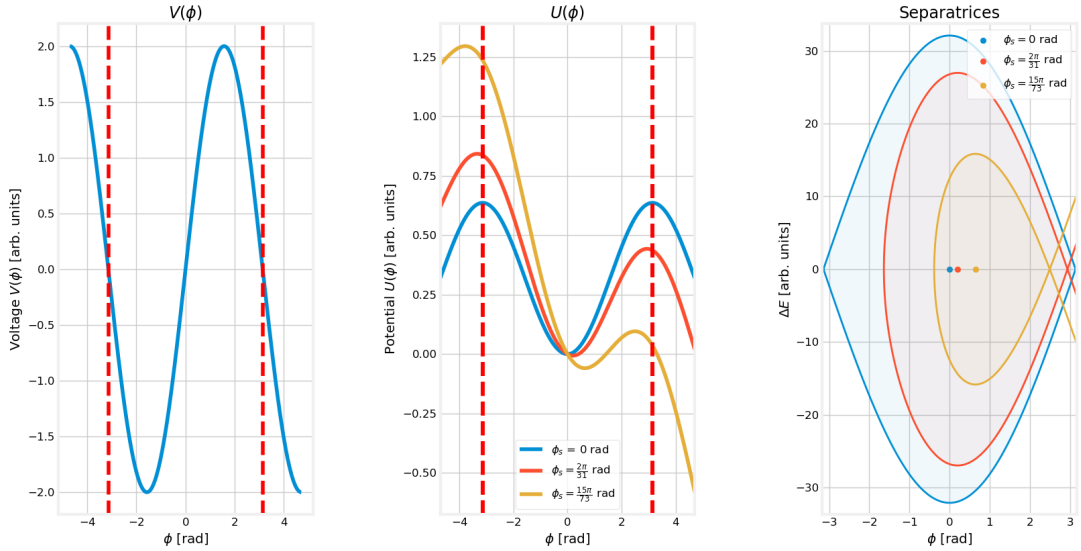


Figure 2.4: Accelerating voltage, potential energy, and separatrix for single-harmonic operation under different ϕ_s (dashed red lines indicate RF period)

In the potential energy in Figure 2.4, the higher the acceleration, the shorter the potential well becomes, and the allowed energy difference and phase divergence from ϕ_s before the particles escape the potential well is smaller as well (this is defined by the smallest maxima present within one period). There is another way to represent this potential well in phase-space with the coordinates $(\Delta E, \phi)$, which gives an even better picture of the stability of particles in the same coordinate system in which the tracking occurs. It is achieved by obtaining the contour in phase-space of the Hamiltonian value at the unstable fixed point $(\Delta E = 0, \phi = \pi - \phi_s)$, meaning the **maxima with the lowest potential energy**. This representation is called a *separatrix* and are shown for the same synchronous phases in the right plot Figure 2.4. This closed contour, also known as *acceleration bucket*, is representing the area in which particles' motion will be bounded (if no perturbations are present) as they oscillate around the synchronous particle in phase-space, located at $(\Delta E = 0, \phi = \phi_s)$. In addition, if the Hamiltonian varies slowly with time, meaning that the system can be considered quasi-adiabatic, then the particles will follow lines of constant Hamiltonian value while they oscillate around ϕ_s, albeit with different frequencies.

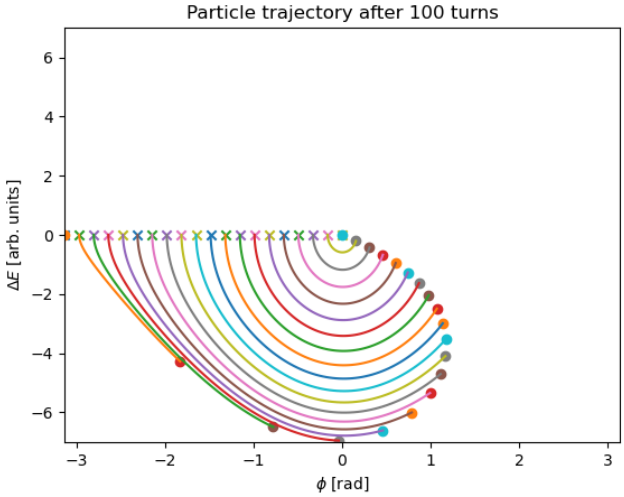


Figure 2.5: The motion of 100 particles with varying initial phases ϕ but with $\Delta E = 0$ in a single-harmonic system

The analysis in Equation 2.29 appears to indicate that the synchrotron frequency of the single-harmonic system, ω_s, is independent of the particle's Hamiltonian value, but this is not the case. This effect can be shown by tracking particles for 100 turns using Equation 2.27 with an initial position along the ϕ-axis distributed with

$\phi = [-\pi, 0]$ as shown in Figure 2.5. As can be seen, particles with a higher absolute deviation from the synchronous phase, $\phi_{max} = \phi - \phi_s$, will have a smaller synchrotron frequency ω_s, whose dependency on ϕ_{max} can be calculated analytically for a single harmonic system [19, 30]:

$$\omega_s\left(\phi_{\max}\right) = \omega_{s0}\frac{\pi}{2K\left[\sin\phi_{\max}/2\right]} \approx \omega_{s0}\left(1 - \frac{\phi_{\max}^2}{16}\right) \tag{2.33}$$

where $K(\cdot)$ is the complete elliptic integral of the first kind, and ω_{s0} is the synchrotron frequency of the linearized system. This concept of oscillating motion with a certain frequency ω_s is important when considering perturbations, such as non-RF-driven voltages, that are modulated by some frequency. This is because of possible resonance between the perturbation and the system of particles oscillating at a certain synchrotron frequency. This resonance would cause oscillations in phase space to have increasing amplitude (and thus higher Hamiltonian values in Equation 2.31) and ultimately lead to particles escaping the potential well (or separatrix) and become lost [19]. As a result, it is beneficial to have a large spread in ω_s as it would imply that a perturbation driven at a particular frequency will not affect one's beam as much as if the synchrotron frequency is less spread out within the separatrix. This effect is known as coherent excitation.

Furthermore, a larger frequency spread enhances a stabilization mechanism known as *Landau damping*, which it becomes important to mitigate the onset of so-called *longitudinal microwave instabilities* in the PSB near extraction. These microwave instabilities are formed by the self-driven excitations inside of the beam relating to the longitudinal eigenmodes of the bunch of particles [24]. In addition, they become an issue at extraction as this is where the beam has the highest energy E and β, meaning that small-amplitude synchrotron frequency ω_{s0} and the tune Q_{s0} will decrease, and Equation 2.33 will be overall more flattened, reducing the synchrotron frequency spread [24].

An example of them is shown in the *waterfall plot* in Figure 2.6, where the waterfall plot is formed by longitudinal bunch profiles stacked on top of each other over time. This instability (which occurs at transition crossing in the PS: when the sign of η changes) causes transient oscillations in the longitudinal bunch profile, with micro-bunches being formed which could lead to beam loss [39]. The understanding of why this happens extends beyond just considering frequency spread and also looking at how the particles are distributed in phase space: the higher the energy spread of the particles and the lower the bunch length, the more resilience the beam has against microwave instabilities [30]. More analysis on this will be given in Section 2.2.3 (but if the reader wants more information on the topic, see the Vlasov equation in [30]).

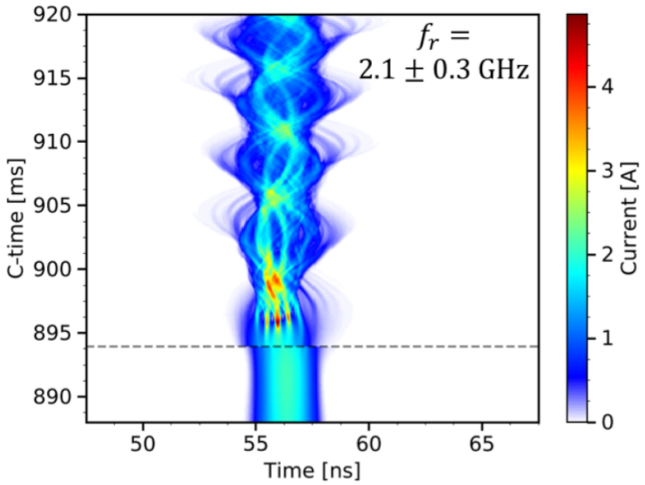


Figure 2.6: Microwave instabilities at transition crossing in the PS shown through a waterfall plot [39]

Multi-Harmonic System

The extrapolation to multiple RF systems starts at Equation 2.26 because here is where the accelerating voltage $V(\phi)$ is present, which is the RF signal inputted into the cavities. This voltage is now not only made

up of one harmonic, but incorporates multiple harmonics of the revolution frequency $h\omega_0$:

$$V(\phi) = \sum_{h=1}^{h_n} V_{\mathrm{rf}}^{(h)} \sin(h\phi + \Phi_{(h)}) \tag{2.34}$$

Here $\Phi^{(h)}$ represents the phasing of each harmonic with respect to the reference phase, which in this representation is $\Phi_{ref} = 0$. From here on out, the subscript in V_{rf} is dropped for convenience and sources of voltage external to the RF system will be defined when appropriate. Now plugging this new accelerating voltage in Equation 2.15 change in energy of the synchronous particle is now:

$$(\Delta E_s)_{\mathrm{gain}} = q \sum_{h=1}^{h_n} V_{(h)} \sin(h\phi_s + \Phi_{(h)}), \tag{2.35}$$

where ϕ_s is the synchronous phase angle that is now modified by the presence of the multiple-harmonic system, and h_n is the highest harmonic used in the cavity (note that $h_n = h_2$ in the double-harmonic operation in the PSB). Additionally, the multi-harmonic synchronous phase ϕ_s can be related to the single-harmonic one ϕ_{s0} such that they both provide the same acceleration rate for the synchronous particle (meaning the same programmed acceleration rate, regardless of the presence of higher harmonics). For this purpose and for future convenience, we present the voltage ratio $r_h = \frac{V_h}{V_1}$:

$$\sin(\phi_{s0} + \Phi_1) = \sum_{h=1}^{h_n} r_h \sin(h\phi_s + \Phi_{(h)}), \tag{2.36}$$

By following the same derivation as earlier, but now replacing $V(\phi)$ in Equation 2.26 with Equation 2.34, the discretized tracking equations are now modified:

$$\begin{aligned} \Delta E_{n+1} &= \Delta E_n + e(V(\phi_n) - V(\phi_s)) = \Delta E_n + e\sum_{h=1}^{h_n} V_{(h)}(\sin(h\phi_n + \Phi_{(h)}) - \sin(h\phi_s + \Phi_{(h)}) \\ \phi_{n+1} &= \phi_n + \frac{2\pi h_1 \eta}{\beta^2 E} \Delta E_{n+1}. \end{aligned} \tag{2.37}$$

Now the optimisation variable for the double-harmonic system becomes clear: Φ_2, while the other parameter which has a substantial effect on the evolution of particles in phase space is the voltage ratio r_2. The importance of both parameters will be made clearer when analysing the potential of the double-harmonic system and the corresponding separatrix. To do so, we again integrate as indicated by Equation 2.32 to get the general potential for multi-harmonic systems:

$$U(\phi) = \frac{e}{2\pi}[\sum_{h=1}^{h_n} \frac{V_{\mathrm{rf}}^{(h)}}{h}(\cos(h\phi + \Phi_{(h)}) - \cos(h\phi_s + \Phi_{(h)})) + \sum_{h=1}^{h_n} V_{\mathrm{rf}}^{(h)} \sin(h\phi_s + \Phi_{(h)})(\phi - \phi_s)]$$

where now $\Phi^{(n)}$ is the relative phase between the fundamental (1st) and nth harmonic of the voltage (i.e. $\Phi_1 = 0$ because $\Phi_{ref} = \Phi_1$). This is defined as such because in the PSB, the voltage harmonic's phases are defined relative to that of the fundamental harmonic.

Double-Harmonic Operation The potential for the double-harmonic $U(\phi)$ is shown in the middle plots of Figure 2.7 and Figure 2.8 for $r_2 = 0.3$ and $r_2 = 0.6$, respectively, where the effect of the second harmonic can be appreciated in the shape of the potential well by comparing with Figure 2.4. The higher the voltage ratio r_2, then the flatter the distribution becomes (up to a certain extent). Additionally, there exists a threshold for the voltage ratio, specifically $r \geq 0.5$ (but this varies with the energy gained per turn ΔE_s), after which an additional minimum and maximum form within a single h_1 period, which is slightly visible in the RF potential

for $r = 0.6$ in Figure 2.8 in comparison to the single minimum found in the potential energy of Figure 2.7. The maximum is now located essentially in the centre of the potential well for the non-accelerating bucket ($\phi_s = 0$), shown in blue.

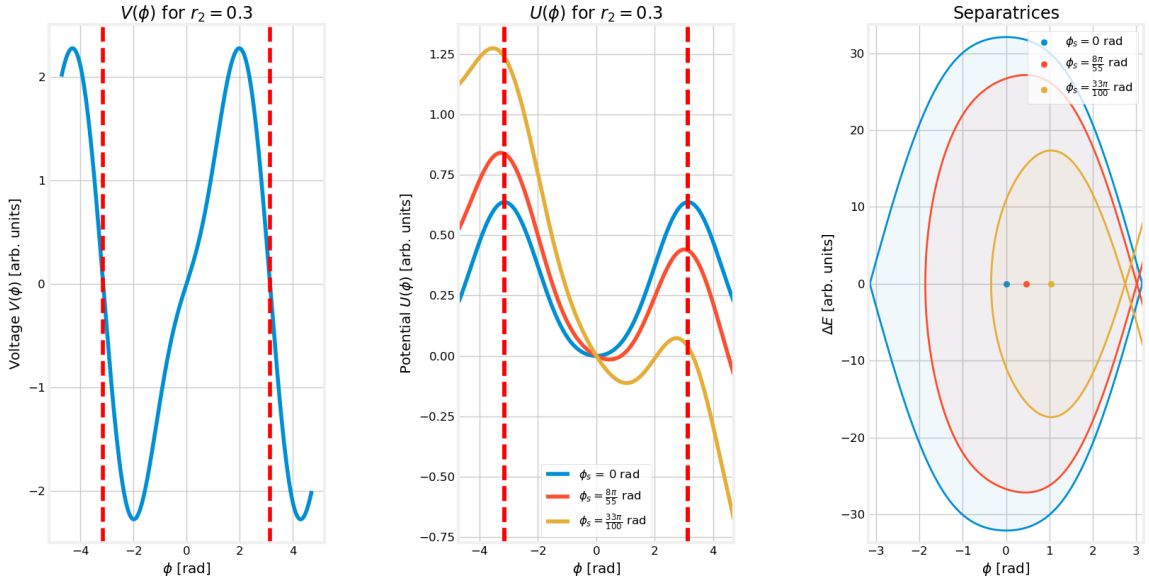


Figure 2.7: Accelerating voltage, potential energy, and separatrices for double-harmonic operation under different ϕ_s for $r = 0.3$ and $\Phi_2 = \pi$ (red lines indicate h_1 periods)

This additional minimum and maximum essentially mean that there is another stable and unstable fixed point within an h_1 period. Thus, by using the same method as in the single-harmonic case, given the presence of three unstable fixed points and 2 stable fixed points, there will be **two separatrices**. This can be appreciated in the blue separatrices of Figure 2.8 for the $r = 0.6$ case, where the dashed line indicates this inner separatrix. Now a particle on this iso-Hamiltonian contour, will follow this closed orbit forming a figure-eight in the absence of external perturbations due to the addition of these fixed points.

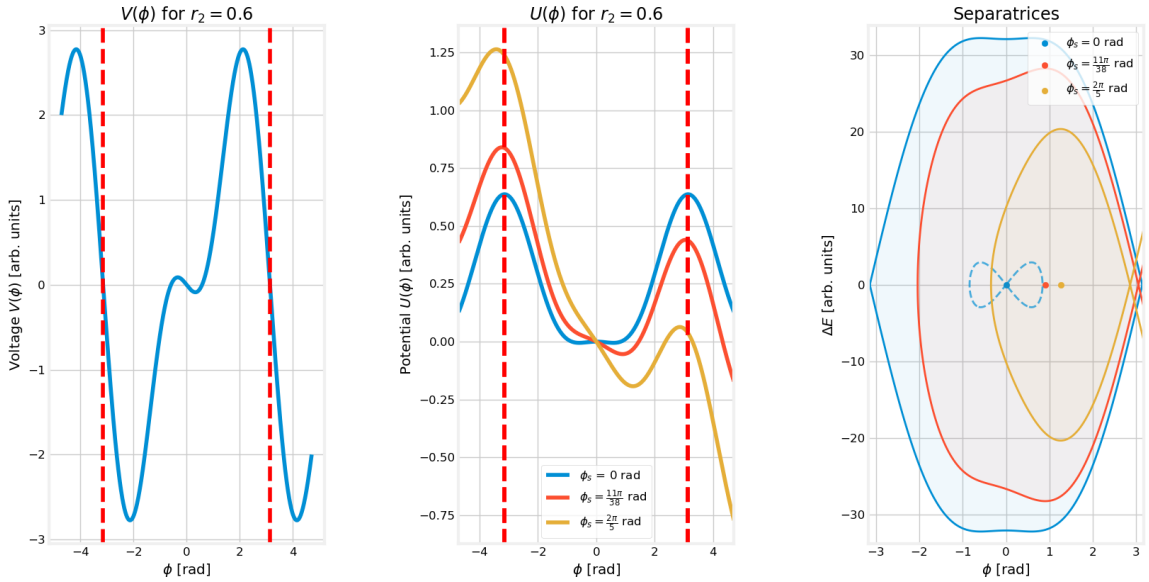


Figure 2.8: Accelerating voltage, potential energy, and separatrices for double-harmonic operation under different ϕ_s for $r_2 = 0.6$ and $\Phi_2 = \pi$ (red lines indicate h_1 periods). The dashed blue line in the right plot (for $\phi_s = 0$) indicates the formation of an inner separatrix.

Moreover, if the voltage ratio is increased even further to $r = 0.9$, then the outer separatrix is even more perturbed such that now it also has more pronounced lobes rather than being flat as shown in the right plot of Figure 2.9. Additionally, it can be appreciated in Figure 2.7, 2.8, and 2.9 that in the presence of acceleration, meaning $\Delta E_s > 0$ and $\phi_s \neq 0$ there is a perturbation in the depth of the RF potential for one of the lobes,

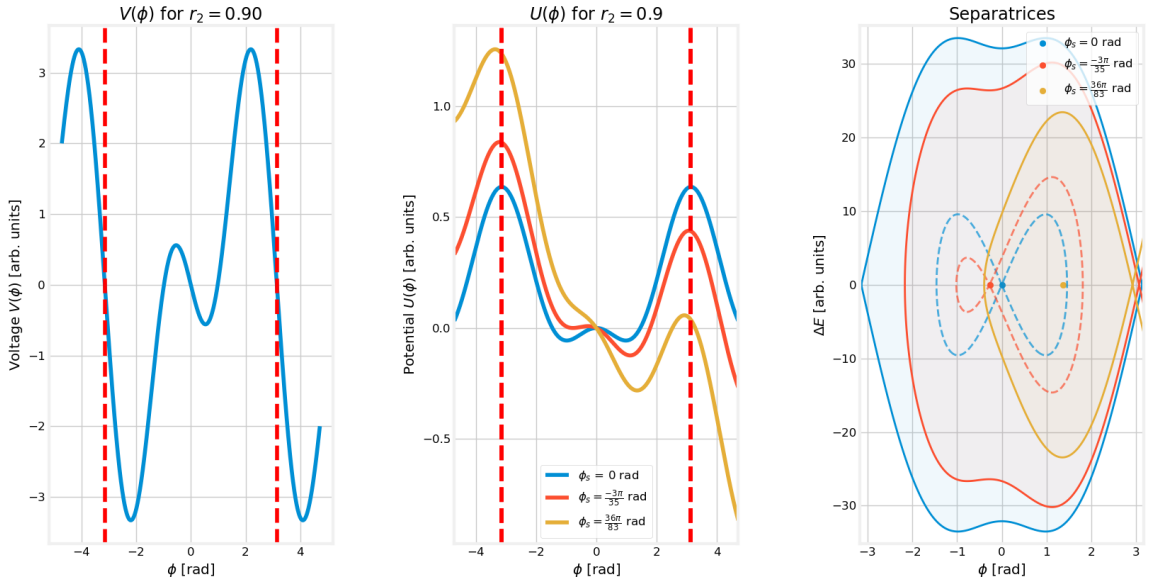


Figure 2.9: Accelerating voltage, potential energy, and separatrices for double-harmonic operation under different ϕ_s for $r_2 = 0.9$ and $\Phi_2 = \pi$ (red lines indicate h_1 periods). The dashed separatrices indicate the formation of an inner separatrix.

which essentially means that one of the lobes in the inner separatrix is going to be larger than the other. This is a characteristic of an asymmetric potential well.

In order to compensate for this, one must also change the phasing of the second harmonic (Φ_2) to shift the synchronous phase ϕ_s according to Equation 2.36 and make the minima have the same potential energy. This is done such that the lobes are symmetric and have the same depth, as the depth of the well is also proportional to how many particles will fall into this well and be at that position in the longitudinal axis and will determine the shape of the longitudinal charge density λ_t. This will be shown graphically in Section 2.4.

It is also imperative to define the two most common operational modes for double-harmonic operation which are defined by Φ_2 in Equation 2.34: Bunch Shortening Mode (BSM) and Bunch Lengthening Mode (BLM). Thus, by choosing the relative phase of the second harmonic ($\Phi_{ref} = \Phi_1$) to be $\Phi_2 = 0$ or π, the operation mode becomes BLM or BSM, respectively. These modes are depicted in phase-space in Figure 2.10 where the closed, constant Hamiltonian contours are shown as well.

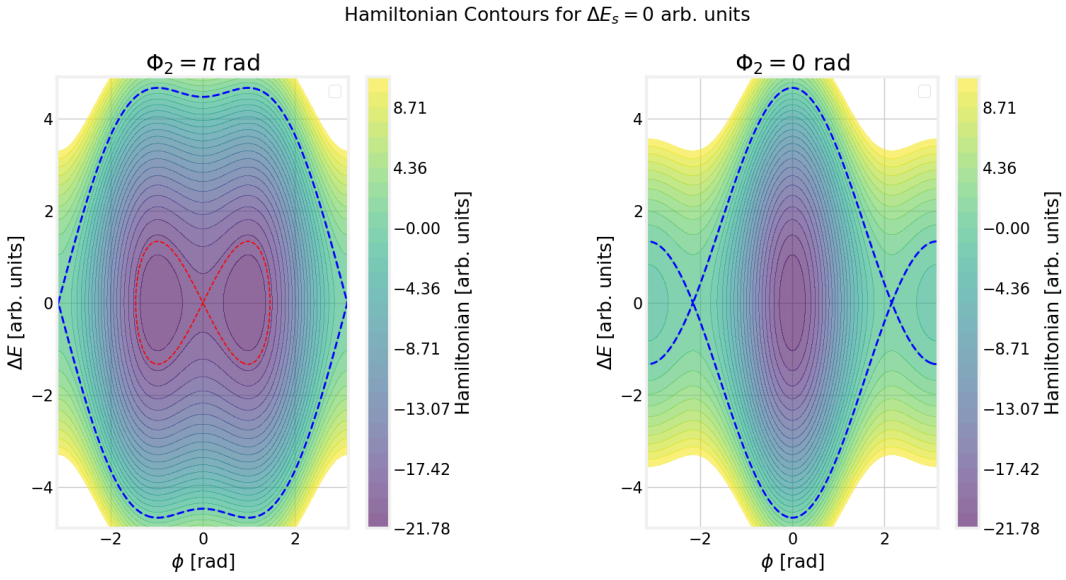


Figure 2.10: Separatrices for Bunch Lengthening Mode (left) Bunch Shortening Mode (right) with closed, constant Hamiltonian contours for $r_2 = 0.9$

Thus, as can be seen in Figure 2.10, particle distributions inside a separatrix in BSM will be compressed in the time (or ϕ) domain, while those in BLM will be elongated. Since what we are obtaining in the longitudinal bunch profile is the amount of particles in discrete bins in time, the bunch profile λ_t will also be more peaked, which we know will induce higher space-charge. Depending on the characteristics of the RF manipulation that operators want to perform, entering either of these modes allows shaping the bunch.

Furthermore, this can be extrapolated to the more general scenario where the beam is accelerating, as shown in the left plot of Figure 2.11. Again, without the correct phasing of Φ_2, the potential well and separatrix are asymmetric, leading to an asymmetric charge density in the longitudinal axis and larger space-charge effects (as will be discussed in Section 2.3). By properly phasing the second harmonic, the separatrix in the right plot of Figure 2.11 can be achieved which represents the symmetric potential well. This implies that, under *matched bunch* conditions, the same amount of particles will sit in each minima of the potential well. By definition, a matched bunch is a bunch distribution which has the same density of particles along every constant Hamiltonian closed orbit that its particles are on (explained graphically in Section 2.4).

In addition, in reality, the RF system operates in a non-ideal environment with effects, besides that of acceleration, that shift the synchronous phase ϕ_s in a multi-harmonic system. These effects can include path delays in the wires that send signals to the cavities, noise in the signals, and other physical phenomena that change the required Φ_2 to make the potential wells symmetric. This again causes a non-symmetric potential well which inevitably leads to a non-symmetric longitudinal bunch profile, which causes issues with stability in relation to impedance effects (see Section 2.3).

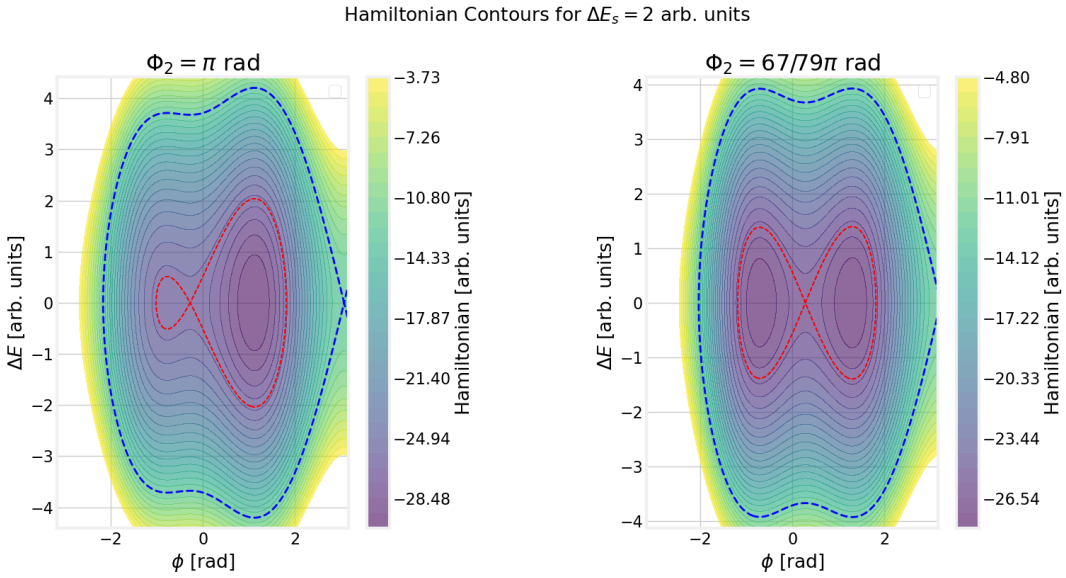


Figure 2.11: Uncorrected and corrected accelerating beam separatrix

Triple-Harmonic Operation To gain insight into how triple-harmonic operation potential wells and separatrices form, the plots in Figure 2.12 were produced. In the left plot of Figure 2.12, the voltage waveform now has a larger extent of ϕ values near 0 voltage due to the formation of an additional maxima and minima because of the presence of h_3. Through the definition in Equation 2.38, this means that there will be less change in the potential well during these phases, and it is represented in the middle plot where the potential is now almost flat for the range $[-2, 2]$ rad (compare with Figure 2.8 where the flat part is only achieved in the range $[-1, 1]$ rad). Furthermore, the appearance of an additional maxima in the potential well, but both maxima with the same potential energy, implies that the inner separatrix now has a more complex shape with 3 loops, rather than 2, with now two synchronous phases ϕ_{s1} and ϕ_{s2} because of the two inner maxima in the potential well. Additionally, in the accelerating case (red) there are two inner separatrices, as the maxima in the potential well have different heights now, causing a slightly more complex motion in phase space.

Note that in Figure 2.12, the phasing for the third harmonic is now $\Phi_3 = 0$ rad, as in general (for a non-accelerating bucket) the phases of additional harmonics must be $\Phi_n = (n-1)\pi$ in order to be in BLM. This is exemplified in the contour plots in Figure 2.13 where the cases of three different phasings are shown for the exact same voltage ratio configuration ($r_2 = 1.3, r_3 = 0.9$) as this was seen to produce the flattest potential well. Additionally, when phasing the system such that the bunch is shortened as much as possible, one is creating contours which cover less extent in the ϕ axis in comparison to BSM in double-harmonic operation, demonstrating the increased bunch shaping capabilities when adding an additional harmonic.

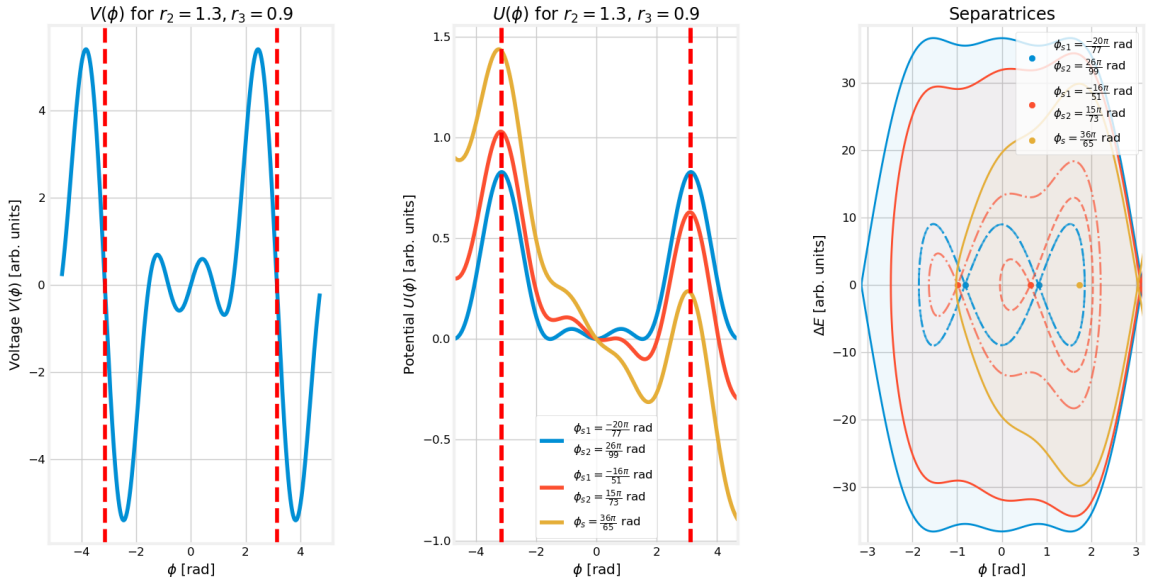


Figure 2.12: Accelerating voltage, potential energy, and separatrices for double-harmonic operation under different acceleration rates ΔE_s for $r_2 = 1.3, r_3 = 0.9$ and $\Phi_2 = \pi, \Phi_3 = 0$ rad (red lines indicate h_1 periods). Different inner separatrices are represented with different dashed line patterns.

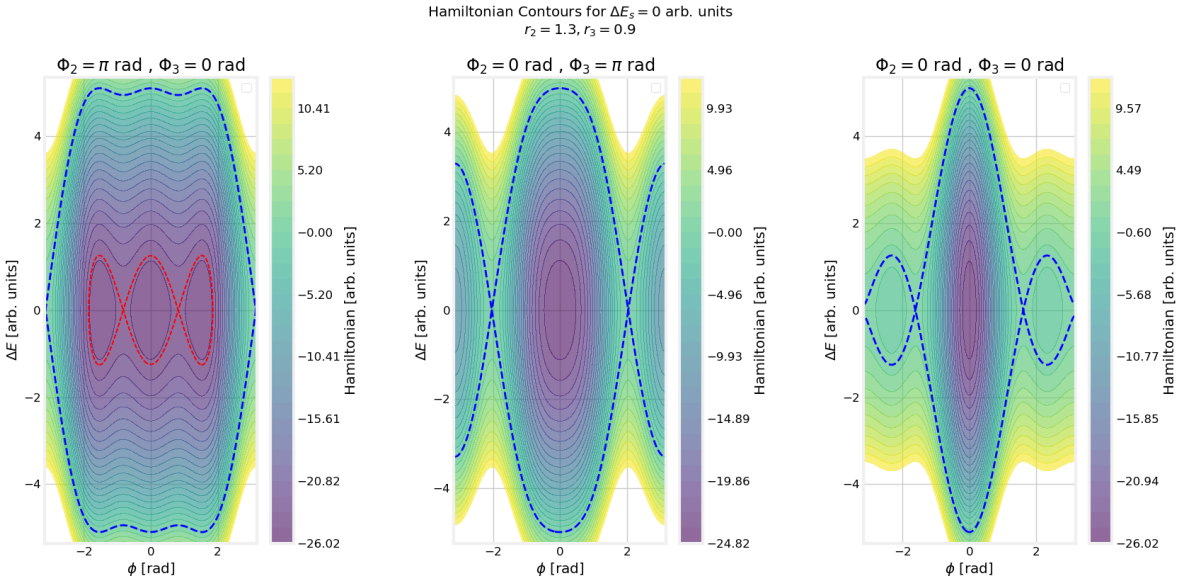


Figure 2.13: Separatrices for the case of no acceleration for different triple-harmonic configurations of with closed, constant Hamiltonian contours

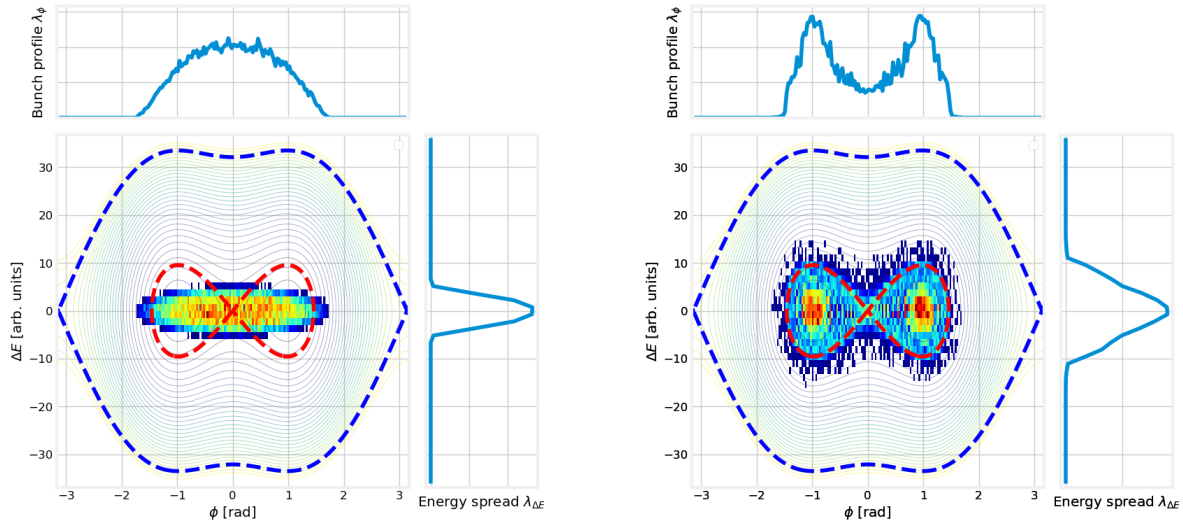
Now that all the theory, tools, operational modes and coordinate spaces for tracking in multi-harmonic operation have been defined, it is important to now look at simulation results **without impedance effects** to understand the motion and the optimization problem at hand better. Thus, a non-linear beam dynamics tracking tool was developed for the purpose of demonstrating the motion of particles in phase-space [47].

2.2.3. Idealized Tracking

Equation 2.37 was used for the turn-by-turn tracking, while the separatrix was computed through the use of the multi-harmonic extension of Equation 2.31 using Equation 2.38. For this simplified tracking discussion, only the non-accelerating case of $\Delta E_s = 0$ is considered for the evolution of the particles. Extending this to the accelerating case does not provide any more insight into phase-space tracking, besides the fact that the separatrix and closed orbits will be deformed as shown in Figure 2.11 and likewise for triple-harmonic operation. Again, this is compensated by proper phasing of Φ_2 and Φ_3. The code used for this tracking simulation can be found here [47]. It is encouraged to use it to gain intuition on how particles move in phase-space under multi-harmonic operation.

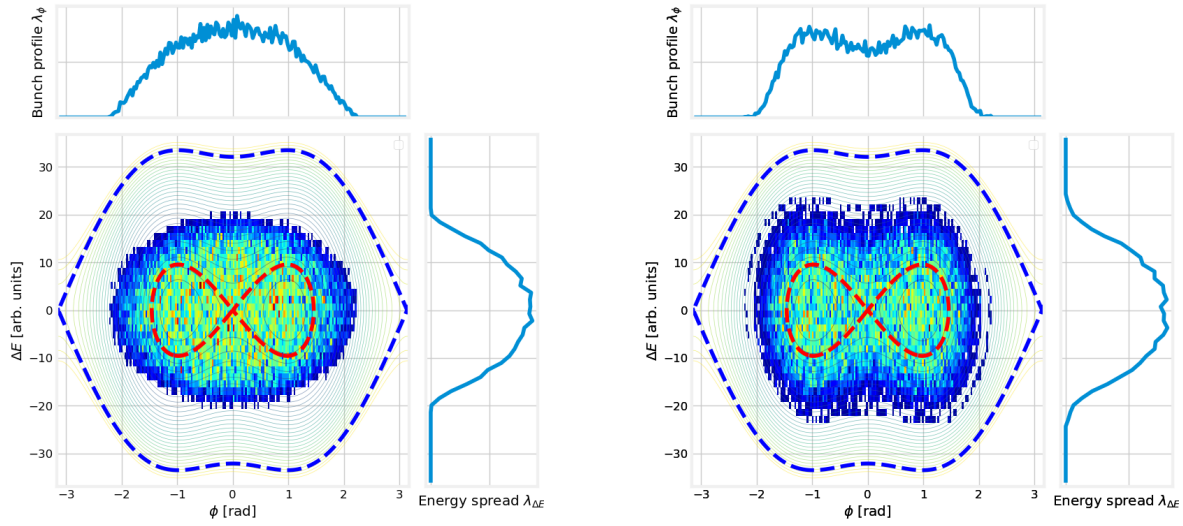
Double-Harmonic Tracking

For this discussion, three different initial distributions are employed to show the effect of the second harmonic on synchrotron motion. They are characterised based on a parameter called the *emittance* ϵ, which is simply the area occupied by the bunch in phase space (usually in units eV·s when transforming from phase ϕ to time t with the frequency of the fundamental harmonic ω_{h_1}). Thus, distributions that are tracked are the low-emittance distribution (Figure 2.14), high-emittance distribution (Figure 2.15), and the shifted distribution (Figure 2.16). The latter is simply a distribution which is not centred around the synchronous particle coordinate ($\Delta E = 0, \phi = 0$), but instead ($\Delta E = 0, \phi = 1.5\,\text{rad}$). The simulation is run for only 1000 turns, as simulating any more would not yield additional insight into the dynamics of particles. Note that the distribution is projected on the ϕ-axis and the histogram is plotted above showing the bunch profile (line density), λ_ϕ, while the distribution projected on the ΔE-axis shows the *energy spread*. Optionally, this can be converted from phase to time through the transformation $t = \frac{\phi}{\omega_{rf}}$ to retrieve the bunch profile in time, which is a direct observable in the accelerator.



(a) Initial distribution **(b)** Final distribution

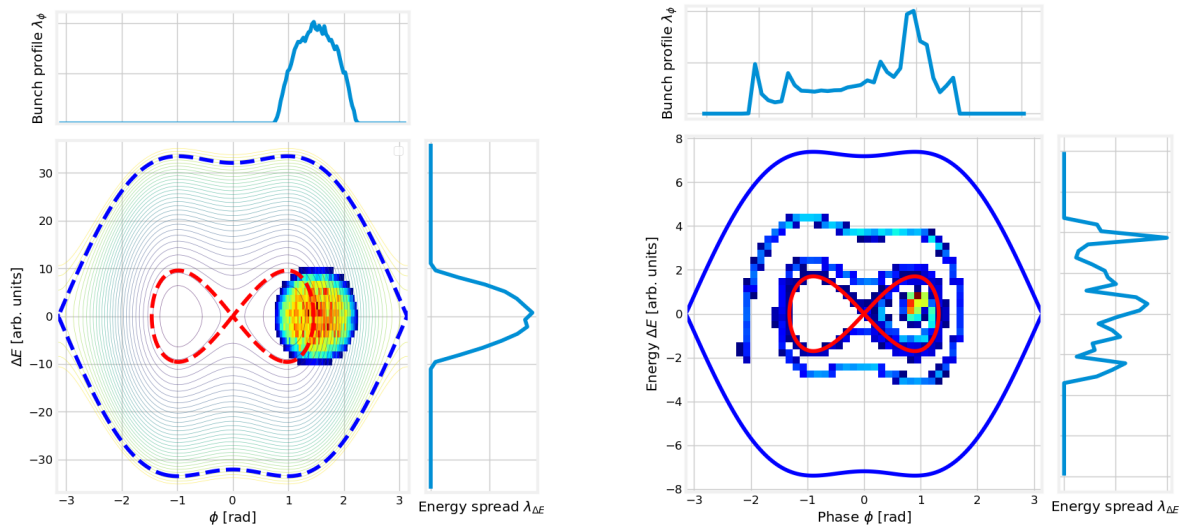
Figure 2.14: Low-emittance particle distribution evolution within separatrix for $\phi_s = 0$ rad for $r_2 = 0.9$



(a) Initial distribution **(b)** Final distribution

Figure 2.15: High-emittance ϵ_s, particle distribution evolution within separatrix for $\phi_s = 0$ rad for $r_2 = 0.9$

As can be seen in the final plots in Figure 2.14b and 2.15b, in both the small and large emittance cases, the particles stabilize around the inner separatrix and they end up with roughly the same symmetric shape despite having different initial conditions, albeit with different final emittances which is important for the automatised tool to be developed (see Section 2.4). However, when the initial particle distribution is shifted, then this effect is lost as witnessed in Figure 2.16, where some particles have been trapped inside one of the loops of the inner separatrix, which inevitably leads to a skewed bunch profile. This can be corrected in practice by performing



(a) Initial distribution **(b)** Final distribution

Figure 2.16: Shifted, particle distribution evolution within separatrix for $\phi_s = 0$ rad for $r_2 = 0.9$

RF manipulations to shape the charge distribution by modifying the orbit of the particles. For example, by driving $V_2 \to 0$, shifting the bucket quickly by changing Φ_1 to move the particles to new contours, then letting the distribution settle into its new Hamiltonian contours, and finally returning V_2 back to its original value. This is not needed in practice as particle distributions injected into an accelerator are matched, such that this shift never occurs.

Moreover, note that due to the fact that all initial particle distributions are unmatched, a process called *filamentation* occurs in all phase-space plots, where the bunch filaments due to the dependence of synchrotron frequency with the Hamiltonian value of the particle. This is clearly visible in Figure 2.16 because of the shift and in Figure 2.17 where the separatrix has not been plotted, but all particles are in the region where orbits are bounded and have been tracked for 100 turns. The particles that are oscillating with different frequencies with a non-trivial relationship between the synchrotron frequency and the deviation of the phase from the centre of the bucket at $(\Delta E = 0, \phi = 0)$, where ϕ_{max} is the deviation from this point. At the centre of the bucket, the oscillation stops as this is an unstable fixed point (i.e. a maxima in the RF potential energy). This ratio of the synchrotron frequency relative to the small-amplitude (single RF) synchrotron frequency, ω_s/ω_{s0}, and ϕ_{max} has been computed numerically in Figure 2.18 for a double-harmonic system [24, 15]. The ratio shown for multiple harmonic ratios ($r_h = h_2/h_1$ in this case) is the ratio between the small amplitude single-harmonic synchrotron frequency, ω_{s0}, and the synchrotron frequency. It can be seen that ω_s/ω_{s0} now appears to be amplitude-modulated due to the presence of the second harmonic, and the curve crosses the single RF curve $r_h - 1$ times [24].

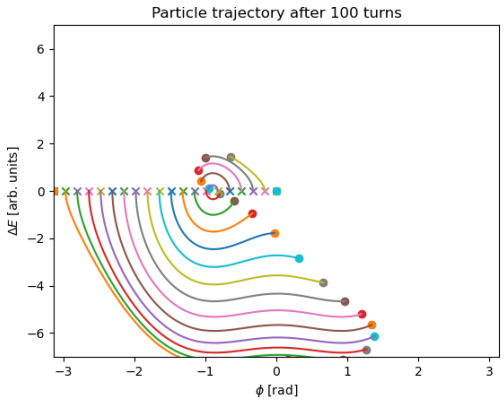


Figure 2.17: The motion of 100 particles with varying initial phases *phi* but with $\Delta E = 0$ in a double-harmonic system (BLM mode)

Even more important to the operational beams is the synchrotron frequency spread mentioned previously that can be obtained in either BSM or BLM. In BSM we see that a higher synchrotron frequency spread is obtained

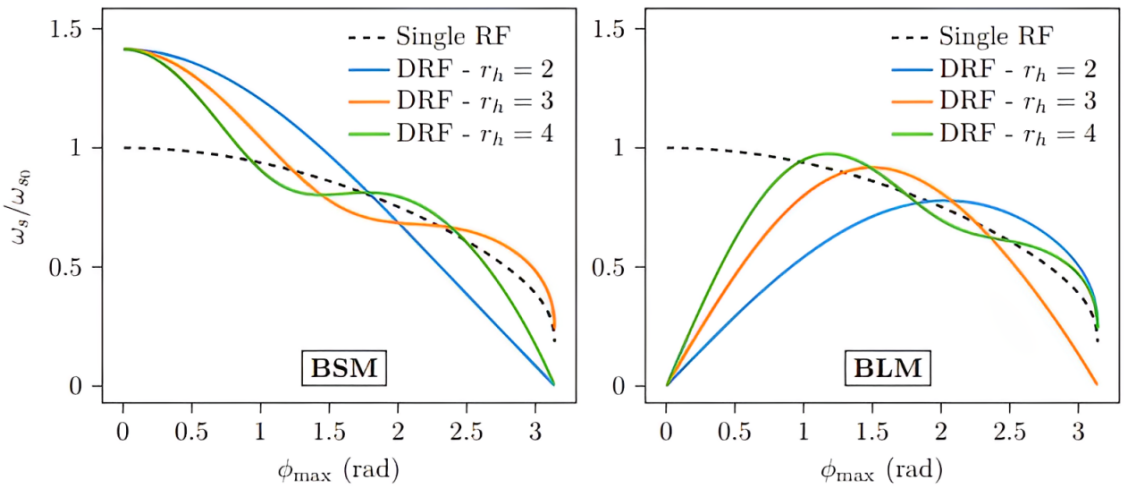
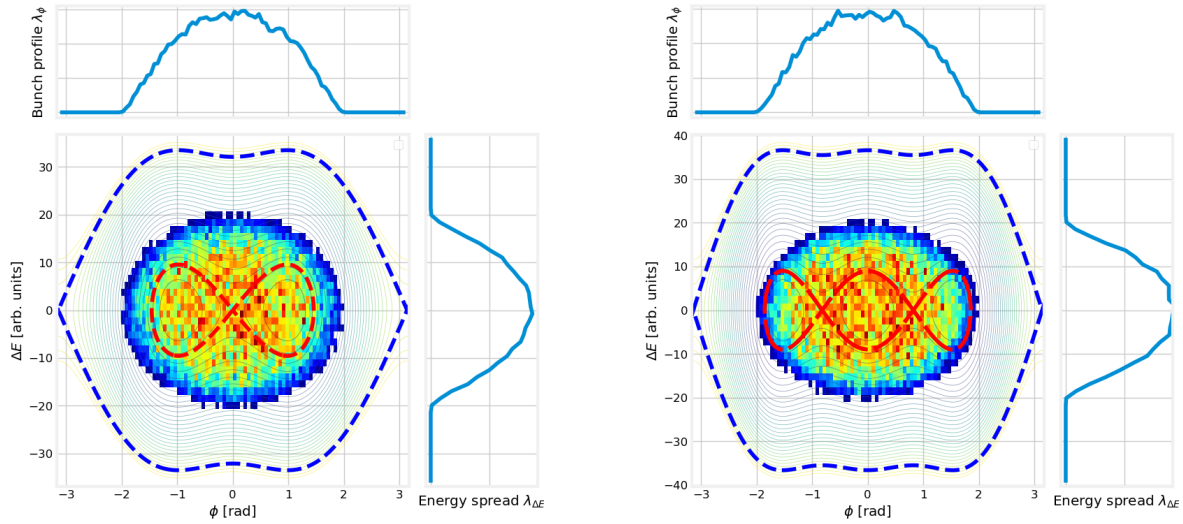


Figure 2.18: Synchrotron frequency ratio for a double-harmonic system in BSM and BLM for multiple r_h where $r = 1/r_h$(non-accelerating) [24]

when the entire bucket is filled, which is good for increasing Landau damping and mitigating the microwave instabilities that can occur at extraction energies in the PSB [2]. This will be discussed in the context of PSB beam types in Chapter 3.

Triple-Harmonic Tracking

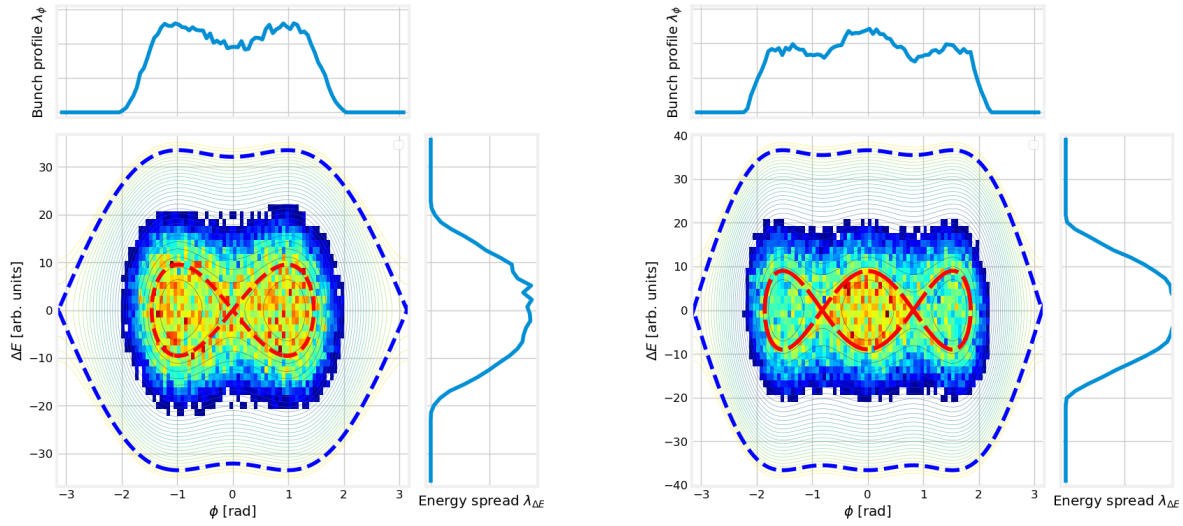


(a) Double-Harmonic **(b)** Triple-Harmonic

Figure 2.19: Initial distributions for particle evolution under double ($r_2 = 0.9$) and triple ($r_2 = 1.3, r_3 = 0.9$) harmonic systems for $\Phi_2 = \pi, \Phi_3 = 0$ rad

Now, in order to compare the double-harmonic to the triple-harmonic operation, it is also interesting to look at the final distributions for the same initial distribution, but evolving under the double and triple-harmonic system. The initial distributions for both are shown in Figure 2.19, whereas the final distributions after evolving for 1000 turns is shown in Figure 2.20. The vertical axes on the initial and final bunch profile plots have been set to the same value in order to be able to compare how λ_ϕ reduces. As can be distinguished in Figure 2.20a and 2.20b, the bunch profile is shorter in the case of the triple-harmonic system as the particles are now oscillating further from the centre of the bucket. This motivates the seek for a quick and efficient solution to setting the parameters in a triple-harmonic system, as now the complexity of adding two more parameters in comparison to a double-harmonic system, becomes higher and more cumbersome for machine operators and RF experts.

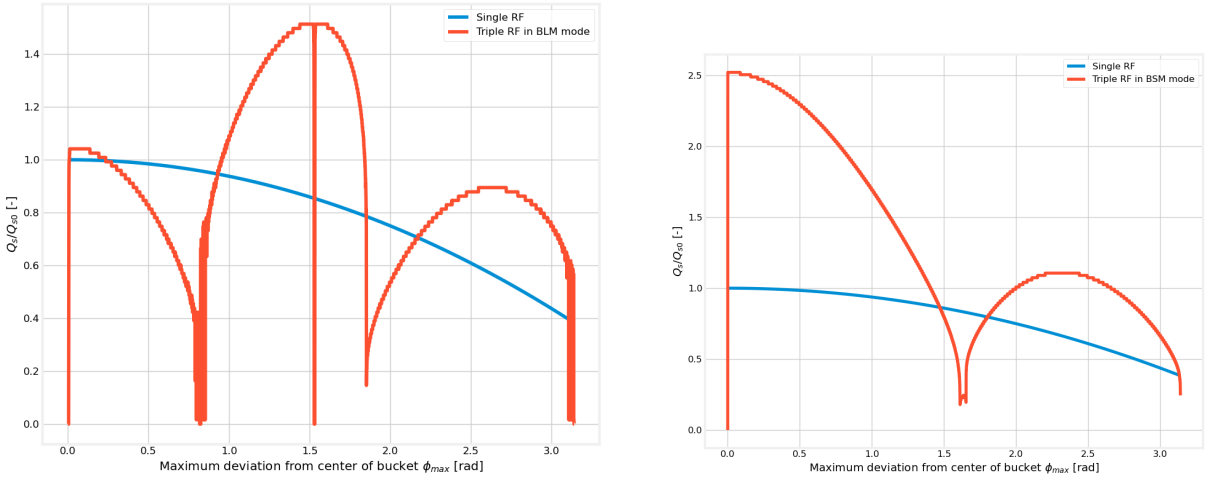
Finally, to complete the discussion, the synchrotron frequency ratio was also numerically computed for the triple-harmonic system in BLM in Figure 2.21a using the tracking code. This ω_s corresponds to the RF system



(a) Double-Harmonic **(b)** Triple-Harmonic

Figure 2.20: Final distributions for particle evolution under double ($r_2 = 0.9$) and triple ($r_2 = 1.3, r_3 = 0.9$) harmonic systems for $\Phi_2 = \pi, \Phi_3 = 0$ rad

in Figure 2.20b, where due to the formation of more loops in the inner separatrix there being 2 stable fixed points and 3 unstable fixed points, there should be 5 points where the synchrotron frequency goes to 0. This is almost achieved in Figure 2.21a, but due to small numerical errors one of the points does not go completely to 0 (it corresponds to the stable fixed point in the right loop of Figure 2.20b). Visibly, in triple-harmonic operation, BLM is even more detrimental to the synchrotron frequency spread, so it should definitely not be used at extraction as it could cause considerable beam instability and losses. However, if microwave instabilities become an issue for high intensity beams such as the LHC Multi-bunch beam types, there is potential to make the synchrotron frequency spread larger by entering BSM as shown in Figure 2.21b, which aids in increasing the resilience of the bunch to this instability mechanism.



(a) BLM ω_s/ω_{s0} **(b)** BSM ω_s/ω_{s0}

Figure 2.21: Numerically approximated synchrotron frequency ratio for a triple-harmonic system in BLM (left) and BSM (right) for ($r_2 = 1.3, r_3 = 0.9$) (non-accelerating)

In conclusion, the incorporation of multiple harmonics (with proper relative phasing) also helps reduce the collective effects that can cause longitudinal instabilities at high beam intensities, I, defined as the number of particles per bunch. The onset of these effects effectively reduce the number of particles which can be packed into a bunch, so correct phasing allows for higher intensities. These collective effects will be discussed below in more detail in relation to selecting the proper parameters for double and triple-harmonic operation in Section 2.3.

2.3. Collective Effects

In the previous section, the tracking simulation was conducted without considering impedance nor collective effects. However, in reality, the beam interacts with its surroundings and itself, leading to electromagnetic fields that influence the beam's energy, stability, and dynamics. These interactions must be considered to better represent the behaviour of the beam in an accelerator. These effects can cause beam instabilities and energy loss, so accurate modelling is essential for understanding beam dynamics and developing the automated tool. This section will introduce three primary types of longitudinal effects: *space-charge*, *resistive-wall impedance*, and *cavity impedance*, which are modelled in the Beam Longitudinal Dynamics (BLonD) code to simulate real-world beam behaviour.

2.3.1. Wakefields and Impedance

When a charged particle beam travels through an accelerator, it generates electromagnetic fields that interact with the surrounding structures, such as the beam pipe, RF cavities, or any geometric discontinuities. These electromagnetic fields, known as *wakefields*, are left behind by the passage of the charged particles and act back on beam, influencing their motion. In the context of the problem, only *longitudinal wakefields*, which affect the energy and phase of particles, will be presented [30].

The concept of impedance provides a way to analyze these wakefields in the frequency domain. Impedance is defined as the *Fourier Transform (FT)* of the wakefield. This means that while wakefields describe the interaction of the beam in the time domain, impedance provides a complementary description in the frequency domain. The longitudinal impedance $Z_{\parallel}(k)$ is related to the longitudinal wake function $W_{\parallel}(s)$ through the following relationship:

$$Z_{\parallel}(\omega) = \int_{-\infty}^{\infty} W_{\parallel}(t) e^{-j\omega t} dt, \quad W_{\parallel}(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} Z_{\parallel}(\omega) e^{j\omega t} d\omega \tag{2.38}$$

where $W_{\parallel}(t)$ (units are $[V/C]$)represents the wakefield induced by a unit charge at a time t behind the leading particle, and ω is the frequency. These relationships indicate that the frequency-dependent impedance characterizes how the beam's induced electromagnetic fields (wakefields) impact the beam itself, and provide ease of computation in comparison to a convolution in the time domain. For a charged particle passing through a cavity which can be modelled through the resonator (RLC circuit) model, the longitudinal wake function is [30]:

$$W(t) = \frac{1}{2\pi} \int Z(\omega) e^{j\omega t} d\omega = 4\pi k_{\mathrm{r}} \left[\cos \tilde{\omega}_{\mathrm{r}} t - \frac{1}{\tilde{\omega}_{\mathrm{r}} T_{\mathrm{f0}}} \sin \tilde{\omega}_{\mathrm{r}} t \right] e^{-t/T_{\mathrm{f0}}} \Theta(t) \tag{2.39}$$

where $\Theta(t)$ is the step function at $t = 0$ imposing causality, $T_{\mathrm{f0}} = \frac{2Q}{\omega_r}$ is the unloaded filling time (time for the field to decay to $\frac{1}{e}$, and $k_{\mathrm{r}} = \frac{\omega_{\mathrm{r}} R_{\mathrm{sh}}}{4\pi \mathrm{Q}}$ is the loss factor of the impedance at frequency ω_r. This wakefield interacts with the voltage being generated by the cavity itself, causing dissipative power loss (through the real part of the impedance) and phase shifts which perturb the shape of the potential well (through the imaginary part of the impedance), but do not cause average power loss.

2.3.2. Space-Charge

Space-charge impedance arises from the Coulomb interactions between particles within the beam itself. At low particle velocities, the repulsive forces due to space-charge can significantly affect the longitudinal and transverse dynamics of the beam, particularly in low-energy accelerators such as the PSB. Given that space-charge is repulsive, it causes particles to obtain a larger off-momentum magnitude $|\delta|$, which causes the particles' revolution frequency to increase or decrease depending on whether the particles are above or below the transition gamma γ_T. For the PSB, the particles are always below transition, meaning that a higher $|\delta|$ will cause a defocusing effect, spreading out the bunch, leading to bunch lengthening, and potentially causing particle loss. As mentioned, this is more important in low-energy accelerators, as in the ultra-relativistic limit the electric fields are essentially perpendicular to the direction of motion, so its longitudinal effect is negligible [30, 50, 24].

Induced Voltage

The longitudinal space-charge can be separated into two components: direct and indirect space-charge. The direct space-charge is the effect of the Coulomb repulsion between the particles, while the indirect space-charge is the interaction of the particles via the environment around it, i.e. the beam pipe. The derivation of the impedance of space-charge involves assuming a perturbation of line density $\delta\lambda_t$ and calculating the electric field generated by the charge perturbation and then using Faraday's law together with the assumption of an inductive beam pipe (see full derivation in [30]). As a result, space-charge impedance can be modelled as a capacitive (i.e. imaginary) impedance at a given harmonic n expressed as [30, 50]:

$$\frac{|Z_{\text{sc}}|}{n} = \frac{Z_0 g}{2\beta_s\gamma_s^2} = \frac{Z_0}{2\beta_s\gamma_s^2}\left(1 + 2\ln\frac{b}{a}\right), \tag{2.40}$$

where Z_0 is the free-space impedance ($377\,\Omega$), b and a are the radius of the beam pipe and the longitudinal beam cross-section, respectively, and g is a geometrical form factor. Furthermore, in Equation 2.40 we retrieve the proportionality mentioned in Chapter 1: the detrimental effect of space-charge scales with $\frac{1}{\beta\gamma^2}$ and higher values of β_s and γ_s (being those of the synchronous particle) will reduce the magnitude of the impedance. However, this equation solely depends on the synchronous momentum, the accelerator geometry and the transverse beam size. To relate it to the longitudinal dynamics, one must consider the induced voltage due to this space-charge effect, which can be defined as follows [50]:

$$V_{\text{sc}}(t) = \frac{e}{\omega_0}\frac{|Z_{\text{sc}}|}{n}\frac{d}{dt}\lambda_t(t) \tag{2.41}$$

Thus, a flatter longitudinal charge density (low values of $\frac{d}{dt}\lambda_t(t)$) over the bucket leads to a lower space-charge induced voltage. This is clearly desired, demonstrating why the phasing of the second harmonic is so important to reduce V_{sc} when considering the discussion had on Figure 2.11. This impedance scales with $\frac{1}{\beta\gamma^2}\frac{d}{dt}(\lambda_t)$, meaning that keeping the gradient of the profile low (reduce any abrupt changes) will reduce the space-charge impedance. However, aside from this induced voltage, which clearly will affect the voltages that are set in the cavities and cause perturbations, a more important effect of space-charge is seen in the transverse plane.

Transverse Tune Shift

Here we must consider the coupled system of a 3D charge distribution in order to see the most critical effect for the PSB (no detailed derivations will be shown as it escapes the scope of the work). In the transverse plane, there is also oscillating motion in the transverse phase space, but now, much like in optical systems, the state is defined by the deviation of the particle from the ideal orbit $\vec{x}$ (or $\vec{z}$) and the angle x' (or z') at which the velocity component v_x (or v_z) is with respect to the on-orbit direction $\hat{s}$: $\cos(x') = \frac{v_x}{|\vec{v}|}$ and likewise for z' (see Figure 2.3). For reference, x' and z' are of order mrad.

Thus, like in synchrotron motion (now called betatron motion for the transverse plane), the particles will have their own betatron frequency $\omega_{\beta_{x/z}}$ and tune $Q_{x/z}$, and will also encounter resonances, but that are more inherent to the magnets and their inevitable imperfections (see p295-302 of [19] for more information). These resonances occur at combinations of $nQ_x + mQ_z = N$ where n, m, and N are integers, and the resonance is of $(n+m)$th order. So, by controlling the strength (and spacings) of the individual magnets that make up the accelerator (see Figure 2.2), one essentially determines the tunes $Q_{x/z}$ such that they are far away from resonances.

The issue is that space-charge causes a tune shift $\Delta Q_{x/z}$ that scales as follows [3, 30, 19, 9, 10]:

$$\Delta Q_{x/z} \propto \frac{\lambda_t}{\beta\gamma^2} \tag{2.42}$$

This causes a spread of the tune that makes the tunes cross resonance lines. This is graphically depicted in the so-called necktie diagram for the PSB at injection in Figure 2.22 ($Q_z = Q_y$) in the graph). Here the tune spread is shown for single (red), double (green) and triple (blue) harmonic operation for the same intensity, where the tune spread is much larger when the peak line density is larger (as it is for the single harmonic bunch). In addition, several resonances are crossed, and this can be partially mitigated by selecting higher working points for the tunes $Q_{x/z} \geq 4.4$, but it still requires mitigation through other tecnhiques like phase-space painting at injection[6], or using triple-harmonic operation.

2.3.3. Resistive-Wall Impedance

Resistive wall impedance is caused by the interaction between the beam and the conducting walls of the accelerator's vacuum chamber. As the beam passes through the vacuum chamber, it induces moving image charges in the chamber walls, which in turn leads to the generation of wakefields that oppose the beam's motion. Additionally, due to the finite conductivity of the wall material, these currents experience resistance. This results in energy loss for the beam and can, again, induce beam instabilities.

The longitudinal resistive wall impedance can be expressed as [30]:

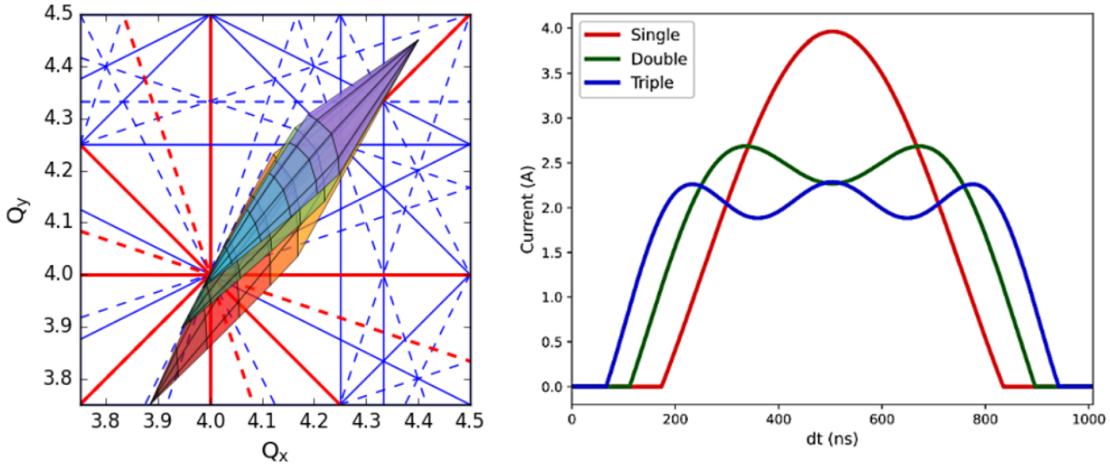


Figure 2.22: Analytically estimated necktie diagram for the PSB injection (left) for the operational high brightness beams for the single (red), double (green) and triple (blue) case. Resonance lines up to 4th order plotted, normal in solid and skew in dashed, systematic in red and non-systematic in blue. The longitudinal bunch profile (right) for single, double and triple-harmonic operation is also shown for the same intensity. [10]

$$Z_{\parallel}(\omega) = (1 + j\,\mathrm{sgn}(\omega))\frac{Z_0\beta}{2b}\left(\frac{|\omega|}{\omega_0}\right)^{1/2}\delta_{\text{skin},0} \tag{2.43}$$

Here $\delta_{\text{skin},0}$ is the penetration depth of the electric field in the material of the beam pipe. Resistive wall impedance has a stronger effect at lower frequencies, making it particularly important for machines where the beam current spectrum has a high component at these low frequencies. This means it is specially important for beams with long bunches or narrow vacuum pipes [19].

2.3.4. Resonator Impedance

Resonator impedance results from the interaction between the beam and resonant structures, such as the RF cavities. The RF cavities are designed to operate at specific resonant frequencies to accelerate the beam efficiently. However, the wakefields of the particles within the beam can also excite higher-order modes (HOMs) in the cavity, leading to unwanted electromagnetic fields that affect the beam.

Resonant impedance is typically categorized into two types: *narrow-band impedance* and *broad-band impedance*. However, they can be represented by an equivalent RLC circuit using the quality factor Q and the shunt impedance R_{sh} to rewrite Equation 2.5 to:

$$Z(\omega) = \frac{R_{\text{sh}}}{1 + j\mathrm{Q}\,(\omega/\omega_{\text{r}} - \omega_{\text{r}}/\omega)}, \tag{2.44}$$

where ω_r is the resonance frequency. Narrow-band impedance occurs typically due to parasitic modes and cavity-like structures in the accelerator which resonate at certain small frequency bands. Broad-band impedance, on the other hand, affects a wide range of frequencies and arises from a combination of geometric discontinuities in the accelerator, such as collimators, beam position monitors, or other non-resonant structures (usually modelled with $Q = 1$). This implies that for the PSB, given the broad-band impedance of the Finemet cavities and Equation 2.39, that wakefields die off quite quickly. Regardless, this model is not entirely accurate for the Finemet cavities, but it does give an intuitive feel for the wakefield decay.

To get to the final induced potential that the beam sees, one must consider the induced voltage V_{ind}, whose computation involves a convolution in time due to the beam passing and affecting itself in the future with its own wakefields. Thus, the computation can be simplified in the frequency domain (convolution in the time domain is a multiplication in the frequency domain) by considering the beam current spectrum $S(\omega)$ which is the FT of the longitudinal charge density $\lambda_t(t)$. Thus, the induced voltage is [50]:

$$V_{\text{ind}}\,(\omega) = -Z_{\parallel}(\omega)S(\omega) \tag{2.45}$$

This would then have to undergo an Inverse Fourier Transform (IFT) to be computed back to the time domain and obtain the effect of wakefields on the voltage inside the resonator. The effect of the beam affecting RF

cavities and the voltage produced is known as beam loading. As a result, the phase of the voltage needs to be adjusted to compensate and fix the potential well, and this effect varies with the intensity of the bunch given that the wakefields are being generated by individual particles. In order to address this effect, the phase of both the 1st and 2nd harmonic can be slightly offset depending on the induced voltage and its phase. This makes it such that the summation of the voltage coming from the amplifier V_{rf} and the induced voltage V_{ind} leads to the desired voltage. This proves why it is relevant to seek an automatized solution, as Φ_2 needs to be adjusted every time the intensity is changed, as the required phasing for BLM (or whatever mode is desired) changes.

However, as mentioned, the impedance model in Equation 2.44 is not fit for the PSB's Finemet cavities, and while there are ongoing works to characterize the impedance with an accurate physical model, the impedance must be defined through measurements and integrated into tracking codes. Regardless, it is important to see how the impedance relates to the voltage that is induced in the cavities by the beam. This becomes even more important when realizing that the **total impedance can be treated as a vector sum of all the impedances in the complex plane** and then transformed into an induced voltage through Equation 2.45.

2.4. Longitudinal Beam Profiles in the PSB

Given how in practice, one cannot easily obtain the separatrix and perturbed potential wells that the particle beam sees in the machine without considerable computational effort, the degree to which parameters are correctly chosen for multi-harmonic operation must be identified solely through the longitudinal bunch profiles. Consequently, it becomes important to identify the variety and statistical properties of the profiles which can be obtained under double and triple-harmonic operation in the PSB that the automatic tool will be looking at. This directly relates to Goal P.G.1, P.G.2, P.G.7 and P.G.8 by analysing the profiles that are produced by the entire span of operational configurations.

Double-Harmonic Bunch Profiles

As was already shown previously, there are several types of potential wells that can be obtained under double-harmonic operation with r_2 and ΔE_s. However, what was not explored is how varying the combination of $(V_{total} = \sum V_i, r_2$ for a given energy gain ΔE_s changes the correct BLM Φ_2 phasing. Indeed, while keeping either one of $(V_{total}, r_2$ and varying the other, the correct phasing changes, even in the absence of impedances of any type. This effect is shown in Figure 2.23 where the the correct phases have been determined in order to flatten the potential well as much as possible or, in the presence of two minimas in the potential well, making the minimas equal to generate a symmetric inner and outer separatrix.

First, observe the plot on the left where $V_{total} = 3.2, 4.8, 6.4$ for the blue, red and yellow curves, respectively, under $r_2 = 0.6$. Thus, while keeping a constant r_2, and increasing the total voltage, the required Φ_2 for BLM increases (or gets closer to its non-accelerating BLM value $\Phi_2 = \pi$). This is because with a higher total voltage, for a given energy gain ΔE_s, the potential $U(\phi)$ does not get deformed as much as for a lower total voltage, so the required change in phasing is closer to its non-accelerating BLM value.

Now, for the case of varying $r_2 = 0.3, 0.6, 0.9$ for the same $V_{total} = 5$ we see that the necessary phasing also changes. The potential wells are stacked on top of each other with lower r_2 having a deeper potential well. This is because V_1 is lower for higher r_2 with a fixed total voltage, and because V_2 is operated in counter-phase it is acting against V_1 which is voltage that accelerates the particles and, thus, increases the bucket area. This is because in BLM V_2 is directed towards shaping the bunch and the potential well, while V_1 is providing the main component of the acceleration. Then, because of the lower V_1, higher r_2 values require a larger shift from the non-accelerating BLM $\Phi_2 = \pi$ because of this effect. It is also apparent when considering the separatrices, as we see that the higher r_2 values have a smaller bucket area because of the smaller V_1. Finally, this is also indicated by the larger ϕ_s for increasing r_2 because when looking at Equation 2.35 with $\Phi_1 = 0$, for the same ΔE_s, a lower V_1 will require a larger ϕ_s.

So, this goes to show that the correct Φ_2 phasing for BLM depends on the combination of these three parameters: $V_{total}, r_2, \Delta E_s$. Then, when one starts to consider the impedance effects, because of the wakefields being proportional to the amount of charge in the bunch, then the correct phasing will also depend on the intensity of the bunch I (and of course the total impedance $Z(\omega)$ of the accelerator). Thus, we arrive at the important conclusion that:

$$\Phi_2(BLM) = f(V_{total}, r_2, \Delta E_s, I), \quad \text{for a given accelerator} \tag{2.46}$$

As for the overall shape of the bunch profile, we have already seen how the emittance ϵ can affect the shape of the bunch by comparing Figure 2.14 and 2.15. To really drive this concept home and show how this happens when having a matched distribution we turn to Figure 2.24 and 2.25 where the formation of a matched distribution is shown for a incorrect and correct BLM Φ_2 phasing, respectively. Here, the concept of *filling factor*, FF, is introduced, which refers to the percentage of the outer separatrix that is filled with particles, such that the emittance of a certain distribution of particles can be defined as $\epsilon_p = FF \cdot A_b$ where A_b is the bucket area. Thus, for a matched distribution ϵ_p increases from the lowest point of the potential $U(\phi)$ up to a certain height in the potential well (like the water analogy used previously), but it could be that the bulk of

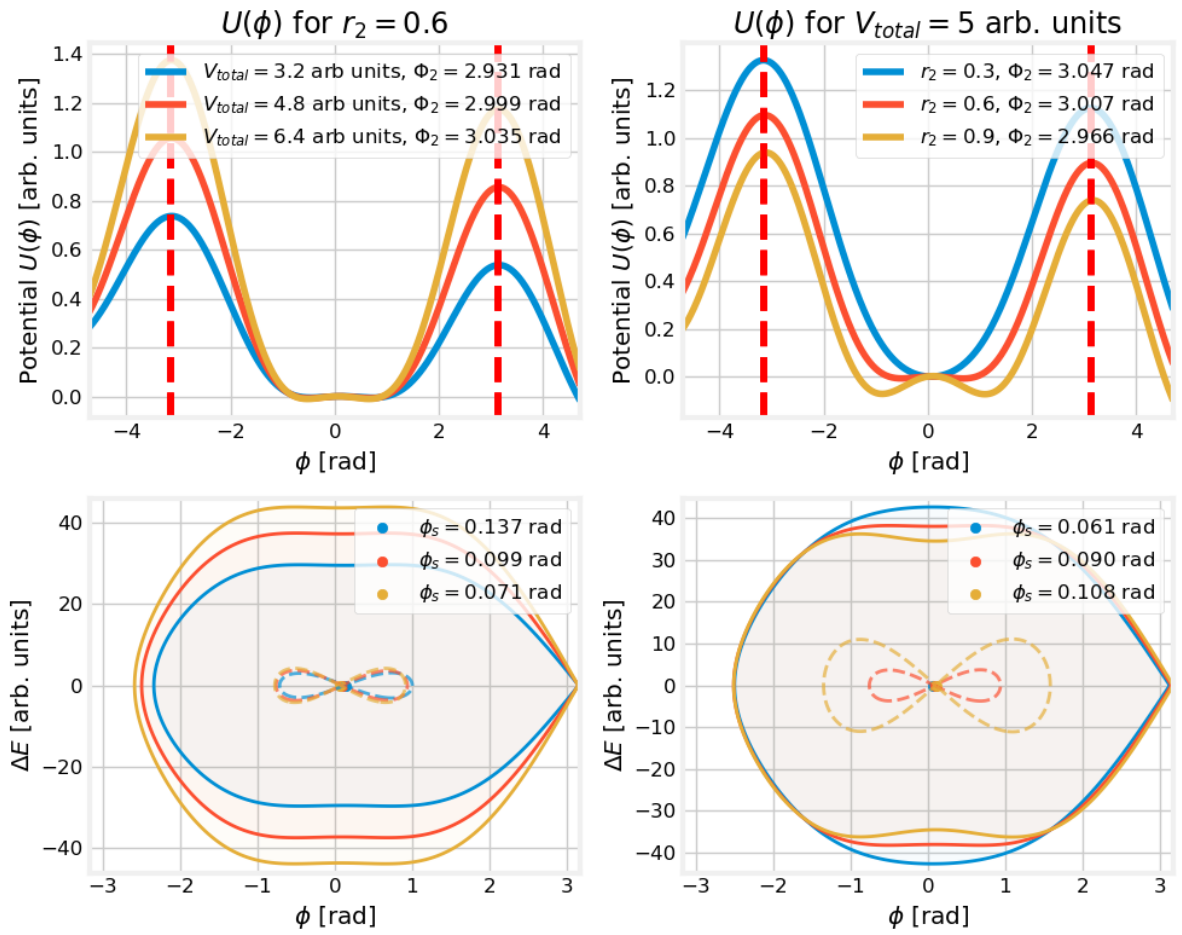


Figure 2.23: Effect of varying V_{total} (left) and r_2 (right) on correct BLM Φ_2 phasing, exemplified through the potential wells and separatrices (red dashed lines indicate a single h_1 period

particles are not completely connected in the ϕ axis. This would occur when the filling factor is low enough and there are two minima such that only the region inside either loop of the inner separatrix is filled. The result of different filling factors on the bunch profile is shown in the right plot of either figure.

The important point here is that the **shape** of the bunch profile λ_t will be determined also by the filling factor FF, so the another important relation comes to light:

$$\lambda_t = f(V_{total}, r_2, \Phi_2, \Delta E_s, I, FF), \quad \text{for a given accelerator} \tag{2.47}$$

where "for a given accelerator" implies for a certain $Z(\omega)$. Finally, this means that we can obtain a large variety of profiles in the PSB, and to exemplify this, Figure 2.26 shows the bunch profiles for a selection of parameter combinations. Here the bunch profile is normalized by the intensity I, which is the same for all plots. Thus, any automatized tool would need to be able to cater to all these shapes, and since the problem is completely geometric in nature, it raises the question if this could be solved through traditional optimization methods.

Traditional optimisation methods for continuous variables (as is the case for Φ_2) can be broken up into *linear*, such as the Simplex method and its variants or interior point search, and *non-linear* methods, such as non-linear programming (NLP) like sequential quadratic programming (SQP) [27]. However, all traditional methods depend on an **objective value**: a figure of merit defining how well phased an RF system is just by looking at the profiles. One could imagine that any of the metrics depicted in Figure 2.27 would aid in this process since the main qualities we look for in profiles are: symmetry, flatness, and a spread out charge distribution.

To that end, we define the metrics:

- **Kullback–Leibler (KL) Divergence** — defines the dissimilarity between the measured bunch profile

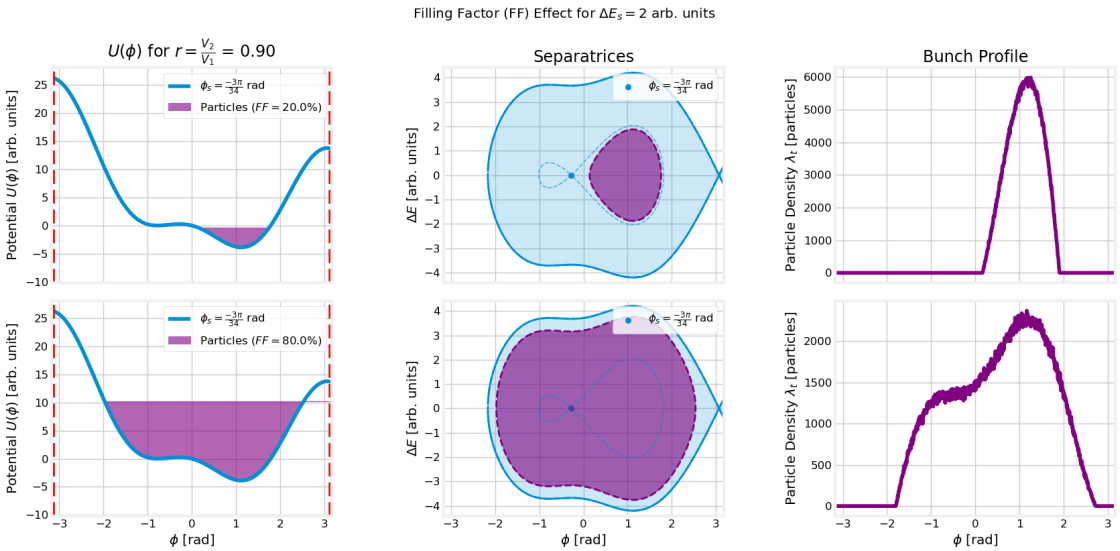


Figure 2.24: Effect of varying filling factor FF in the case of an asymmetric potential and separatrix (incorrect BLM Φ_2 phasing)

$\lambda_t(t)$ and a uniform distribution of particles λ_{mean} (as this would be the most optimal configuration to minimise space-charge), where both are normalised by the intensity (as the metric is defined for probability distributions). Minimising it should drive the profile toward the most spread out and lengthened solution:

$$D_{\text{KL}}(\lambda_t \parallel \lambda_{\text{ref}}) = \frac{1}{I} \int \lambda_t(\phi) \ln \left(\frac{\lambda_t(\phi)/I}{\lambda_{\text{ref}}(\phi)/I} \right) d\phi$$

- **Symmetry** — defines mirror balance of the bunch about the centre of charge ϕ_0. The symmetry of the profile reflects the symmetry of the potential well, so this value should be minimised:

$$S = \frac{1}{I} \int_0^{\pi} |\lambda_t(\phi_0 + \tau) - \lambda_t(\phi_0 - \tau)| \, d\tau$$

- **FWHM** — defines the bunch length at half of its peak. This value is to be maximized in order to be in BLM:

$$\text{FWHM} = \phi_2 - \phi_1, \quad \lambda_t(\phi_1) = \lambda_t(\phi_2) = \frac{1}{2}\lambda_t^{\max}, \quad \phi_1 < \phi_2$$

- **Flatness** — defines how uniform the top 80% plateau of the bunch profile is, but also favours profiles whose flatness extends for a larger domain. Higher flatness again indicates the flatness of the potential well, so this value should be minimised:

$$F = \frac{\lambda_t(\phi)_{\phi \in W} - mean(\lambda_t(\phi)_{\phi \in W})}{\phi_2 - \phi_1}, \quad W = \{\phi_1 \leq \phi \leq \phi_2\}, \quad \lambda_t(\phi_1) = \lambda_t(\phi_2) = 0.8 \cdot \lambda_t^{\max}$$

- **Bunching Factor** — defines ratio between the mean and the maximum line density in the bunch. It should be maximized in order to have the bunch in BLM [15]:

$$b = \frac{mean(\lambda_t(\phi))}{\lambda_{t\max}}$$

Now one could make a combination of these metrics in order to find an objective (or loss) function $\mathcal{L}$ that correctly minimises its value at the correct BLM phasing Φ_2(BLM). In the pursuit of achieving this, the following generalised objective function was defined:

$$\mathcal{L}(\mathbf{w}) = \mathbf{w} \cdot [D_{\text{KL}}, S, -FWHM, F, 1/b]^T \tag{2.48}$$

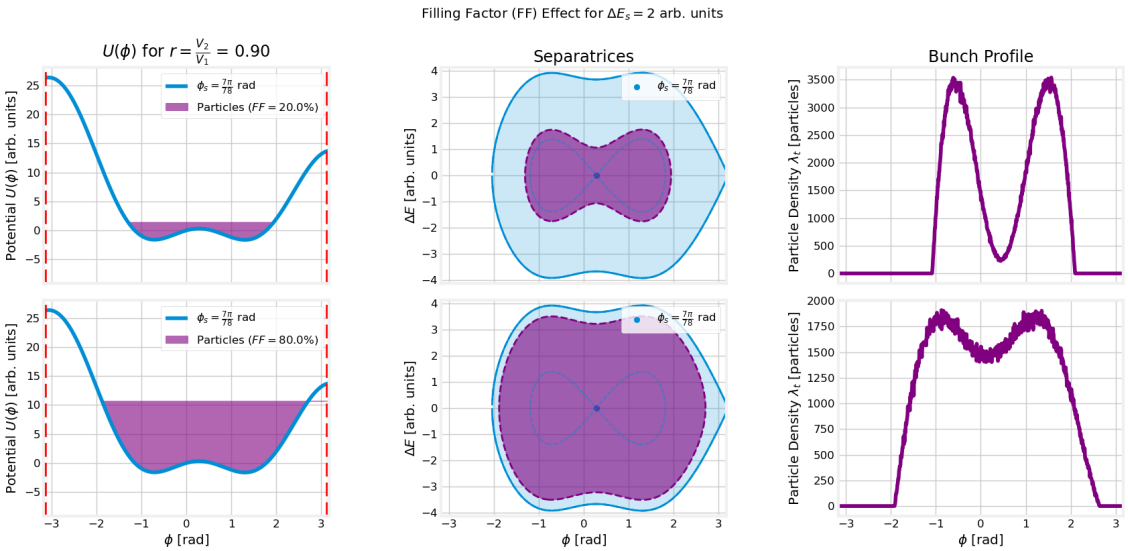


Figure 2.25: Effect of varying filling factor FF in the case of an symmetric potential and separatrix (correct BLM Φ_2 phasing)

with $\mathbf{w}$ a (strictly positive) 6D weight vector. Now, in order to find the optimal weights that would generate such a loss function, the correct Φ_2 phases for BLM for a variety of different parameters were defined through manual phasing (much like how it is currently done in the PSB). Using this dataset, the true deviations of each profile to the optimal phase were defined and normalised by π, yielding the "ground truth", $\Delta\Phi_2^{opt}$. Finally, the weights were optimised through Bayesian optimisation to minimise the difference between the predicted optimum of the loss function and that of the ground truth ($\Delta_1 = |\min_{\Phi_2}(\mathcal{L}(\mathbf{w})) - \min_{\Phi_2}(\Delta\Phi_2^{opt})|$), and the correct prediction of the sign of the gradient at all points such that $\mathcal{L}(\mathbf{w})$ could leverage gradient information when close to the optimum ($\Delta_2 = |\text{sign}(\frac{d}{d\Phi_2}\mathcal{L}(\mathbf{w})) - \text{sign}(\frac{d}{d\Phi_2}\Delta\Phi_2^{opt})|$). Thus, the loss used to optimise the weights was $\Delta = \Delta_1 + \Delta_2$. The full explanation of this process and Bayesian optimisers in general is given in Appendix A.

The results of the attempt to find a general objective function can be derived from Figure 2.28 and 2.29. In the left plot of Figure 2.28, the ground truth $\Delta\Phi_2^{opt}$ is plotted together with the optimised loss function $\mathcal{L}(\mathbf{w}_{opt})$, while on the right the optimal profile and an example of the incorrect profile is shown. As can be seen, the loss function correctly identifies the location of $\Phi_2(BLM)$, and the gradient behaves correctly in the vicinity of the optimum, but breaks down later on due to profiles in BSM being very symmetric and somehow having a lower loss than profiles such as those in the upper-middle plot of Figure 2.26. However, as long as the traditional optimizer used has sufficient exploration behaviour, it should find the minimum in this case.

However, in Figure 2.29, the loss function fails in identifying the minimum and also producing the correct gradients. This occurs at low filling factors because, as shown in Figure 2.25 and 2.24, the matched profiles give a bad representation of the potential well. This can also be seen in essentially all the incorrectly phased profiles at low filling factors of Figure 2.26, they all look almost identical and also they can be confused with a well phased profile at low voltage ratios (as seen in the lower-left plot). As a result, **this justifies the pursuit of Machine Learning (ML) based methods that can identify more features than those possible through simple metrics**. Finally, knowing this, analysing the dependency shown in Figure 2.23 becomes quite important when generating the dataset for these ML methods, as there will need to be a grid scan of parameters in order to generate sufficiently diverse situations such that the neural network can generalize to all operational conditions.

Triple-Harmonic Bunch Profiles

In the case of triple-harmonic operation, there are many different shapes of profiles that can be obtained as now the RF system has more versatility. Essentially, now the bunch shape λ_t adds dimensions to Equation 2.47 and has the following dependency:

$$\lambda_t = f(V_{total}, r_2, r_3, \Phi_2, \Phi_3, \Delta E_s, I, FF), \quad \text{for a given accelerator} \tag{2.49}$$

Thus, it becomes impractical to define all the types of profiles that can be obtained, especially since in this case, there is one singular goal: minimize space-charge by reducing λ_t and $\frac{d}{dt}\lambda_t$ in equations Equation 2.41 and 2.42. Additionally, since operationally space-charge only becomes an issue when the amount of particles

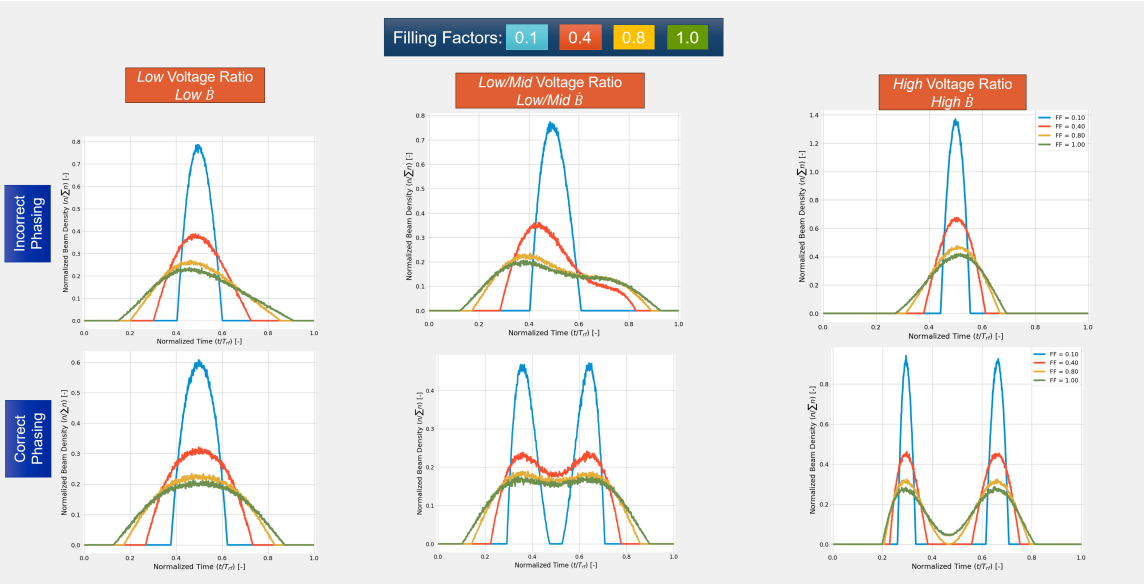


Figure 2.26: Variety of profiles possible with double-harmonic operation, shown with incorrect (top) and correct (bottom) phasing with varying filling factors

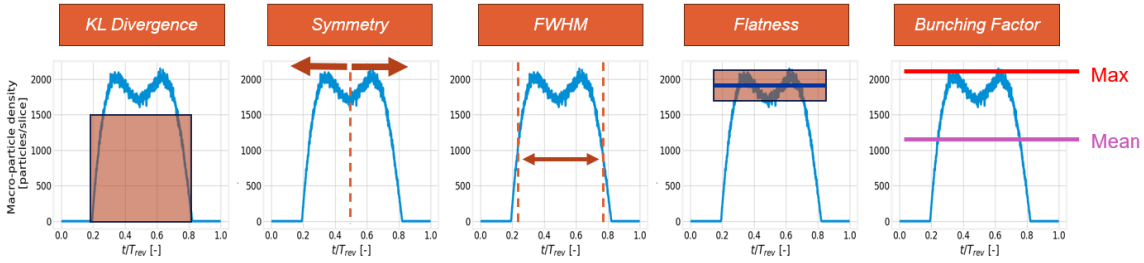


Figure 2.27: Different profile metrics to solving the double-harmonic problem through traditional optimization approaches

is considerable, there is no need to worry about the cases of low filling factors that hinder the definition of an objective value. Consequently, multiple test can be performed in simulation with the aforementioned metrics and a robust black-box optimizer to determine if the metric is fit for optimizing the parameters in a 5D setting, without needing to map out the objective value manifold in 5D. Much like the previous section, a Bayesian Optimizer is fit for the task (now using the Optuna framework defined in Section 4.8 and the procedure outlined in Appendix A).

Given the fact that with triple-harmonic operation there is the possibility to obtain bunches which are not simply connected in the ϕ axis (i.e. we obtain concentrations of charge distribution as indicated by the separatrix on the right plot of Figure 2.13), the FWHM metric is not used as is caused substantial issues. Meanwhile, for the same reason, the flatness metric was also yielding incorrect results and the flatness is better represented by the KL divergence metric anyway. Thus, the three metrics were used across a range of acceleration rates represented by the $\dot{\mathcal{B}}$ in the PSB, with the least complicated case to optimize being $\dot{\mathcal{B}} = 0$ T/s as there are multiple solutions of V_1, V_2, V_3 with $\Phi_2 = \pi, \Phi = 0$ rad for which a flat bunch is achieved under matched conditions. As such, this case is shown in Figure 2.30, where KL-divergence is the metric which best represents a flat distribution. Additionally, to prove that this metric can be used at other acceleration rates, Figure 2.31 shows the optimization results for moderate (1.18 T/s) and high (3.7 T/s) in Figure 2.31a and 2.31b, respectively. Nevertheless, the solution identified after 5000 trials is not optimal, which is suspected to improve if more trials are used. Indeed, running the optimizer several times yields flatter profiles in some cases, while underperforming in others, showing how high $\dot{\mathcal{B}}$ require more fine tuning of the 5D in comparison to lower acceleration rates.

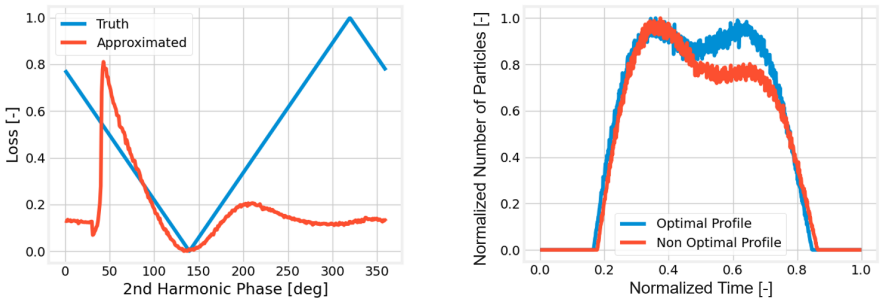


Figure 2.28: Successful identification of minima by the loss function $\mathcal{L}(\mathbf{w})$

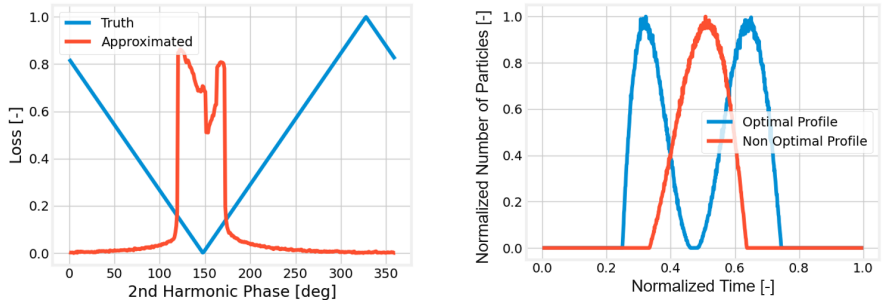


Figure 2.29: Incorrect identification of minima and also incorrect gradients by the loss function $\mathcal{L}(\mathbf{w})$

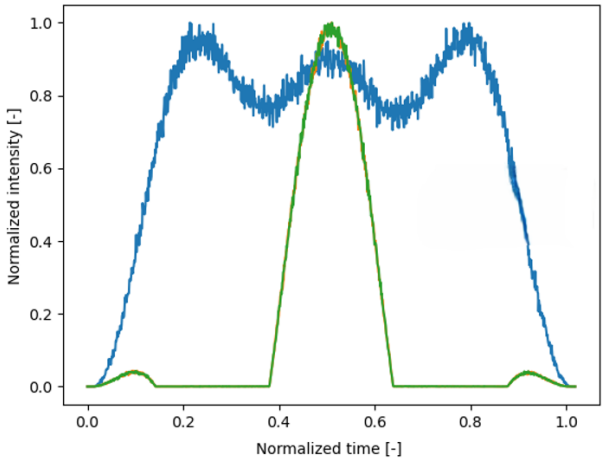


Figure 2.30: Optimal profiles for a non-accelerating bucket ($\Delta E_s = 0$) after 5000 trials for three separate metrics as the objective function to minimise: KL divergence (blue), symmetry (orange) and bunching factor (green)

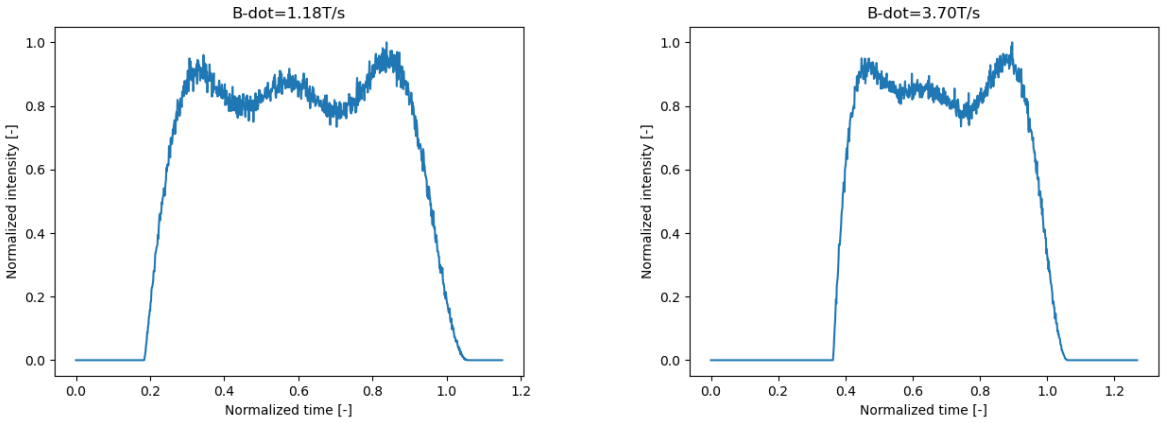
Regardless, this goes to show that triple-harmonic optimization could be achieved using an unsupervised dataset in the case of offline learning, or to generate the data in simulation as parameters are explored in an online learning scenario. This will be important when considering the ML methods used to optimize the 5D parameter space for space-charge mitigation.

2.5. Simulation

Now that all the theory and concepts behind the formation of bunch profiles have been outlined, it is time to consider how all the data will be generated in simulation for the optimization tool. Considering that exploring parameters in the machine does not cater to safety requirements in operations due to the potential of irradiating the beam pipe and the electronics with lost particles, the simulation needs to be as faithful as possible to the physical phenomena that occur in the PSB. Thus, this section caters to how the Beam Longitudinal Dynamics (BLonD) code is set up and its fidelity to the PSB processes.

2.5.1. BLonD Code Components

BLonD is a particle tracking simulator similar to the idealised tracking code that was shown in Section 2.2.3, but highly optimised and with many built in functionalities regarding impedances and feedback loops present in the accelerators [58]. Nevertheless, the feedbacks are not considered, and solely the impedances are modelled



(a) Moderate $\dot{\mathcal{B}}$ **(b)** High $\dot{\mathcal{B}}$

Figure 2.31: Bayesian Optimization solutions after 5000 trials for non-zero acceleration rates ($\Delta E_s \neq 0$) using KL-divergence as the objective function

in the present work. Furthermore, the parameters for the PSB which are used in the code are summarised in Table 2.1 with their respective descriptions, and the main objects that are used in the simulation are shown and described in Table 2.2. The objects are defined in a sequential manner as how they appear in the code, for example the `Ring` object needs to be defined before the `RFStation` object, as the latter uses the revolution frequency derived from the momentum (defined by the magnetic field and the bend radius through Equation 2.8) to determine the frequency of the harmonics. The `RingAndRFTracker` serves as the central component of the BLonD simulation framework. It integrates the models of the beam, RF system parameters, and induced voltages, thereby providing an object that can track the effects of the modelled system on the beam on a turn by turn basis. Nevertheless, full documentation with examples and a guide can be found in the GitLab repository of the code [4].

Table 2.1: Key parameters for PSB modeling in BLonD

Parameter	**Value(s)**	**Unit**	**Description**
Radius R	25.0	m	Mean machine radius
Bend radius ρ	8.239	m	Bending radius of the dipole magnets
Transition γ	4.4	–	Transition Lorentz factor γ_T
Number of particles N_p	0.9×10^{13}	particles	Highest achieved operational intensity
Number of macroparticles N_m	1×10^6	macroparticles	Macro-particles used in tracking
Harmonic numbers h_n	[1, 2] or [1, 2, 3]	–	RF harmonic numbers
Number of RF systems	2 or 3	–	Number of active RF systems

The aspect of the code that is important for its fidelity to the data acquired in the accelerator is how the cumulative impedance is modelled of the space-charge, Finemet, and any broadband resonator components such as discontinuities in the accelerator. The total impedance is as follows [50, 40]:

$$Z(\omega)_{total} = Z_{sc} + Z_{Finemet}(\omega) + Z_{disc} \tag{2.50}$$

$$= \frac{-376.730313462}{\beta\gamma^2} + Z_{Finemet}(\omega) + 34.6669349520904 \cdot \omega[\text{GHz}] \tag{2.51}$$

Table 2.2: Overview of Key Objects in BLonD Implementation [58]

Object	Purpose	Key Defining Parameters
`Ring`	Represents the accelerator ring, including its geometry and beam dynamics properties. The particle type is chosen and the momentum program is set.	`circumference`, `bend_radius`, `alpha_0`, `magnetic_field`, `Particle`
`RFStation`	Models the RF cavities, and their effects on the beam. Configures both voltages and phases over the cycle for all harmonics.	`Ring`, `harmonics`, `voltages`, `phases`
`Beam`	Represents the particles, storing their longitudinal coordinates and energies.	`number_of_particles`, `Ring`, `intensity`
`Profile`	Computes the longitudinal charge density of the beam for each time step.	`Beam`, `bin_width`, `number_of_bins`
`TotalInducedVoltage`	Represents the induced voltages from beam interactions with structures such as Finemet cavities and resistive walls. Includes inputs for impedance models.	`InputTable` for Finemet and Resistive-Wall Impedance, `InductiveImpedance` for Space-Charge Impedance and discontinuities in the beam pipe
`RingAndRFTracker`	Integrates all other objects to simulate the evolution of the beam over time, applying the RF fields and tracking the response of the beam to external voltages. It is the core object that manages the dynamic interaction between the beam and the RF systems.	`beam`, `rf_station`, `total_induced_voltage`, `time_step`

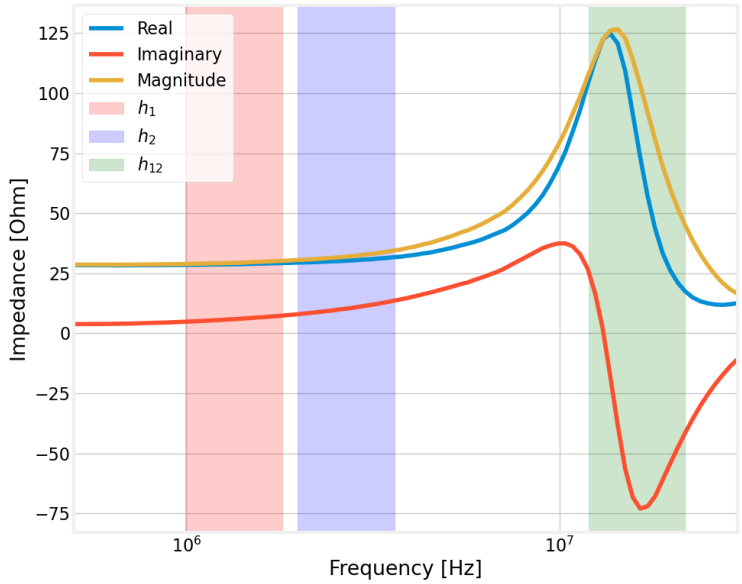


Figure 2.32: Real and imaginary impedance of a single Finemet cell, including some of the relevant ranges of interaction with the beam h_1, h_2 and h_{12} [40]

where $Z_{Finemet}$ is given by an `InputTable` which defines the real and imaginary components at a range of frequencies. This impedance was calculated through single stretched wire measurements in 2024 and is shown for one single cell in Figure 2.32 [40]. Given that this is the impedance for one cell, and there are 4 cells per cavity, 3 cavities per straight section of the PSB and 3 straight sections in a ring, this impedance needs to be multiplied by $4 * 3 * 3 = 36$.

However, this Finemet impedance does not consider the action of the *servoloops*, which are feedback loops that

compare the voltage in the cavity to the desired value, and act to reduce the error through the LLRF (Low-Level RF) system. These servoloops create notches in the real and imaginary impedance shown in Figure 2.32 at the location of the harmonics from h_1 to h_{16} [12]. Also, the notches are dynamic, as the revolution frequency varies significantly over the cycle as the particles are moving faster and the ring circumference is constant. This is shown in the ranges of h_1,h_2 and h_{12} in Figure 2.32. It was found that its effect on the overall bunch profiles can be approximated by dividing the overall real and imaginary impedance by 50. As such, $Z_{Finemet}(\omega)$ is equal to the curves shown in Figure 2.32 but multiplied by a factor of $\frac{36}{50} = 0.72$. On a final note, it is important to note that the space-charge and Finemet impedances Z_{sc} and $Z_{Finemet}$ dominate the impedance contribution, while discontinuities account for a very small portion of the impedance.

2.5.2. Caveat: Bunch Profile Measurement

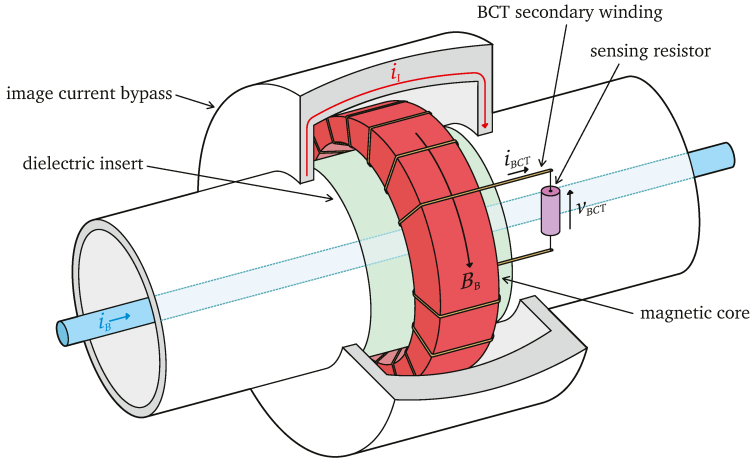
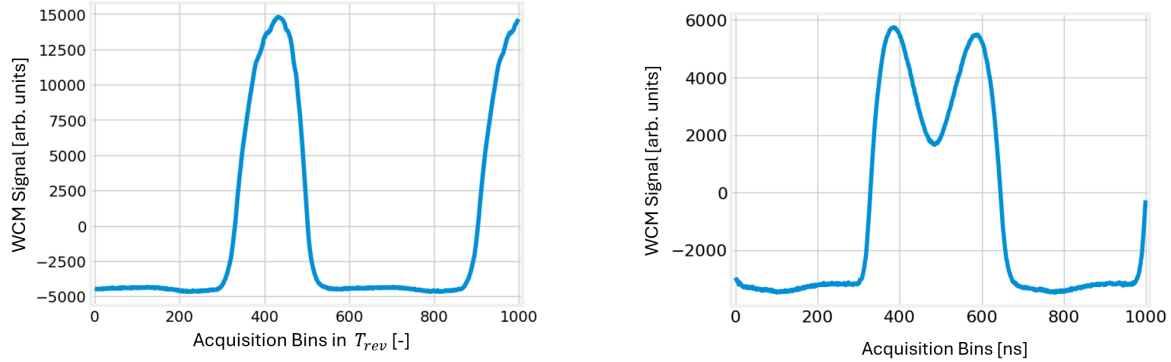


Figure 2.33: Schematic of the Wall Current Monitor (WCM) used in the LHC [28]

In the PSB, Wall Current Monitors (WCMs) are employed to detect the longitudinal beam profile. The issue is that a notable difference is observed between the measured longitudinal bunch profiles and those outputted in simulation. This discrepancy arises from how the beam is detected and the induced wakefields that are created by the bunches.



(a) Low r_2 **(b)** High r_2

Figure 2.34: Induced signal in the PSB's WCM after the bunch detection due to wakefields of the passing bunch

A WCM is installed around the beam pipe of the PSB and consists of conductive electrodes, placed in a segmented cylindrical arrangement as shown in the schematic in Figure 2.33. Note that Figure 2.33 is the WCM in LHC, not the PSB, but it helps to visualize the phenomena that occur with detection in the PSB. As the beam travels through the pipe, it induces image currents on the inner surface of the chamber due to the movement of charges as explained in Section 2.3.3. These image currents are proportional to the instantaneous beam current and generate a corresponding voltage signal across the WCM electrodes. The WCM detects this voltage, converting it into a time-resolved electrical signal that represents the bunch's longitudinal profile [28]. Unlike some other monitors, the WCM is sensitive to the complete frequency spectrum of the bunch, including fast, short-time structure. Its high bandwidth makes it especially suitable for measuring rapid changes in beam current, such as the passage of bunched beams.

Simulated profiles represent only the direct bunch charge distribution, and the effect of the induced wakefields on that charge distribution, assuming an idealized detection that does not account for secondary electromag-

netic effects. In contrast, the real measurements are influenced by the delayed induced voltages generated by the wakefields of the charges in the bunch. The principal manifestation is a baseline distortion or "droop" in the measured signal as shown in Figure 2.34. This droop shifts the base level of the profile measurement away from its flat value, and its temporal evolution does not match the simulated, undistorted profile coming from BLonD. The amplitude and time dependence of this effect are dictated by the chamber geometry, material properties (e.g., skin depth and resistivity), and the frequency spectrum of the bunch [13]. As a result, this phenomenon needs to be considered when generating the simulated profiles and will be discussed in Section 4.1.

3

Requirements Analysis & Definition

This chapter provides a comprehensive requirements analysis for the development of an automated tool for longitudinal bunch profile optimisation in the CERN PSB. As derived from Chapter 2, the tool addresses two primary optimisation challenges: second-harmonic phase optimisation in double-harmonic operation and five-dimensional (5D) triple-harmonic optimisation for V_1, V_2, V_3, Φ_2, and Φ_3. These requirements are derived from the project goals outlined in Section 1.5 and the operational constraints of the PSB beam dynamics environment.

The requirements follow established Product Requirements Document (PRD) guidelines and are formulated to be SMART (Specific, Measurable, Achievable, Relevant, and Time-bound). This systematic approach ensures that the developed tool will meet operational demands while maintaining the safety and reliability standards required for particle accelerator operations.

3.1. Requirements Analysis

3.1.1. Stakeholder Analysis

The primary stakeholders for this automated optimisation tool include:

- **PSB Operations Team**: Requires reliable, fast, and accurate optimisation to maintain operational efficiency across diverse beam types, while also needing fail-safe mechanisms to prevent beam loss and equipment damage.
- **Beam Physics Experts**: Need a tool that preserves the physical understanding of beam dynamics while automating routine optimisation tasks.
- **Experimental Users**: Depend on consistent, high-quality beam delivery for physics experiments (ISOLDE, nTOF, AD, LHC injection).
- **Current and Future Accelerator Projects**: Benefit from generalized solutions applicable beyond the PSB.
- **Control Systems Team**: Benefit from using existing controlling software for the tool to integrate easily with the machine.

All of these stakeholders have to be catered to by the operational tool. As a result, they must be considered by the requirements that are to be fulfilled by the optimisation tool.

Parameter Control in CERN's Accelerators

The last item on the list is quite important and has not been discussed in Chapter 2. The status quo of RF system parameter control, and also the control of any system in the accelerator, is to modify *device* parameters through a variety of means, where a device refers to a specific setting of the accelerator and the sub-settings are referred to as *fields*. The most common means is the Java front-end, available through Graphical User Interfaces (GUIs) on computers connected to the CERN Technical Network. It allows manual modification of the voltage and phase time-series. Parameters can also be set through the Java API (Application Programming Interface), or Java API for Parameter Control (JAPC); with the mass adoption of Python, a Python interface was developed on top of the Java API: PyJAPC [17]. To decouple from the Java API and provide a data model specifying the type of data a device field expects, the Control Systems Team at CERN developed Python Device Access (PyDA) [18]. This is the tool used to set parameters through the Python API and allows the optimisation tool to integrate easily with the control systems, while also allowing for modification of settings should anything change in the future.

3.1.2. Operational Beam Type Requirements

The automated harmonic optimisation system must accommodate the PSB's diverse operational range, which spans three orders of magnitude in intensity (from $\mathcal{O}(10^{10})$ to $\mathcal{O}(10^{13})$ particles per bunch) and longitudinal emittances ϵ ranging from 0.3 to 3 eVs. Each beam type presents unique optimisation challenges and constraints that directly impact system requirements [2].

LHC Multi-Bunch Beams (BCMS and LHC25)

Operational Parameters:

- **Intensity**: around 3.2×10^{12} protons per bunch.
- **Extraction Energy**: 2 GeV.
- **Longitudinal Emittance at Extraction**: 3.0 eVs (highest among operational beams)
- **Primary Objective**: Brightness preservation for High Luminosity (HL) LHC operations.

Optimisation Requirements: These beams represent the most demanding brightness requirements, where transverse emittance blow-up directly impacts LHC beam brightness, as brightness is inversely proportional to the transverse cross-section. As a result, space-charge mitigation is critical during the early cycle phase (275–400 ms), requiring double- (or ideally triple-) harmonic Bunch Lengthening Mode (BLM) operation. The system must maintain voltage ratios and relative phases that minimise peak longitudinal charge density while preserving the large longitudinal emittance required at extraction.

Technical Constraints: The optimisation system must handle space-charge mitigation (early cycle) and transition to the controlled longitudinal emittance blow-up (mid-cycle). Since the tool is to set RF system parameters consistently to achieve BLM, it will not directly cater to the emittance blow-up which is primarily done using h_{10}.

Neutron Time-of-Flight (nTOF) Beams

Operational Parameters:

- **Intensity**: 8.5×10^{12} protons per bunch
- **Extraction Energy**: 2.0 GeV
- **Longitudinal Emittance**: 1.7 eVs
- **Temporal Requirements**: Ultra-short pulses from the PS (7 ns RMS width at target).

Optimisation Requirements: nTOF beams demand a similar optimisation strategy to LHC multi-bunch beams, but have more than double the intensity, so microwave instabilities become more of an issue. Initial BLM operation mitigates space-charge effects during acceleration, followed by extraction with a very low voltage ratio r_2 to maximise synchrotron frequency spread and preserve stability.

ISOLDE High-Intensity Beams

Operational Parameters:

- **Intensity**: Up to 9×10^{12} protons per bunch (highest operational intensity).
- **Extraction Energy**: 1.4 GeV.
- **Longitudinal Emittance at Extraction**: similar to that of nTOF.
- **Primary Objective**: Radioactive isotope production through high-intensity proton bombardment.

Optimisation Requirements: ISOLDE beams present one of the most severe space-charge challenges in the PSB due to the high intensity, requiring considerable longitudinal profile flattening during injection and early acceleration phases [10]. The system must transition from intensive space-charge mitigation at least from injection to C-400 ms (the C refers to C-Time; injection occurs at C-275 ms) to beam stability considerations as energy increases. Triple-harmonic operation becomes essential at maximum intensities, with the optimisation system managing the five-dimensional parameter space while ensuring tune shift remains below resonance-crossing thresholds ($\Delta Q_{x,z} < 0.5$).

SPS Fixed Target Proton (SFTPRO) Beams

Operational Parameters:

- **Intensity**: 2.5×10^{12} protons per bunch
- **Longitudinal Emittance**: 1.3 eVs

- **Unique Requirement**: Bunch splitting in every ring to produce 2 bunches per ring such that the beam is optimised for injection with (2) h_8 RF cavities operating in PS to spread out the beam longitudinally as much as possible while maintaining a constant current for the fixed target experiments.

Optimisation Requirements: SFTPRO beams present the most complex optimisation challenge, requiring longitudinal bunch-splitting operations. These operations require the most precise Φ_2 settings to ensure the same amount of charge per bunch in each ring [2].

LHC Single Bunch Beams (LHCINDIV and LHCPILOT)

Operational Parameters:

- **LHCPILOT**: $0.5 - 2 \times 10^{10}$ protons per bunch, at $\epsilon = 0.2$ eVs
- **LHCINDIV**: $2 - 12 \times 10^{10}$ protons per bunch, at $\epsilon = 0.3$ eVs
- **Primary Objective**: Consistently reproducible low-intensity, low-emittance beam production

Optimisation Requirements: These beams operate outside the primary scope of the automated system due to their exclusive use of fundamental harmonic (h_1) operation.

Antiproton Decelerator (AD) Beams

Operational Parameters:

- **Intensity**: Moderate intensity levels optimised for antiproton production
- **Extraction Scheme**: Ring-specific extraction harmonics (rings 1,2,4 at h=1; ring 3 at h=2)
- **PS Compatibility**: Optimised for PS h_8 capture and subsequent manipulations using h_{21} to only fill one quarter of the ring [34].

Optimisation Requirements: AD beams require ring-specific optimisation capabilities, with the system managing different harmonic configurations across the four PSB rings simultaneously. Ring 3 presents the only unique h_2 extraction requirement and demands specialised bunch splitting for improved antiproton production [34]. As a result, the optimisation system must be able to act independently on rings to cater to these cases, or at least be able to run different instances of itself to handle per-ring parameters.

Operational Constraints and System Integration Requirements

Intensity-Dependent Parameter Relationships: as demonstrated in Section 2.3, due to induced voltages and phase shifts, the system needs continuous parameter adjustment capabilities throughout the acceleration cycle as beam conditions evolve, and also be general enough not to depend on knowledge of impedances. Induced voltages change with intensity, and this will also aid generalisation capabilities across different accelerators.

Filling Factor Considerations: The system must accommodate varying filling factors (percentage of RF bucket area occupied by particles), which directly influence bunch profile characteristics and optimization effectiveness. Low filling factors present particular challenges for traditional optimization metrics, requiring more capable pattern recognition capabilities than statistical metrics allow for.

Collective Effects Integration: Real-time optimisation must account for impedance effects from Finemet cavities, space-charge, and resistive-wall interactions, but without explicit knowledge of them. The system must essentially integrate beam-loading compensation by solely looking at the profiles, as operators do (they do not calculate impedances to perform adjustments).

Operational Window Management: Each beam type requires specific optimisation windows within the acceleration cycle, with some demanding continuous optimisation (ISOLDE) while others require selective activation periods to avoid instability. As such, the tool should be selectively applicable over certain ranges of C-Time without issue.

This operational beam type analysis directly informs the technical requirements for the automated harmonic optimisation system, establishing the foundation for robust, generalisable optimisation algorithms capable of meeting the diverse demands of the PSB and possibly of other accelerators requiring similar capabilities.

3.1.3. Technical Constraint Analysis

Several technical constraints shape the requirements definition:

- **Supercycle Timing**: 30-second operational window for complete optimization cycle to at least be able to apply a correction with the minimum super-cycle efficiency: 1 slot inside of the super-cycle.
- **RF System Limitations**: Mainly relating to total voltage constraints, which is 24 kV at maximum when summed across all cavities, but kept to 20 kV to have a safety margin.
- **Measurement Systems**: The system must be able to cope with the SNR of the profiles captured through the WCM (this is not a limiting characteristic as demonstrated by the profiles shown in Figure 2.34).
- **Computational Resources**: There are processing limitations within the control system environment such that the optimization algorithm does not cause issues with other operational programs (i.e. crash the control system).
- **Safety Considerations**: The optimization tool must not cause significant beam loss and if it does, it must be able to stop the operations and remove the intensity from the particular ring or from all rings.

3.1.4. Risk Assessment & Mitigation

It is important to also analyse the risks that come with automated tools, such that they can be mitigated in software or in hardware. The risks that have been contemplated are the following, along with their likeliness L and severity S in brackets where their values range from H =High, M=Medium and L= Low:

- **Convergence Failure ($L = H$, $S = H$)**: Risk of optimization algorithm failing to find suitable solution and oscillating wildy, possibly causing significant beam loss in doing so.

  Mitigation: When deploying the tool, first perform the necessary corrections at a very low beam intensity, such that if the RF system enters in BSM (for double-harmonic) or does not flatten the bunch (for triple-harmonic), the beam loss is limited to a minimum. Furthermore, for the triple-harmonic optimizer, it must never be able to decrease the total voltage to a minimum of 6 kV such that a separatrix always forms. This can always be scaled down after the optimization is performed if desired (for example for longitudinal shaving or bunch manipulations).

- **Parameter Drift ($L = H$, $S = L$)**: Risk of solution becoming invalid due to machine parameter changes or the devices that are being interacted with no longer existing or have been migrated.

  Mitigation: Consider solutions that do not consider machine parameters and focus solely on profile geometry and generalized machine parameters. Additionally, when impractical or impossible to do so, give sufficient documentation to be able to modify the code where necessary to integrate well with the existing system.

- **Beam Loss ($L = H$, $S = H$)**: Risk of optimization causing excessive particle losses, whether in the converged solution or in the pursuit of that solution.

  Mitigation: There are already systems in place in the accelerator that will remove the injection intensity from a cycle that is causing too much beam loss. However, in the case that the solution varies significantly causing instabilities in the bucket position, the user using the optimizer will be informed and a safe approximation for $\Phi_2(t)$ (for double-harmonic) or $V_1(t), V_2(t), V_3(t), \Phi_2(t), \Phi_3(t)$ (for triple-harmonic) will be used. Additionally, the corrections that can be performed to the parameters will be bounded in some region to avoid large changes.

- **System Integration ($L = L$, $S = H$)**: Risk of incompatibility with existing control system due to being computationally prohibitive.

  Mitigation: Optimizing techniques that require too much computing power will be neglected, and the computing power of the solutions implemented will be monitored in simulation before their implementation in the PSB.

- **Performance Degradation ($L = L$, $S = H$)**: Risk of tool performance degrading under high-intensity conditions and failing to achieve operational luminosity.

  Mitigation: The optimizing tool should be agnostic to the amount of intensity and its generalization to different intensities will be verified in simulation before its implementation in the PSB.

- **Time Constraints ($L = H$, $S = L$)**: Risk of not having enough time during the project to deploy the optimization tool operationally with all desired features.

  Mitigation: Provide sufficient documentation and baseline code such optimization efforts can be contin-

ued, with possible improvements well documented.

3.2. System Requirements

Given the operational beam constraints, the technical constraints presented and the project goals outlined in Section 1.5, while also addressing the risks presented previously, the following functional requirements have been set for the optimisation tool. The requirements in Table 3.1 must all be catered to if the tool is to be used online continuously in a safe and reliable manner. For all requirements, "the system" refers to the optimisation tool and the project goals that the requirement caters to are referenced.

Table 3.1: System Requirements

Req. ID	Requirement Description	Rationale	Project Goals
	Functional Requirements		
	Double-Harmonic Phase Optimiser		
DHF-001	The system shall optimise the second-harmonic phase ϕ_2 to achieve Bunch Lengthening Mode (BLM) operation for all supported beam types.	BLM operation is primarily used in the PSB and is critical for mitigating space-charge effects.	P.G.1, P.G.4
DHF-002	The system shall derive optimal phase corrections from longitudinal bunch profiles and general RF system parameters.	Profile-based optimisation provides machine-independent solutions that are robust to parameter variations and allows for quick inference.	P.G.1,P.G.3, P.G.7, P.G.13, P.G.14
DHF-003	The system shall support optimisation for all operational beam voltage programmes using h_1 and h_2 harmonics, excluding LHCINDIV and LHCCPILOT beams.	Coverage of all major operational beams ensures the tool provides comprehensive operational support.	P.G.1, P.G.7
DHF-004	The system shall handle bunch-lengthening operations for space-charge mitigation and bunch-splitting operations for SFTPRO beams.	Different beam types require different optimisation objectives while using the same RF hardware.	P.G.1, P.G.7
	Triple-Harmonic Optimiser		
THF-001	The system shall optimise five parameters simultaneously: three voltage amplitudes (V_1, V_2, V_3) and two relative phases (Φ_2, Φ_3).	Full 5D optimisation is required to achieve optimal bunch flattening for maximum space-charge mitigation.	P.G.2, P.G.4
THF-002	The system shall achieve longitudinal bunch profiles with three peaks of equal height for optimal space-charge mitigation.	Equal peak heights provide quasi-uniform particle distribution minimising local space-charge.	P.G.2, P.G.4, P.G.15
THF-003	The system shall base optimisation on profile geometry and general RF system parameters.	Geometry-based approach ensures robustness to machine and accelerator variations and quick inference.	P.G.1, P.G.3, P.G.7, P.G.13, P.G.14
THF-004	The system shall maintain optimisation effectiveness across the full range of operational beam intensities.	PSB operates with intensities varying by orders of magnitude requiring adaptive optimisation.	P.G.2, P.G.7
THF-005	The system shall operate effectively from injection (C-275) to extraction (C-805).	Different cycles require space-charge mitigation for different periods.	P.G.2, P.G.8, P.G.14
THF-006	The system shall implement fallback to flattest produced profile when convergence is not achieved.	Ensures safe operation even when convergence cannot be achieved within time constraints.	P.G.8, P.G.10

Continued on next page

Table 3.1 – continued from previous page

Req. ID	Requirement Description	Rationale	Project Goals
THF-007	The system shall operate effectively across bucket area range from 1.3 eVs to 3 eVs under operational beam conditions.	Different emittances require different RF parameter precision for optimal flattening (see Figure 2.25).	P.G.1, P.G.2, P.G.7
	Performance Requirements		
	Timing Requirements		
TIM-001	The system shall complete one full optimisation cycle within one PSB supercycle ($\leq$ 30 seconds), including data acquisition, processing, and correction implementation.	To be at least as performant as operators, it requires super-cycle latency to maintain beam quality throughout operation.	P.G.3, P.G.9
TIM-002	Double-harmonic optimisation shall converge to BLM within at least 30 super-cycles.	Convergence time must balance optimisation quality with operational efficiency requirements.	P.G.3
TIM-003	Triple-harmonic optimisation shall produce a flattened profile within at least 50 super-cycles.	5D optimisation requires more iterations but must still meet operational time constraints.	P.G.3
	Accuracy Requirements		
ACC-001	Double-harmonic phase Φ_2 shall be set within at least ± 3 degrees of optimal BLM phase for high-intensity operations.	Phase accuracy directly impacts bunch lengthening effectiveness and beam stability.	P.G.1, P.G.6, P.G.15
ACC-002	Triple-harmonic parameters shall be adjusted to achieve peak height variations $< 5\%$ between the three profile peaks.	Uniform peak heights ensure optimal space-charge distribution along the bunch.	P.G.1, P.G.6, P.G.11, P.G.15
ACC-003	Long-term optimisation accuracy shall be maintained over at least 30 cycles.	Operational stability requires consistent performance over extended periods.	P.G.2, P.G.7
	Safety Requirements		
SAF-001	The system shall not cause an intensity loss of 20×10^{10} ppb for more than 2 consecutive cycles across all supported beam types and magnetic field ramps.	Beam loss limits protect machine components and ensure operational feasibility.	P.G.8, P.G.10
SAF-002	Upon optimisation failure, the system shall automatically alert the operator, revert to the synchronous phase estimate, and for the triple-harmonic optimiser set second-harmonic voltage ratio to 0.5 ($r_2 = 0.5$) and disable the third harmonic ($r_3 = 0$).	Fail-safe operation ensures beam preservation when optimisation cannot converge and a somewhat flattened bunch in the case of triple-harmonic, as allowing $r_3 \neq 0$ could cause even higher peak line densities λ_t (see right plot in Figure 2.13).	P.G.8, P.G.10
SAF-003	The system shall detect and handle cases where voltage ratio < 0.2 by reverting to the synchronous phase estimate for the double-harmonic optimiser.	Low voltage ratios indicate a poorly defined BLM phase which can lead to phase oscillation.	P.G.8, P.G.10
SAF-004	All optimisation parameters shall be constrained within predefined safe operational limits.	Parameter bounds prevent hardware damage and beam instabilities.	P.G.8, P.G.10

Continued on next page

Table 3.1 – continued from previous page

Req. ID	Requirement Description	Rationale	Project Goals
SAF-005	Operators shall maintain capability to override automatic optimisation and manually control parameters.	Human supervision ensures safety control in abnormal situations.	P.G.10, P.G.11
SAF-006	Before implementation in the PSB, the system shall be verified across all beam types in simulation with varying intensities and major impedances modelled.	Required to ensure that the optimisation tool works in principle, and that there is a certain degree of performance.	P.G.5, P.G.7
	Reliability Requirements		
REL-001	The system shall maintain functionality across all operational beam voltage programmes and magnetic field ramps, excluding when BLM is not well defined.	PSB operates diverse beam types requiring consistent optimisation performance, but the cases where BLM is not well defined are excluded to avoid solution oscillation.	P.G.7, P.G.14
REL-002	The system shall operate effectively from minimum operational intensities up to maximum PSB capacity (9×10^{12} ppb for ISOLDE).	Intensity variations of several orders of magnitude occur during normal operation, which lead to different emittances.	P.G.7
REL-003	The system shall maintain performance under typical accelerator environmental conditions.	This is to exclude cases where there is something malfunctioning in the PSB.	P.G.7
REL-004	The system shall adapt to variability in profile measurement system characteristics and calibration states.	Measurement systems might change over time requiring robust data processing.	P.G.2, P.G.7
REL-005	The system shall automatically recover from transient faults within two supercycles.	Quick recovery minimizes impact of beam losses and increases tool efficiency.	P.G.7, P.G.8
	Compatibility Requirements		
COM-001	The system shall integrate with existing PSB control systems and beam instrumentation.	Seamless integration ensures operational deployment without infrastructure changes and possibility to use in other accelerators with similar systems.	P.G.9, P.G.13
COM-002	The system shall interface with standard PSB data acquisition systems for profile measurements and parameter setting.	Standard interfaces ensure maintainability and compatibility with future upgrades.	P.G.9, P.G.14
	Operational Requirements		
OPR-001	The system shall operate autonomously during bunch-lengthening periods in the cycle with minimal operator intervention.	Automation reduces operator workload and ensures consistent optimisation.	P.G.1, P.G.9
OPR-002	The system shall provide comprehensive diagnostics including optimisation convergence status, parameter values, and performance metrics.	Transparency builds operator confidence and enables troubleshooting.	P.G.11, P.G.12
OPR-003	The system shall log all optimisation activities, parameter changes, and performance data for post-analysis.	Data logging enables performance analysis and continuous improvement.	P.G.11, P.G.15
OPR-004	The system shall provide intuitive operator interface for monitoring, configuration, and manual override functions.	User-friendly interfaces ensure safe and efficient operation.	P.G.9, P.G.11

Continued on next page

Table 3.1 – continued from previous page

Req. ID	Requirement Description	Rationale	Project Goals
OPR-005	The system shall support configuration changes for different beam types and operational scenarios.	Flexibility accommodates diverse operational requirements and future changes.	P.G.14
OPR-006	The system's training framework, parameters and integration with the PSB shall be documented extensively.	Extensive documentation is required for future-proofing and also giving operators insight into how the tool functions.	P.G.11, P.G.12

3.3. Requirements Verification

3.3.1. Optimization Tool Verification & Validation Methodology

The verification process for the optimization tool involves comprehensive testing at multiple levels for all operational beam types to ensure the tool meets all specified requirements before operational deployment.

Simulation Verification

All operational beam types will be verified in simulation using the BLonD physics framework, as specified in requirement SAF-006. This allows comprehensive testing across the full parameter space including:

- All voltage programmes and magnetic field ramps
- Complete intensity range from minimum to maximum values (up to 9×10^{12} ppb for ISOLDE as per REL-002)
- Various initial conditions and beam distributions
- Edge cases and failure scenarios including low voltage ratios (SAF-003)
- Major impedance sources to verify performance under realistic conditions

The simulation environment enables safe testing of extreme conditions and systematic exploration of parameter spaces that would be prohibitive in the actual machine.

Machine Validation

Contrary to the the verification in simulation, in validation procedures, the testing will focus on a representative subset of beam types that cover the operational diversity:

- **ISOLDE**: High-intensity operation with strong space-charge effects, testing the upper intensity limits.
- **nTOF**: Medium intensity with specific bunch structure requirements.
- **BCMS**: LHC-type beam with stringent emittance preservation needs.
- **SFTPRO**: Bunch splitting operation requiring precise phase control for equal particle count in each bunch(DHF-004).

These four beam types represent sufficient variety in:

- Intensity ranges (from 1×10^{11} to 9×10^{12} ppb).
- Bunch profiles (single bunch to multi-bunch operations).
- Optimisation objectives (lengthening vs. splitting).
- Relevant optimisation interval from injection (C-275) to extraction (C-805).
- Bucket areas from 1.3 to 3 eVs as specified in THF-007.

3.3.2. Requirement Verification Criteria

Each requirement category has specific verification criteria, all of this information is shown in Table 3.2. Note that for the accuracy, reliability and one of the operational requirements (OPR-005), they are already satisfied

by previous verification criteria.

Table 3.2: Verification Criteria

Ver. ID	Verification Description	Related Requirements
	Functional Requirements Verification	
VER-F-001	Demonstration of BLM achievement for all supported beam types within ±3 degrees of optimal phase, or at least visual comparison with existing operational phase setting to compare performance.	DHF-001, ACC-001
VER-F-002	Verification of being able to perform only profile-based optimization without requiring specific machine parameters.	DHF-002, THF-003 , OPR-005
VER-F-003	Confirmation of triple-harmonic peak matching within 5% tolerance for all three peaks.	THF-002, ACC-002
VER-F-004	Characterisation of the performance of the achieved solution with validation beam types and intensities across the full operational range.	DHF-003, DHF-004, THF-004, THF-005, THF-007, REL-001, REL-002
VER-F-005	Validation of fallback mechanisms when convergence is not achieved, ensuring system defaults to flattest achieved profile.	THF-006
VER-F-006	End-to-end demonstration of simultaneous 5D optimization with all of V_1, V_2, V_3, Φ_2, Φ_3 adjusted and correctly applied.	THF-001
	Performance Requirements Verification	
VER-P-001	Confirming super-cycle response time with full optimization cycle completed within 30 seconds.	TIM-001
VER-P-002	Convergence testing demonstrating BLM achievement within 30 cycles for double-harmonic optimization.	TIM-002
VER-P-003	Convergence testing demonstrating profile flattening within 50 cycles for triple-harmonic optimization.	TIM-003
VER-P-004	Long-term stability verification over 30 cycles after initial convergence (this necessitates convergence to be achieved first).	ACC-003
	Safety Requirements Verification	
VER-S-001	Beam loss monitoring confirming losses remain below 20×10^{10} ppb limit during optimization.	SAF-001
VER-S-002	Testing of automatic reversion to synchronous phase for double-harmonic optimization and returning to safe settings for triple-harmonic optimization.	SAF-002, SAF-003
VER-S-003	Validation of parameter bounds enforcement when corrections indicate exceeding predefined operational limits.	SAF-004
VER-S-004	Verification of operator override capabilities with successful manual parameter control.	SAF-005
VER-S-005	Reporting tool performance for all operational beams in simulation with impedance models included.	SAF-006
	Integration Requirements Verification	
VER-I-001	End-to-end testing with setting parameters and checking successful parameter application through the Java front-end.	COM-001
VER-I-002	Data flow validation from measurement systems to parameter settings with graphical demonstration of the complete data pipeline.	COM-002
VER-I-003	Performance testing under realistic operational conditions including typical machine variations.	REL-003
VER-I-004	Performance testing using noisier data coming from the Wall Current Monitor (WCM) to verify robustness.	REL-004

Continued on next page

Table 3.2 – continued from previous page

Ver. ID	Verification Description	Related Requirements
VER-I-005	Recovery testing demonstrating automatic recovery from transient faults within two super-cycles.	REL-005
	Operational Requirements Verification	
VER-O-001	User interface validation covering autonomy monitoring, diagnostics, logging, and manual override: UI displays convergence, parameters, and metrics; logs captured and exportable; operator takeover and hand-back succeed.	OPR-001, OPR-002, OPR-003, OPR-004
VER-O-002	Documentation review checklist completed covering training framework, parameters, and PSB integration procedures.	OPR-006

3.4. Requirements Traceability

After the completion of the project, the state of requirement fulfillment with a Requirement Traceability Matrix will be presented in order to give a proper hand-off to the next optimization efforts.

3.4.1. Risk Mitigation Mapping

The requirements framework systematically addresses all identified risks:

- **Convergence Failure**: Mitigated by THF-006 (fallback to flattest profile), SAF-002 (reversion to synchronous phase), TIM-002/TIM-003 (convergence time limits)
- **Parameter Drift**: Mitigated by DHF-002, THF-003 (geometry-based approach), REL-004 (adaptation to measurement variations)
- **Beam Loss**: Mitigated by SAF-001 (loss limits), SAF-004 (parameter bounds), SAF-006 (simulation verification), REL-005 (quick recovery)
- **System Integration**: Mitigated by COM-001, COM-002 (standard interfaces), TIM-001 (performance requirements)
- **Performance Degradation**: Mitigated by REL-002 (full intensity range), THF-004 (intensity adaptability), THF-007 (emittance range coverage)
- **Time Constraints**: Mitigated by OPR-006 (extensive documentation), OPR-005 (configurability), OPR-003 (data logging for future analysis)

3.5. Conclusion

This chapter has established a comprehensive requirements framework for the automated longitudinal bunch profile optimization tool for the CERN PSB. The framework comprises 37 specific requirements organized into six categories outlined in Table 3.1.

Each requirement is justified by clear rationale linking it to operational needs and constraints, with explicit traceability to the 15 project goals established in Chapter 1. The requirements address all identified project risks through specific mitigation strategies embedded within the requirements themselves.

The validation and verification strategy emphasizes a two-tier approach: comprehensive simulation verification using all operational beam types with major impedances modeled (SAF-006), followed by targeted machine validation using four representative beam types (ISOLDE, nTOF, BCMS, and SFTPRO) that encompass the full range of operational scenarios. This approach balances thorough testing with practical constraints of machine time and operational efficiency.

Key performance targets include sub-super-cycle response times (TIM-001), phase accuracy within ± 3 degrees (ACC-001), peak height equalization within 5% (ACC-002), and beam loss limits below 20×10^{10} ppb (SAF-001). The framework ensures operational safety through automatic fallback mechanisms (THF-006, SAF-002), parameter bounds (SAF-004), and mandatory operator override capabilities (SAF-005).

This requirements specification serves as the foundation for the subsequent design, implementation, and validation phases of the project, providing clear success criteria and verification methods for each aspect of the optimization tool's functionality. The comprehensive documentation requirement (OPR-006) ensures knowledge transfer and long-term maintainability of the system.

4

Machine Learning Background

As shown in Section 2.4, traditional optimisation methods fall short when dealing with the control of RF system parameters in double-harmonic operation due to the wide range of beam conditions which it must cater to, and in triple-harmonic due to the complex 5D parameter space that needs to be optimised. As a result, this chapter explores the machine learning (ML) frameworks considered for solving both the one-dimensional (1D) second harmonic phase optimisation problem for routine PSB operations and the five-dimensional (5D) parameter space optimisation for space-charge mitigation, encompassing total voltage, voltage ratios of harmonics, and phases.

This chapter presents the dataset creation methodology and constraints, the selected ML framework, Convolutional Neural Networks (CNN) with attention mechanisms, stratified sampling strategies, reinforcement learning (RL) and memory-augmented RL, possibilities of augmented feature extraction, and hyperparameter optimisation with functional Analysis of Variance (fANOVA).

4.1. Dataset Creation Methodology and Constraints

The selection of appropriate ML frameworks is fundamentally constrained by the practical limitations of dataset generation for multi-harmonic operation. Unlike conventional machine learning applications in other fields within accelerator physics, such as fault detection and diagnosis, where experimental and historical datasets can be readily collected from databases like Next CERN Accelerator Logging Service (NXCALS), multi-harmonic operation requires collecting data and careful consideration of operational safety.

4.1.1. Simulated or Experimentally Acquired Dataset

Since profile datasets are not available in double- and triple-harmonic operation for the PSB, new data would need to be generated in a safe manner. As the optimisation tool would need to learn from good and badly set parameters, there exists the possibility that a certain parameter combination could generate significant beam losses, irradiating the beam pipe. As a result, to be able to safely generate data, a simulation-based approach to dataset generation is needed. This decision is driven by several critical factors:

- **Safety considerations**: Experimental parameter sweeps in operational accelerators risk significant beam losses, potentially leading to component damage and radiation hazards [35].
- **Operational constraints**: The extensive parameter space exploration required would consume prohibitive amounts of beam time.
- **Reproducibility**: Simulated datasets provide consistent, noise-free training examples that enable controlled algorithm development and comparison.
- **Coverage**: Simulation allows exploration of parameter regimes that might be operationally risky or impossible to achieve experimentally.

The simulation framework generates beam profiles using validated accelerator physics codes [58], ensuring that the synthetic data maintains physical consistency with real accelerator behaviour while providing the comprehensive coverage necessary for robust ML training. Furthermore, this imposes specific requirements on the applicable ML architectures due to computational limitations.

4.1.2. Temporal vs. Static Dataset Considerations

The initial consideration for this optimisation problem involved the potential creation of temporal and causal datasets that would capture the dynamic evolution of beam parameters throughout the acceleration cycle and allow the model to learn the temporal dependencies between the parameters. Such datasets would need to span sufficiently broad combinations of total voltage, voltage ratios of the second and third harmonics, their respective phases, and magnetic field ramps. This approach would theoretically enable the application of time-series based methods such as temporal transformers or temporal convolutional networks (TCN) which would compute the entire time-series of the beam parameters (be it only the second harmonic phase or the entire 5D parameter). In practice, it would also enable the model to learn the dependencies between the parameters in the 5D scenario and understand the causal relationships between the profiles obtained and the parameters that led to that profile, which would also allow corrections to be made considering the entire history

of the longitudinal profile evolution at once. This would mean that the magnetic field ramp would be sampled throughout the cycle as the profile would be allowed to evolve between the n_{opt} computation points, dividing the cycle in intervals between injection (C-275) and extraction (C-805).

However, the generation of a temporal dataset becomes prohibitively complex when one considers the number of combinations that would need to be sampled. The combinatorial explosion of parameter combinations across the six-dimensional space given in Equation 2.47 (with ΔE_s being determined by the sampling point in the magnetic field ramp), results in a massive number of required simulations. For the double-harmonic dataset, each combination of total voltage V_{total}, voltage ratios $r_2 = V_2/V_1$, phases Φ_2, and initial filling factors FF_0 and intensity I_0 at n_{opt} (so n_{opt} energy gains ΔE_s) computation points would require a huge amount of temporal samples. Additionally, given the discussion had in Section 2.4 and Equation 2.46, it means that the phase would need to be computed manually at each one of these combinations as a supervised dataset is required.

As an example, let's consider $n_{opt} = 20$, meaning 20 points in which the phase is computed, for a single initial intensity (since we also want to be agnostic to the intensity in order to be able to generalise to different accelerators) and the following amount of discrete sampling in each parameter:

- Total voltage V_{total}: 8 (i.e. from 6 to 20 kV in steps of 2 kV)
- Voltage ratio r_2: 6 (i.e. from to 0.25 to 0.875 in steps of 0.125)
- Phase Φ_2: 360 (i.e. 360° in steps of 1°)
- Initial filling factor FF_0: 6 (i.e. from 10% to 100% in steps of 15%)

This would imply that at every computation point, there would be a total possible combination of $8_{V_{total}} \times 6_{r_2} \times 360_{\Phi_2} \times 20_{n_{opt}} = 345,600$ different full cycle-simulations. If we consider that the initial filling factor is not fixed, but rather is a parameter that is sampled for each full cycle simulation, then the total number of simulations to be performed would be $345600 \times 6 = 2,073,600$. This is a huge number of profiles to be computed, and this doesn't consider the fact that this is only for one initial intensity and one of the possible magnetic field ramps, as there are 2 different operational ramps, but more are being investigated to mitigate space-charge even further [3]. Additionally, the n computation points are now fixed and it could lead to issues if the amount of points is changed when actually running the optimization on the machine, because one would ideally want to sample where the estimated synchronous phase ϕ_s changes considerably or wherever it is relevant (for example right before splitting operations to get improved performance).

As for why the entire cycle would need to be simulated, in order to be able to capture the true dependencies between the parameters, considering as well how many particles would be lost due to large changes in the parameters between successive computation points, the entire ramp would have to be simulated turn by turn, leading to an excessive amount of time to be spent on the dataset generation. For reference, it takes about 30 minutes to simulate an entire ramp which is comprised of about 800,000 turns. This is not as relevant to the double-harmonic optimizer as almost all the parameters would be set a priori, and only Φ_2 would be computed by the tool, but for the triple-harmonic optimizer understanding this relation becomes important. As a result, these constraints make it impossible to generate a temporal dataset in a reasonable amount of time, without a significant amount of computational resources, automatically discarding the idea of time-series prediction models. Additionally, one would have to consider an even larger dataset for the 5D parameter space optimization due to the additional parameters that would need to be considered as shown in Equation 2.49.

4.1.3. Static Dataset Approach

Given these computational limitations, the dataset generation focuses instead on generating individual beam profiles at discrete points in the magnetic field ramp, without considering the temporal dependencies between the parameters. For the double-harmonic dataset, each profile represents a snapshot at a specific $\dot{B}$ value, corresponding to a certain C-Time in the cycle, and then the dataset spans the different combinations of $(V_{total}, r_2, \Phi_2, FF)$. Notice that the intensity is not included again as we want the tool to be agnostic to the intensity to be able to generalise to more accelerators as indicated by Goal P.G.14. This also means that in practice, **the simulation will be done without impedance effects**, since it should not have a significant effect on characteristics that we are looking for in the profiles (symmetry of peaks and lengthening). The accurate impedance modelling described in Section 2.5 is important for the verification procedures described in Section 3.3.1.

This approach alters the nature of the optimization problem from a temporal sequence prediction task to a pattern recognition problem for a given combination of parameters. As a result, the double-harmonic dataset consists of beam profiles λ_t, paired with their corresponding parameter sets $(V_{total}, r_2, \Phi_2, \dot{\mathcal{B}}, FF)$ and label mapping the influencing parameters (for all filling factors) to their the optimal BLM phasing: $(V_{total}, r_2, \dot{\mathcal{B}}) : \Phi_2(BLM)$. This methodology is particularly well-suited for the PSB environment where profiles in individual

points in time are not affected substantially by the wakefields generated by points farther in time due to the quick wakefield decay discussed in Section 2.3.4. However, they will of course be influenced if Φ_2 varies significantly in time causing big shifts in ϕ_s which moves the bucket and could cause mismatches that influence the inference of the tool (see Figure 2.16).

4.1.4. Double-Harmonic Dataset

The double-harmonic dataset is generated through a systematic grid scan of the five key parameters that govern the RF system behaviour. The generation of this dataset enables the possibilities of offline learning to not restrict the possible ML frameworks that can be used Table 4.1 presents the parameter space explored for creating the profiles for the training dataset.

Table 4.1: Double-Harmonic Dataset Grid Scan Parameters

Parameter	Symbol	Unit	Range	Grid Points	Values
Total voltage	V_{total}	kV	6–20	8	6, 8, 10, 12, 14, 16, 18, 20
Voltage ratio	r_2	–	0.25–4	11	0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1, 1.14, 1.6, 2, 4
2nd harmonic phase	Φ_2	∘	0–359	360	0, 1, 2, ..., 358, 359
Magnetic field gradient	$\dot{\mathcal{B}}$	T/s	0–3.70	7	0, 0.76, 1.05, 1.54, 2.10, 2.80, 3.70
Filling Factor	FF	%	10–100	7	10, 25, 50, 60, 75, 90, 100

The comprehensive grid scan results in a total of $8 \times 11 \times 360 \times 7 \times 7 = 1,552,320$ unique profiles, and $8 \times 11 \times 7 = 616$ different optimal BLM phases Φ_2 providing coverage of the operational parameter space. This sampling ensures the machine learning model can accurately interpolate between grid points for any operational scenario encountered in the PSB. To visualise what the magnetic field looks like, its gradient, and how the sampling points lie on it, Figure 4.1 summarises this information. A final caveat of the dataset is that an approximate and normalised synchronous phase $\phi_{s_{approx}} = \sin^{-1}(\frac{\Delta E_s}{V})/\pi$, where $V = \max(V_1, V_2)$, is included in order to give more information as to the acceleration rate of the bunch.

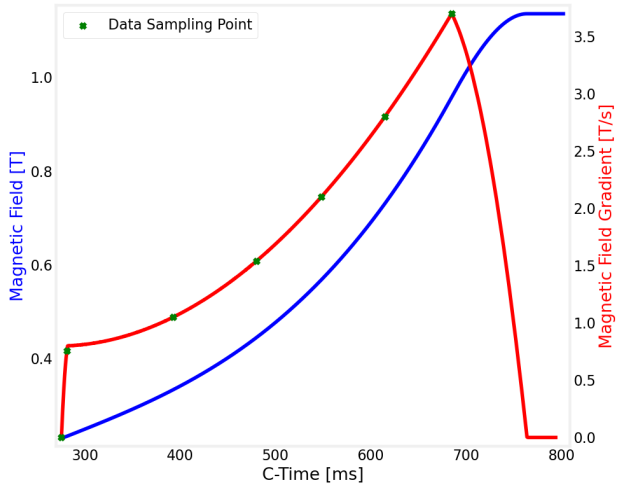


Figure 4.1: Sampling points on the magnetic field $\mathcal{B}$ and its gradient $\dot{\mathcal{B}}$

Finding Optimal Phases

Now, as to how the optimal profiles are identified to create the mapping $(V_{total}, r_2, \dot{\mathcal{B}}) : \Phi_2(BLM)$, it was done considering the characteristics of the separatrices for double-harmonic operation and how the filling factor influences the matched profiles. Consider again Figure 2.25 where the effect of the filling factor can be seen for a voltage ratio r_2 high enough to generate two minima in the potential well, and imagine there is a slight offset in Φ_2 such that the minima in the potential wells are not completely equal. In the case of a low filling factor, the bunch profile would show a much more evident deviation between the height of the peaks than what could be distinguished in the high filling factor case. As such, in these cases, having a low filling factor allows more precision in setting the optimal phase. However, in the case that there no sufficient voltage ratio to generate a double minima, then looking at a low filling factor profile will not give any important information on the

shape of the separatrix as it will just look like a profile with a very small bunch length. As such, in these cases, it is best to look at the profile matched with a high filling factor to be able to discern when the line density λ_t is minimized and provide the best estimate for Φ_2.

In order to be able to generate this mapping in the most efficient way (considering there are 616 different cases that need to be mapped), a GUI was developed. It is shown in Figure 4.2. The GUI has several options which are relevant to mention:

- **Auto Sim**: allows automatically simulating the profile whenever Φ_2 changes.
- **Mode**: allows switching from $FF = 10\%$ (Fine) to 100% (Coarse) depending on what is needed. The Fine mode is shown in Figure 4.3.
- **Finalize Phase**: once the optimum is reached, then this button stores the $(V_{total}, r_2, \dot{\mathcal{B}}) : \Phi_2(BLM)$ entry.
- **Skip**: if it is not possible to find an optimum due to a very high $\dot{\mathcal{B}}$ and a low total voltage, then that parameter combination can be skipped. This helps to not train the tool on combinations that would never occur in practice, and if the combination is removed, then it represents $360_{\Phi_2} \times 7_{FF} = 2520$ profiles less in the dataset.

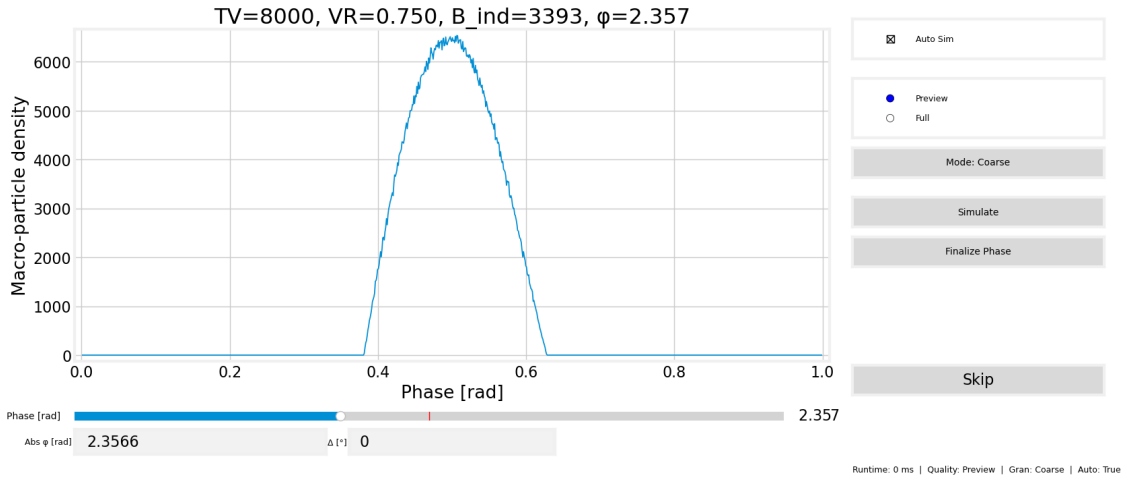


Figure 4.2: UI for finding the optimal Φ_2 under varying conditions (no impedance)

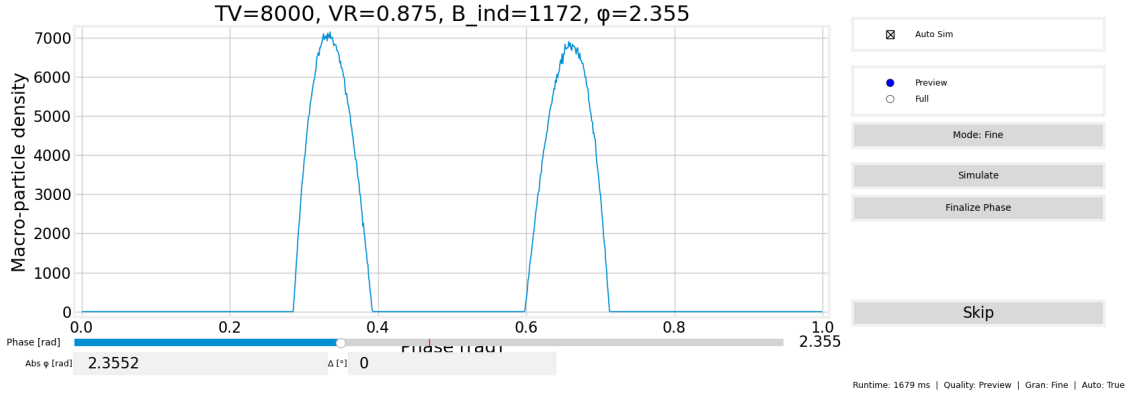


Figure 4.3: Refined estimation of optimal Φ_2 under varying conditions through profile simulation at low filling factors (no impedance)

This act of removing unphysical or impossible to optimise parameter combinations yields the dataset distribution plot shown Figure 4.4. In this plot, the colour represents the highest $\dot{\mathcal{B}}$ that could be obtained with that particular combination of (V_{total}, r_2). As can be expected, with a higher V_{total} it is possible to increase $\dot{\mathcal{B}}$ as there is a higher V_1 to accelerate the particles. However, for increasing r_2, the lower V_1 is and it fails to provide sufficient bucket area for accelerating particles (see Figure 2.23 to see how the bucket area decreases for a given V_{total}). Thus, after pruning all the incompatible combinations, the amount of profiles is now reduced to $1,247,449$ (after also adding the profiles at the optimal phases Φ_2(BLM)), representing a 20% reduction

in the dataset size. Also note that the amount of profiles is not divisible by 360, and this is because of two aspects

1. The optimal profiles matched at $\Phi_2(BLM)$ are added, making a complete parameter group 361 profiles in length. However, the amount of profiles is not divisible by 361 either.

2. There are some Φ_2 values in cases where there is barely any potential well formation due to low total voltage V_{total}, and a voltage ratio r_2 high enough, that changing the phase changes the shape of the RF potential sufficiently such that there is no potential well formation, and therefore no matching possible. As such, some parameter groups contain fewer than 361 profiles.

Finally, Figure 4.5 shows an example of the profiles produced for a specific parameter group at low and high filling factors FF in Figure 4.5a and 4.5b, respectively, where each slice represents a profile normalised by its maximum value phased at a certain Φ_2. Here it can be seen that the profile passes from being phased in BSM to BLM, and it can be appreciated that even in the case of a low $\dot{\mathcal{B}}$, the region in Φ_2 space where BLM features are visible is quite small. This effect becomes even greater at a high $\dot{\mathcal{B}}$ as shown in Figure 4.6, where a change of just 1° in phase has a huge impact on the matched profiles and the gradual changes are lost. This is one of the main reasons why traditional methods fail for these problems as indicated in Section 2.4.

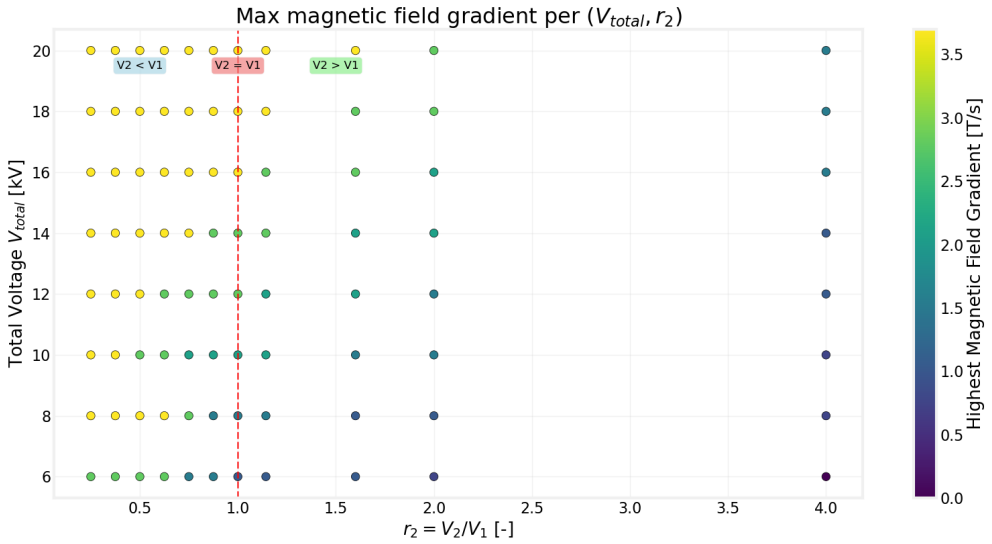
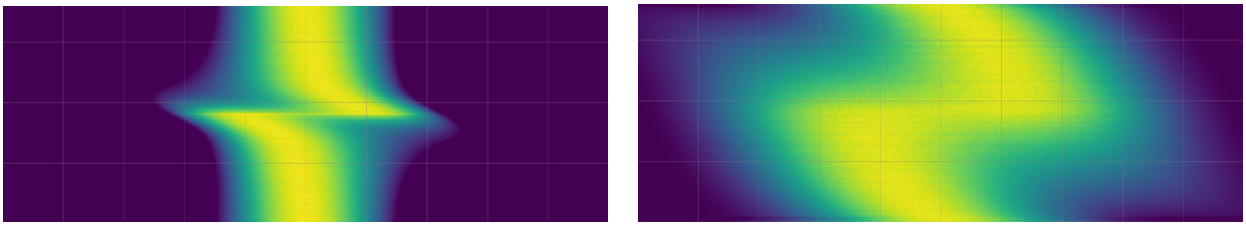


Figure 4.4: Dataset distribution after neglecting non-physical or impossible to optimise $(V_{total}, r_2, \dot{\mathcal{B}})$ combinations



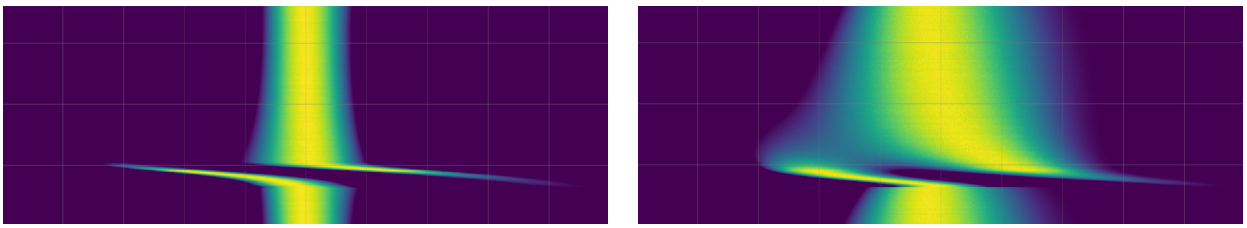
(a) $FF = 25\%$ **(b)** $FF = 100\%$

Figure 4.5: Example of data visualization after processing for $V_{total} = 10$ kV, $r_2 = 0.625$ and $\dot{\mathcal{B}} = 0$ T/s

Physics-based Data Augmentation

Now, to be able to augment the data from this raw dataset (which already represents a size of about 9 GB) with other non-ideal features of profile acquisition in the PSB, there are two methods which are considered: adding the droop discussed in Section 2.5.2, and adding simple Gaussian noise to the profiles. This would make it such that the optimisation tool does not see the profiles acquired from the PSB as "out-of-distribution" inputs.

For the Gaussian noise, it is quite simple as noise is added at each bin in the simulated profiles with a standard deviation $\sigma = 3\% \times \max(\lambda_t)$. Meanwhile, to mimic the droop, the solution has to be a little more involved, but still done quickly and through tensor operations in order to be efficient. As such, the procedure is presented



(a) $FF = 25\%$ **(b)** $FF = 100\%$

Figure 4.6: Example of data visualization after processing for $V_{total} = 10$ kV, $r_2 = 0.625$ and $\dot{\mathcal{B}} = 2.8$ T/s

in Algorithm 1. First, the non-zero portions of the profile are masked, and then a sinusoid S_i with a random phase shift and an amplitude of $2\% \times \max(\lambda_t)$ is added to the non-zero regions, followed by a smoothing Gaussian kernel being passed through the padded profile to make sure no kinks or discontinuities exist in the profile and padded to conserve the same length. The results are shown in Figure 4.7 for a few profiles.

Algorithm 1 Baseline Droop Addition to Longitudinal Beam Profiles

Require: Beam profiles tensor $\mathbf{P} \in \mathbb{R}^{N \times L}$ where N is number of profiles, L is profile length
Ensure: Drooped profiles $\mathbf{P}_{\text{droop}} \in \mathbb{R}^{N \times L}$

Step 1: Generate random phase shifts
$\boldsymbol{\phi} \leftarrow \text{rand}(N) \times 2\pi$ ▷ Random phase for each profile
Step 2: Create sinusoidal droop pattern
$\mathbf{t} \leftarrow \text{linspace}(0, 1, L)$ ▷ Normalized time vector
$\mathbf{M} \leftarrow \mathbf{1}(\mathbf{P} \leq 0)$ ▷ Binary mask: 1 where profile is zero
for $i = 1$ to N **do**
  $\mathbf{S}_i \leftarrow 0.02 \times \sin(\phi_i + 2\pi\mathbf{t}) \odot \mathbf{M}_i$ ▷ Sinusoidal droop
  $\mathbf{P}^i_{\text{droop}} \leftarrow \mathbf{P}_i + \mathbf{S}_i$ ▷ Add droop to profile
end for
Step 3: Normalize profiles
for $i = 1$ to N **do**
  $p_{\min} \leftarrow \min(\mathbf{P}^i_{\text{droop}})$
  $p_{\max} \leftarrow \max(\mathbf{P}^i_{\text{droop}})$
  $\mathbf{P}^i_{\text{droop}} \leftarrow \frac{\mathbf{P}^i_{\text{droop}} - p_{\min}}{p_{\max} - p_{\min}}$ ▷ Min-max normalization
end for
Step 4: Apply Gaussian smoothing
$k \leftarrow 51$ ▷ Kernel size
$\sigma \leftarrow 7.0$ ▷ Standard deviation (in bins)
$\mathbf{x} \leftarrow [-\frac{k-1}{2}, ..., \frac{k-1}{2}]$ ▷ Kernel coordinates
$\mathbf{g} \leftarrow \exp\left(-\frac{\mathbf{x}^2}{2\sigma^2}\right)$ ▷ Gaussian kernel
$\mathbf{g} \leftarrow \frac{\mathbf{g}}{\sum \mathbf{g}}$ ▷ Normalize kernel
Step 5: Pad and convolve
$\text{pad} \leftarrow \lfloor k/2 \rfloor$
$\mathbf{P}_{\text{padded}} \leftarrow \text{reflect\_pad}(\mathbf{P}_{\text{droop}}, \text{pad})$ ▷ Reflective padding
$\mathbf{P}_{\text{droop}} \leftarrow \mathbf{P}_{\text{padded}} * \mathbf{g}$ ▷ 1D convolution with Gaussian
return $\mathbf{P}_{\text{droop}}$

4.1.5. Triple-Harmonic Dataset

Given that the triple-harmonic optimisation does have an objective value which can be optimised for as seen in Section 2.4 (KL divergence), it becomes unnecessary to follow a similar grid scan methodology, not to mention also quite cumbersome in the added dimensionality of the parameter space. For example, if we were to follow the same procedure, but now add the third harmonic voltage ratio r_3 and also the third harmonic relative phase Φ_3 with the same amount of grid points, then we would have now a dataset that is $11_{r_3} \times 360_{\Phi_3} = 3960$ larger, which is far too massive in the order of magnitude of 10 TB. This would now imply a completely prohibitive amount of simulation time to generate such a dataset.

As a result, making use of the KL divergence, we explore methods that are capable of analysing the profiles and then deriving their flatness to know what continuous corrections should be performed in order to improve that

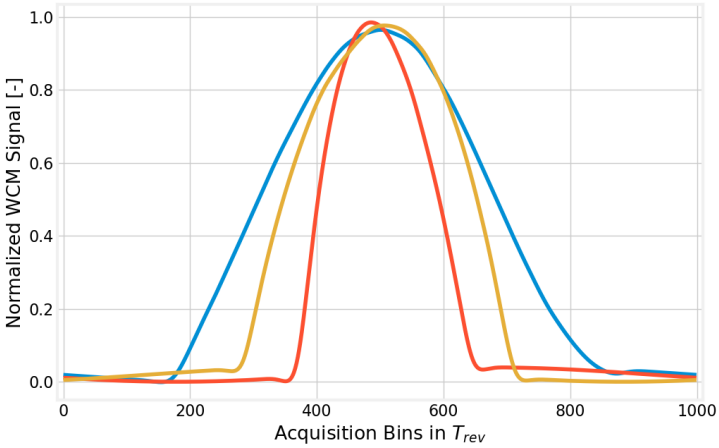


Figure 4.7: Artificial augmentation of simulation profiles to show baseline droop as occurs in real data

flatness. This means that it would follow an **online learning approach**, interacting with the simulation itself and seeing what profiles are obtained after its correction. Furthermore, now that most of the fixed parameters become optimisation variables, the only varying fixed parameter is the acceleration rate exemplified through $\dot{\mathcal{B}}$. The filling factor FF is now removed from consideration as it causes problems with using KL divergence as the measure and it is also consistent with the fact that when triple-harmonic operation is used is when there is a large amount of beam intensity I, which correlates to higher filling factors. This online learning approach would have to integrate the KL divergence as a method to rank the profile or rank the quality of a correction and explore the 5D parameter space using it.

4.2. Artificial Intelligence Framework Selection

The constraints imposed by the static profile dataset approach eliminate several classes of ML methods from consideration. Temporal neural networks, including recurrent neural networks (RNNs), long short-term memory (LSTM) networks applied to time series, and transformer architectures designed for sequential data, become inappropriate due to the absence of temporal dependencies in the dataset structure.

First we look at the methods available for double-harmonic optimisation. It is important to consider that for the optimisation problem, we consider a **correction approach**, instead of determining the absolute phase at which the profile should be phased. This is to be able to generalise more and regress into one possible parameter combination $(V_{total}, r_2, \dot{\mathcal{B}})$ with its particular Φ_2(BLM), and instead work in the space of $\Delta\Phi_2$. Therefore, the remaining viable approaches that are either inherently better at profile analysis or generalising to different types of impedance which can shift the optimum phase Φ_2(BLM) fall into three primary categories:

1. **Convolutional Neural Networks (CNN) with Attention Mechanisms**: Direct regression from beam profiles through convolutional analysis to a phase correction. As a result of the attention mechanisms, it would also give the user information on what the model is paying attention to, making it more transparent to the operators.

2. **Reinforcement Learning (RL) Agents**: Policy-based optimisation treating parameter adjustment as sequential decision-making based on the profile and maximising a reward.

3. **Memory-Augmented Reinforcement Learning**: Extension of RL approaches to incorporate historical correction information.

4.2.1. Benefits and Detriments

Each framework addresses different aspects of the optimization challenge and exhibits distinct advantages and limitations in the context of accelerator parameter optimization. While CNN with attention mechanism, also known as Convolutional Block Attention Module (CBAM), use convolutional layers that provide good signal analysis, they are still regressing a profile to a correction and that limits the generalization capabilities as they are learning directly on the information that is given to them. In this case, the dataset that is being given to train them does not include impedances, so the regression will be dependent on the characteristics of the bucket and how shifted it becomes during acceleration, which as we have seen in Chapter 2 in turn depends on a mixture of accelerator geometry, the magnetic field ramp and the voltages applied.

On the other hand, as RL methods are not focusing solely on a regression, but more on learning a policy

which essentially is a strategy to maximize the reward. So instead of just computing the correct phase correction, they can have more intricate behaviour such as slowly approaching the optimum when necessary, rather than expecting to converge in one single correction. More importantly, RL methods usually employ an online learning approach which is a significant benefit to triple-harmonic optimization considering the dataset principles which make it prohibitive to have offline learning in the first place. As a result, for the triple-harmonic optimization, only the RL frameworks are considered.

Nevertheless, RL agents lack the specialized signal processing that CNNs provide as RL agents with simpler architectures, using only using traditional Multi-Layer Perceptrons (MLP) layers, usually perform better than more complex ones like the CBAM CNNs [53]. On top of this, there is a fundamental incompatibility between training RL agents that employ convolutional layers for feature extraction, as will be explained later. First, the foundations of each framework considered for this work are explained.

4.3. CNNs with Attention Mechanisms

4.3.1. Theoretical Foundation

Convolutional Neural Networks represent a class of deep learning models specifically designed to process data with a grid-like topology, such as images or, in the present case, one-dimensional longitudinal beam profiles [41]. Their architectural bias for spatial feature extraction makes them exceptionally well-suited for identifying characteristic patterns of BLM in longitudinal bunch profiles that correlate with optimal harmonic phase corrections settings as has been shown by numerous applications for even more intricate waveforms such as respiratory signals[**Pan2021AnVentilation**].

Convolutional Operations

The fundamental operation of a CNN is the convolution, which involves sliding a learnable kernel (or filter) across the input data to produce a feature map. For a one-dimensional beam profile $\mathbf{p} \in \mathbb{R}^L$, where L represents the number of longitudinal sampling points (e.g., 1000 bins), the convolution operation with a kernel $\mathbf{w} \in \mathbb{R}^k$, where k is the size of the kernel, at position i is given by:

$$(\mathbf{p} * \mathbf{w})_i = \sum_{j=0}^{k-1} p_{i+j} \cdot w_j + b, \tag{4.1}$$

where b is a bias term. Additionally, the profile can be padded with zero values on either end of the profile such that $(\mathbf{p} * \mathbf{w})$ still has L values (this padding is equal to the integer division $int(k/2)$. Multiple such kernels in a single layer learn to detect different, low-level features. In the context of beam profiles, initial convolutional layers may learn to identify elementary shapes such as sharp peaks, troughs, slopes, and shoulders—features intrinsically linked to the phasing of RF harmonics.

Feature Hierarchy Construction

A core strength of deep CNNs is their ability to construct a hierarchical representation of features. The output feature maps from an initial convolutional layer serve as the input to subsequent layers. This process can be described by:

$$\mathbf{f}^{(l+1)} = \sigma\left(\mathbf{W}^{(l+1)} * \mathbf{f}^{(l)} + \mathbf{b}^{(l+1)}\right), \tag{4.2}$$

where $\mathbf{f}^{(l)}$ represents the feature maps at layer l, $\mathbf{W}^{(l+1)}$ are the learnable weights of the kernels, $\mathbf{b}^{(l+1)}$ are bias terms, and σ is a non-linear activation function such as the Rectified Linear Unit (ReLU), defined as $\text{ReLU}(x) = \max(0, x)$ [25]. Through this process, early layers capture simple, local features (e.g., edges in an image, or a local spike in a beam profile). Subsequent layers combine these simpler features to form more complex, abstract representations (e.g., the overall symmetry of a bunch, the presence of multiple peaks, or the flatness of a distribution top). This hierarchical composition, combined with the fact that the kernel size k can vary per layer, allows the network to develop a rich, multi-scale understanding of the input profile's geometry. So in essence its characteristics can be summarized in the following:

- **Local receptive fields**: Each neuron connects to a small region of the input, so the spatial encoding is preserved.
- **Parameter sharing**: The same filter weights are applied across different input locations, which increases efficiency.
- **Pooling**: Down-sampling operations that reduce dimensionality while preserving important features.

Convolutional Block Attention Modules

To enhance the network's ability to focus on the most important regions of the input profile, self-attention mechanisms can be integrated [43]. These mechanisms compute a set of importance weights, often through a

small neural network, that highlight relevant features while suppressing less informative ones. This CBAM is represented graphically in Figure 4.8.

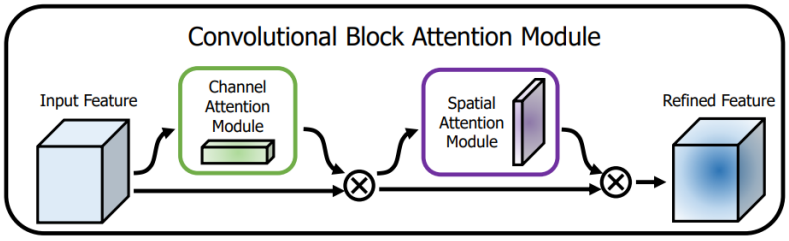


Figure 4.8: Convolutional Block Attention Module [43]

As to how they are defined mathematically, the attention weights α_i for *spatial* locations in this work are computed by taking the mean and the max of each spatial location across all feature maps n_f, and convoluting the result with one convolutional layer with $k = 3$ (Conv_3) and a padding of 1 (to preserve the L bins):

$$\alpha_i = \text{sigmoid}(\text{Conv}_3(\text{mean}(\mathbf{f}_i)) + \text{Conv}_3(\max(\mathbf{f}_i))) \tag{4.3}$$

where $\mathbf{f}_i$ are the values of all feature maps at bin i, while the sigmoid function allows keeping all values in the range $[0, 1]$. The spatially attended feature representation $\mathbf{c_s}$ is then a weighted sum:

$$\mathbf{c_s} = \sum_{i=1}^{L} \alpha_i \mathbf{f}_i. \tag{4.4}$$

This allows the model to adaptively weight different segments of the beam profile based on their relevance to the phase prediction task, effectively ignoring noisy or uninformative regions. Meanwhile, the channel weights are again values in the range $[0, 1]$ for every feature produced. The weights μ^j are computed by summing the components of either averaging or taking the maximum of the entire feature over the spatial dimensions and then passing it through a learnable MLP network that bottlenecks with a certain reduction ratio r_{att}, and then expands out again. So, the learnable MLP network has 3 layers of size $[n_{channels}, int(n_{channels}/r_{att}, n_{channels})]$:

$$\mu^j = \text{sigmoid}(\text{MLP}(\text{mean}(\mathbf{f}^j)) + \text{MLP}(\max(\mathbf{f}^j))) \tag{4.5}$$

where $\mathbf{f}^j$ is the feature map j for all the bins. Thus, the channel attended representation for feature j, $\mathbf{c_c}^j$, is:

$$\mathbf{c_c}^j = \mu^j \mathbf{f}^j. \tag{4.6}$$

Learning Process

The CNN learns a direct mapping from an input beam profile $\mathbf{p}$ to a predicted phase correction $\Delta\hat{\Phi}$. In practice, the network learns to predict the 2D vector $[\cos\Delta\hat{\Phi}, \sin\Delta\hat{\Phi}]$ such that it understand the cyclic nature of the phase space such that $\Delta\hat{\Phi} = \pi = -\pi$ and incorporates this in its loss gradient estimations. The learning process involves minimizing the cosine loss function $\mathcal{L}_{\text{cosine}}$ that quantifies the discrepancy between the prediction $\Delta\hat{\Phi}$ and the true optimal correction $\Delta\Phi^*$. This cosine loss helps enforce the periodic nature of the loss:

$$\mathcal{L}_{\text{cosine}} = 1 - \frac{1}{B}\sum_{i=1}^{B} \cos(\Delta\hat{\Phi}_i - \Delta\Phi_i^*) \tag{4.7}$$

where B is the batch size. The network parameters θ (weights and biases) are iteratively updated via back-propagation and gradient descent to minimize this loss:

$$\theta_{t+1} = \theta_t - \eta\nabla_\theta\mathcal{L}, \tag{4.8}$$

where η is the learning rate. Optimizers like AdamW are typically used to efficiently manage this process [54]. AdamW is a variant of the Adam optimizer that correctly implements weight decay regularization. This helps in not overfitting to the data available. The update rule for AdamW can be formulated as follows:

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1)\nabla_\theta \mathcal{L}(\theta_{t-1}) \tag{4.9}$$

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2)\nabla_\theta \mathcal{L}(\theta_{t-1})^2 \tag{4.10}$$

$$\hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1 - \beta_1^t} \tag{4.11}$$

$$\hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1 - \beta_2^t} \tag{4.12}$$

$$\theta_t = \theta_{t-1} - r_l \left(\frac{\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t} + \epsilon} + \lambda \theta_{t-1} \right) \tag{4.13}$$

Where:

- θ_t represents the parameters at time step t
- $\mathbf{m}_t$ and $\mathbf{v}_t$ are the first and second moment estimates (mean and variance of gradients)
- $\hat{\mathbf{m}}_t$ and $\hat{\mathbf{v}}_t$ are the bias-corrected first and second moment estimates
- β_1 and β_2 are exponential decay rates for moment estimates (typically $\beta_1 = 0.9$, $\beta_2 = 0.999$)
- r_l is the learning rate and its value can dynamically change over the training in a process called *scheduling*. In the current work, the scheduling is done by waiting for 20 epochs to see if the validation loss decreases, if it doesn't by more than 1%, then the learning rate halved.
- ϵ is a small constant for numerical stability (typically 10^{-8})
- λ is the weight decay coefficient (in this case 0.01)

Finally, the entire training procedure for the CNN is outlined in Algorithm 2. Here the profiles are used as input, but the approximated synchronous phase $\phi_{sapprox}$ is also used in order to give more information into how large or small corrections have to be in the presence of acceleration due to the previous analysis exemplified in Figure 4.5 and 4.6. The symbol $\odot$ represents element-wise multiplication. This model is depicted in a graphical sense in Figure 4.9 to give a general idea of the forward pass, but an exact description will be given when the hyperparameter optimization is discussed in Chapter 5

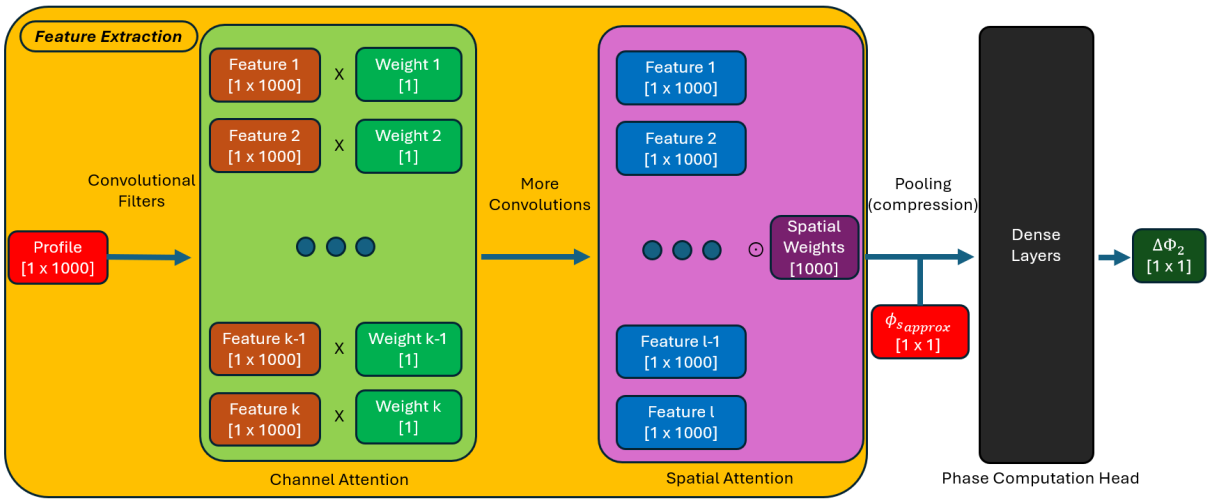


Figure 4.9: CNN model with CBAM attention mechanisms: forward pass

4.3.2. Applicability and Limitations

The CNN with attention approach offers several advantages for the phase correction task:

- **Direct Mapping and Feature Extraction**: The network learns an end-to-end function from raw beam profiles to phase corrections, eliminating the need for manual feature engineering. It autonomously discovers which spatial characteristics of the profile are most indicative of the correct phase.
- **Computational Efficiency**: Once trained, inference is extremely fast, on the order of milliseconds,

Algorithm 2 Training Procedure for CNN with Attention

Initialize network parameters θ
Load beam profile dataset $\mathcal{D} = \{(\mathbf{p}_i, \phi_{s_{approx}}, \Delta\Phi^*)\}_{i=1}^{N}$
Initialize sampling technique
Initialize learning rate r_l and AdamW Optimizer
for epoch = 1 to max_epochs **do** ▷ discussed in Section 4.4
  for each batch $B \subset \mathcal{D}$ **do**
    # Forward Pass
    $\mathbf{f}^{(0)} \leftarrow \text{Input}(\mathcal{B}_{\text{profiles}})$ ▷ Shape: [B, 1, L+1] profiles and $\phi_{s_{approx}}$
    $\mathbf{f}^{(1)} \leftarrow \sigma(\text{Conv1D}(\mathbf{f}^{(0)}))$ ▷ Extract initial features
    $\boldsymbol{\mu} \leftarrow \text{ChannelAttentionModule}(\mathbf{f}^{(1)})$ ▷ Compute channel attention weights
    $\mathbf{f}_{\text{att}}^{(1)} \leftarrow \boldsymbol{\mu} \odot \mathbf{f}_{\text{att}}^{(1)}$ ▷ Apply channel attention
    $\mathbf{f}^{(2)} \leftarrow \sigma(\text{Conv1D}(\mathbf{f}_{\text{att}}^{(1)}))$ ▷ Extract higher-level features
    $\boldsymbol{\alpha} \leftarrow \text{SpatialAttentionModule}(\mathbf{f}^{(2)})$ ▷ Compute spatial attention weights
    $\mathbf{f}_{\text{att}}^{(2)} \leftarrow \boldsymbol{\alpha} \odot \mathbf{f}^{(2)}$ ▷ Apply spatial attention
    $\mathbf{f}_{\text{flat}} \leftarrow \text{Flatten}(\text{GlobalPooling}(\mathbf{f}_{\text{att}}^{(2)}))$ ▷ Max Pool features and reshape to [B,dim($\mathbf{f}_{\text{flat}}$]
    $\Delta\hat{\mathbf{\Phi}} \leftarrow \text{MLP}(\mathbf{f}_{\text{flat}})$ ▷ Final prediction
    # Loss Calculation and Backpropagation
    $\mathcal{L} \leftarrow \mathcal{L}_{\text{cosine}}(\Delta\hat{\mathbf{\Phi}}, \Delta\mathbf{\Phi}_B^*)$
    $\nabla_\theta \mathcal{L} \leftarrow \text{BackwardPass}(\mathcal{L})$ ▷ Compute gradients
    $\theta \leftarrow \text{Optimizer.step}(\theta, \nabla_\theta \mathcal{L})$ ▷ Update parameters
  end for
  Evaluate on validation set
  Check if we have to reduce the learning rate
end for

and can be batched for each individual C-Time to get all corrections at once, making it suitable for integration into real-time control systems with tight latency constraints.

- **Interpretability**: Techniques like Gradient-weighted Class Activation Map (Grad-CAM) can be applied to visualise the attention weights, providing insights into which parts of the beam profile most strongly influenced the network's decision, thereby building operator trust [56].

- **Static Pattern Recognition**: It is perfectly suited for the static dataset methodology, where each data point is an independent snapshot of the beam state.

However, significant limitations emerge when considering the broader context of accelerator optimization:

- **Lack of Convergence Behavior**: The most critical shortcoming is its inherently static nature. The CNN provides a single, one-shot correction based solely on the current profile. It possesses no memory of past actions or states and cannot model the iterative, closed-loop process inherent to tuning the phase, where each small adjustment provides information that informs the next. This can lead to suboptimal convergence or oscillatory behaviour if applied repeatedly.

- **No Exploration Capability**: The model is purely deterministic and predictive. It cannot explore alternative adjustment strategies that might lead to a better overall outcome, as it simply regresses to the average optimal correction from its training data. This means there could be a certain point where it has to compromise between correcting profiles at certain parameter combinations, and not performing well for others.

- **Limited Adaptability**: The CNN learns a fixed mapping. While potentially robust to small variations, it may not adapt well to significant changes in machine state or beam conditions that were not well-represented in the training dataset.

- **No Compatibility with Triple-Harmonic Online Dataset Approach**: The fact that the dataset would need to be generated beforehand in order to be able to determine the relevant loss for all the parameters, makes this approach incompatible with the dataset generation methodology. Additionally, with triple-harmonic optimization, there would be several solutions in the 5D parameter space that will yield the desired flat profiles, albeit with different bucket areas, so this lack of a global minima would also cause issues when defining how to propagate the loss through the network and also cause confusions when making corrections (as both solutions would look practically the same due no potential-well information being used).

This absence of iterative reasoning and memory is the primary motivator for exploring alternative frameworks that can capture the sequential decision-making nature of the tuning process, leading naturally to the formulation of the problem as a task for Reinforcement Learning. First, however, we discuss the sampling strategies for the training and validation datasets for the model.

4.4. Sampling Strategies

Traditional random sampling in machine learning often yields imbalanced datasets that under-represent certain physical configurations, leading to biased neural network models with poor generalisation capabilities. To address this, we implement a *stratified sampling* technique that preserves the distribution of the defining parameters across training and validation datasets while ensuring representative coverage of all operational regimes.

Stratified sampling is a probability sampling method where the population is partitioned into non-overlapping subgroups, known as *strata*, based on shared characteristics. Unlike simple random sampling, which treats all population members equally, stratified sampling ensures specific subgroups are adequately represented in the sample, making it particularly valuable for heterogeneous parameter spaces. This has been shown to improve prediction accuracy in learning tasks by 6 to 8% [55].

4.4.1. Fundamentals of Strata and Stratification

A *stratum* is a subset of the population that is relatively homogeneous with respect to specified characteristics compared to the broader population. In the context of neural network training for double-harmonic optimization, each stratum corresponds to a unique combination of RF system parameters that define distinct operational conditions and different types of profiles.

The stratification process partitions the population into mutually exclusive and collectively exhaustive subgroups. This basically means that every profile, approximate synchronous phase, and phase correction triple $(\mathbf{p}, \phi_{s_{approx}}, \Delta\Phi^*)$ belongs to exactly one stratum and cannot belong to another one simultaneously. Also, this assures that all the triples in the dataset belong to some stratum such that none are excluded. This partitioning is advantageous when subpopulations vary significantly, because independent sampling from each subgroup captures variability more effectively than simple random sampling, which is exactly what is desired for the many beam shapes that the CNN has to deal with.

4.4.2. Multi-Dimensional Stratum Encoding

We encode five parameters into discrete strata:

$$S = V_T, r_2, \dot{\mathcal{B}}, FF, \sigma. \tag{4.14}$$

Here,

- V_T is the total RF voltage,
- r is the voltage ratio between harmonics,
- $\dot{\mathcal{B}}$ is the magnetic field derivative,
- FF is the filling factor of the potential well,
- σ represents the standard deviation of the Gaussian noise added to the profiles, or the amplitude of the sinusoids added to simulate droop. In essence, it represents different types of augmentation.

Each dimension is discretized by identifying unique values within the dataset and mapping them to indices (a sort of 5D variant of the data presented in Figure 4.4):

$$f_{\text{map}}(x) = i \quad \text{where} \quad x \in X_{\text{unique}}[i]. \tag{4.15}$$

where x is the value in that particular dimension (i.e. 8 kV for V_{total}). This converts a continuous parameter space into a finite set of discrete operational states, enabling systematic stratification across all parameter combinations through parameter encoding.

4.4.3. Mixed-Radix Encoding for Compact Label Generation

To manage multi-dimensional stratification efficiently, we employ a *mixed-radix* encoding. Mixed-radix numeral systems use position-dependent bases, making them ideal for encoding multi-dimensional indices with different amount of sampling points within them (also known as cardinalities). The combined stratification label is

$$L = i_{V_T} \cdot (n_r \cdot n_{\dot{\mathcal{B}}} \cdot n_{FF} \cdot n_\sigma) + i_r \cdot (n_{\dot{\mathcal{B}}} \cdot n_{FF} \cdot n_\sigma) + i_{\dot{\mathcal{B}}} \cdot (n_{FF} \cdot n_\sigma) + i_{FF} \cdot n_\sigma + i_\sigma, \tag{4.16}$$

where n represents the number of unique values for each parameter (i.e. the cardinality), and i represents the index of a specific value within its parameter dimension.. This defines a bijective mapping from the multi-index

space to $\mathbb{N}$, enabling efficient grouping and sampling. To understand how Equation 4.16 gives unique indices for each combination of $(V_T, r_2, \dot{\mathcal{B}}, FF, \sigma)$ consider the encoding of a simplified, 2D case in Table 4.2.

Table 4.2: Visualization of mixed-radix encoding for stratified sampling

Parameter Values		**Indices**		**Mixed-Radix**	**Final**
Total Voltage (V)	Voltage Ratio	i_{TV}	i_{VR}	**Calculation**	**Label ID**
200	0.1	0	0	$0 \times 2 + 0$	0
200	0.2	0	1	$0 \times 2 + 1$	1
250	0.1	1	0	$1 \times 2 + 0$	2
250	0.2	1	1	$1 \times 2 + 1$	3
300	0.1	2	0	$2 \times 2 + 0$	4
300	0.2	2	1	$2 \times 2 + 1$	5

This table demonstrates the encoding scheme for a simplified 2-dimensional case with:

- V_{total} with 3 unique values: $[200, 250, 300]$ kV
- r_2 with 2 unique values: $[0.1, 0.2]$

4.4.4. Proportional Allocation Strategy

We adopt *proportional allocation*, where the sample size from each stratum is proportional to the size of that stratum relative to the total population. For a stratum h with N_h profiles contained within it, total amount of profiles N, and desired total sample size n (because we want to split between the training and the validation dataset),

$$n_h = n \times \frac{N_h}{N}. \tag{4.17}$$

This preserves the dataset's natural distribution, which includes less amount of profiles where the matching conditions were much more restrictive, but still ensures that these parameter combinations are not excluded.

4.4.5. Stratified Dataset Division Algorithm

The dataset division maintains proportional representation of all strata in both training and validation subsets via the following procedure, which mirrors the supplied implementation:

Algorithm 3 Stratified Index Splitting for Balanced Dataset Division

Input: Label array L containing stratum identifiers, training fraction f_{train} (80%), random seed
Output: Training indices, validation indices
▷ *Initialization*
Initialize empty dictionary to map each stratum to its sample indices
Initialize empty lists for training and validation indices
Set random number generator seed for reproducibility
▷ *Group samples by stratum*
for each sample index i with corresponding stratum label l_i in dataset **do**
    Add index i to the list of indices belonging to stratum l_i
    This groups all samples with identical parameter combinations
end for
▷ *Proportional split within each stratum*
for each unique stratum s in the dataset **do**
    Retrieve all sample indices belonging to stratum s
    Randomly shuffle these indices to ensure unbiased selection
    Calculate number of training samples (round-up): $n_{\text{train}} \leftarrow \lceil f_{\text{train}} \times \text{number of samples in stratum} \rceil$
    ▷ *Allocate samples to training and validation sets*
    Assign first n_{train} shuffled indices to training set
    Assign remaining indices to validation set
end for
▷ *Combine all strata assignments*
Concatenate training indices from all strata into final training index list
Concatenate validation indices from all strata into final validation index list
Return training indices, validation indices

Using $\lceil \cdot \rceil$ (which is the same as rounding up) ensures very small strata contribute at least one training sample when possible, preventing exclusion of rare conditions, even though it does not happen to that extreme extent in the dataset (perhaps the lowest amount profiles in a parameter group is around 200 profiles).Furthermore, as mentioned before, this balanced sampling approach offers several advantages in the context of machine learning for longitudinal bunch profile control:

- **Enhanced generalization**: exposure to diverse operational conditions improves robustness across the parameter space.
- **Bias reduction**: prevents under-representation of rare yet important situations that could potentially occur in the accelerator, such as low total voltages to perform longitudinal shaving in BCMS beams. Even though at these low total voltages, there will be a lower amount of matched profiles across the $\dot{\mathcal{B}}$ as shown in Figure 4.4.
- **Reliable validation**: stratified validation reflects true operational performance across the different operational regimes, and different types of profiles, giving more confidence in the generalization capabilities of the model.
- **Statistical efficiency**: typically lower variance in the computed solution than simple random sampling when strata differ substantially, as is the case for the different types of profiles generated[55].

4.5. Reinforcement Learning Framework

4.5.1. Theoretical Foundation

Reinforcement Learning (RL) approaches the accelerator optimisation problem as a sequential decision-making task, where an agent learns to select parameter adjustments based on observed beam states [22]. This methodology accommodates the iterative nature of accelerator tuning where one does not expect to immediately converge to the optimal profile in one correction, and provides mechanisms for exploration and adaptation to optimise its behaviour.

Markov Decision Process Formulation

The accelerator optimisation problem is formulated as a Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$:

- **State space** $\mathcal{S}$: Beam profiles and current parameter settings
- **Action space** $\mathcal{A}$: Parameter adjustments $\Delta\Phi_2$, for double-harmonic and $\Delta V_{total}, \Delta r_2, \Delta r_3, \Delta\Phi_2, \Delta\Phi_3$ for triple-harmonic.
- **Transition dynamics** $\mathcal{P}$: System response to parameter changes (in this case it is the profiles coming from from the phase correction)
- **Reward function** $\mathcal{R}$: Beam quality metrics

Central to many RL approaches is the concept of value functions, which estimate the long-term expected reward for taking specific actions in particular states. The idea, in general, is to maximize this value as it should relate to good performance of the algorithm:

$$Q(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_{t+1} \middle| s_t = s, a_t = a\right] \tag{4.18}$$

Where:

- $Q(s,a)$ represents the expected cumulative reward
- s is the current state
- a is the action taken
- γ is the discount factor (typically between 0 and 1), higher values indicate rewards down the line have a higher influence on the cumulative reward, and vice-versa.
- $\mathcal{R}_{t+1}$ is the reward received at timestep $t+1$

Other important concepts in RL are:

- **The Agent**: The decision-making entity that interacts with the environment, it usually attempts to follow what it believes will maximize $Q(s, a)$.
- **The Environment**: The system in which the agent operates, which in this case is the BLonD simulation code.

Furthermore, there are several RL algorithms that can be used to achieve this goal, and they are tailored to either discrete, continuous or binary action spaces [52]. In the current task, we look for something that can handle continuous action spaces such that it can theoretically achieve any type of accuracy, given that the model converges to such a degree. As a result, given its many advantages in stability during training through several tricks, Twin Delayed Deep Deterministic Policy Gradient (TD3) is used, but it is also important to explain how it inherits from its predecessor Deep Deterministic Policy Gradient (DDPG).

Policy Gradient Methods

Policy gradient methods directly learn a parametrized policy $\pi_\theta(a|s)$ (parametrized by θ) that maps states to actions, also referred to as the "actor". Unlike value-based methods that first estimate values using a parametrized function $Q_\theta(s, a)$ (like deep Q-learning), also referred to as the "critic", and then derive a policy to maximize those values (simply selecting the action with the highest associated Q-value), policy gradient methods optimize the policy parameters directly[52]. Here the objective is to maximize the expected cumulative reward by finding the right parametrized policy π_θ:

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\mathcal{R}(s_t, a_t)\right] \tag{4.19}$$

Deep Deterministic Policy Gradient (DDPG)

The previous two methods set the stage to explain DDPG, which bridges the gap between policy gradient methods and deep Q-learning, designed for continuous action spaces[22]. It introduces two key neural networks:

- **Actor Network**: Learns the policy $\pi(s|\theta^\pi)$
- **Critic Network**: Estimates the Q-value $Q(s, a|\theta^Q)$

Furthermore, the actor network is trained by using the Q-values estimated by the critic network. DDPG employs an *off-policy* learning approach, meaning it can learn from previously collected experiences stored in a replay buffer which do not correspond to the actions taken by the current policy (i.e. they relate to previous policies). Additionally, the key innovations in DDPG include:

1. Deterministic policy for continuous action spaces.
2. Separate target networks for stability, which will be clearer after introducing the loss function.
3. Experience replay to de-correlate training samples, as sequential samples of the environment are usually highly correlated as a certain action is related to the previous one. For example, following a big correction to get close to the optimum, will ideally lead to a smaller correction to slowly approach the optimum.

The DDPG loss function for the critic network is formed by attempting to reduce the Bellman error (also called temporal difference or TD error) and can be expressed as:

$$L(\theta^Q) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[(r + \gamma Q'(s', \pi'(s')) - Q(s, a))^2\right] \tag{4.20}$$

Where:

- $\mathcal{D}$ represents the replay buffer.
- Q' and π' are target networks for the critic and the actor, respectively.

This loss function can be propagated backwards much like how it was done in the case of the CNN, and the parameters can be updated. Here, the relevance of the target networks can be noted, as if Q and Q' where parametrized by the same θ, then when minimizing the loss function, this would cause instability, as $\gamma \approx 1$. Thus, in DDPG, the parameters of the target networks generally lag behind the actual critic to avoid this issue.

Despite its innovations, DDPG suffers from several limitations: overestimation of Q-values (also called over-estimation bias), sensitivity to hyperparameters of actor and critic networks, and potential instability in the

minimization of the loss function in Equation 4.20 caused by the overestimation of Q-values, which then influence the learned policy. However, these limitations can be addressed by similar algorithms. Specifically, the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm can be used to overcome these detriments [22].

Twin Delayed Deep Deterministic Policy Gradient (TD3)

TD3 extends the Deep Deterministic Policy Gradient (DDPG) method [31] with several key improvements to improve stability and performance which are catered towards reducing the impact of overestimation bias in the training process.

The algorithm maintains four neural networks:

- Actor network $\pi_\theta(s)$: Maps states to actions
- Target actor network $\pi_{\theta'}(s)$: Slowly updated copy of the actor
- Twin critic networks $Q_{\phi_1}(s,a)$ and $Q_{\phi_2}(s,a)$: TD3's twin critic architecture helps mitigate overestimation bias as we will see.
- Target critic networks $Q_{\phi'_1}(s,a)$ and $Q_{\phi'_2}(s,a)$: Slowly updated copies of the critic networks.

Aside from the introduction of the twin critics, there are two more innovations of TD3:

1. **Delayed Policy Updates**: The policy network is updated less frequently than the critic networks, which allows the policy to be more robust to unreliable estimates of Q-values.
2. **Smoothed Target Actions**: Noise is added to target action predictions to prevent overconfident exploitation of possibly erroneous Q-values. This is also known as *policy smoothing* as it prevents less peaked policies caused by peaked Q-values.

So, like in DDPG, critic networks are trained to minimize the Bellman error:

$$\mathcal{L}_Q = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[(y - Q_\phi(s,a))^2\right] \tag{4.21}$$

but now the target value y is the minimum of the twin target critics, which helps reduce the overestimation again:

$$y = r + \gamma \min_{i=1,2} Q_{\phi'_i}(s', \tilde{a}') \tag{4.22}$$

with $\tilde{a}' = \pi_{\theta'}(s') + \epsilon$ and ϵ representing target policy smoothing noise.

Then the actor network has its own loss with is attempting to maximize the expected Q-value:

$$\mathcal{L}_\pi = -\mathbb{E}_{s\sim\mathcal{D}}\left[Q_{\phi_1}(s, \pi_\theta(s))\right] \tag{4.23}$$

Additionally, the target networks in this case are updated through a process called Polyak averaging, meaning that they do still lag the actor and critic networks, but they are updated using a Polyak factor factor τ. Thus the parameters are updated as follows whenever the policy is updated:

$$\phi'_i = \tau\phi_i + (1-\tau)\phi'_i \quad \text{for} \quad i = 1,2 \tag{4.24}$$

$$\theta' = \tau\theta + (1-\tau)\theta' \tag{4.25}$$

RL Neural Network Learning in Context

The neural networks in TD3 learn through different mechanisms:

Critic Networks: Learn to estimate the value of state-action pairs by minimizing temporal difference errors. The networks develop internal representations that capture the relationship between the longitudinal bunch profile λ_t^i, parameter corrections (or actions), and resulting bunch profile λ_t^f.

Actor Network: Learns a deterministic policy mapping from bunch profiles to optimal parameter corrections. Through the policy gradient theorem, the network parameters are updated in directions that increase the expected cumulative reward.

As mentioned before, in vanilla RL, the critic and actor networks are based on classic MLP modules (from now referred to as dense layers), relying on the for feature extraction and computing the final action to be taken. This could potentially be augmented with the use of convolutional layers to benefit from the signal analysis capabilities of CNNs. However, as (very) recent research has shown, there is a fundamental disconnect between convolutional layers and the dense layers that ensue when setting up an RL agent to use convolutional blocks as a feature extractor [57]. This is mainly because the connection between the convolutional and dense layers generates a bottleneck that, if not properly tuned, generates dormant nodes in the dense layers, meaning they don't contribute to the final result nor the learning. The authors also propose a simple and effective solution of using a global average pooling (GAP) layer on the feature maps over the spatial dimension (which for the current case would be the profile bins), and then feeding them to the dense layers and this reactivates the dormant nodes [57]. However, at the time of making the models, this solution was not known and the standard dense layer approach is considered.

To summarize the training procedure, Algorithm 4 shows this for the TD3 agent. Note that in the "update step" criteria, it also implies that we wait until there are sufficient samples such that the update is meaningful (usually after 8000 transitions). Again, the approximate synchronous phase is added to the state and in this case the learning rate is kept constant over the training due to the fact that the learning is much more explorative and dependent on the Q-values than just back-propagating the gradients as was done in the CNN, so it makes no sense to reduce the learning rate when doing so can keep the actor in a local minima far away from the true optima. Note that in this case, the model would be comparable to Figure 4.9, but completely bypassing the feature extraction component and going straight to the dense layers.

4.5.2. Applicability and Advantages

The RL frameworks, in general, offer several advantages for accelerator optimization:

- **Sequential decision-making**: iterative optimization procedures to approach convergence, which gives the optimization tool the desired behaviour not provided by CNNs.
- **Exploration capability**: ability to discover different approaches which lead to the fastest and most efficient path to the desired outcomes. This becomes more important for the triple-harmonic optimizer as the parameter space is more complicated and the amount of features that change when one or various parameters are modified is much higher and less predictable than for second-harmonic operation.
- **Adaptability**: continuous learning and adaptation to changing conditions. Again this is not the case for double-harmonic as it depends on the labelled dataset, but it does become important for triple-harmonic as there is a figure of merit that allows for continuous adaptation and the possibility of continuing the learning process in the machine itself.
- **Safety considerations**: action space constraints can limit potentially harmful parameter changes in a way that is engrained in the model and it is not enforced, as will have to be done in the implementation of the CNN.
- **Convergence behaviour**: as the reward function can be modelled to our liking, there is an implicit modelling of the optimization trajectory that we would like the tool to take. As a result, convergence behaviour, including small adjustments when near the optimum, can be encouraged.

The safety aspect deserves particular emphasis. The continuous action space can be constrained to ensure that parameter adjustments remain within safe operational bounds, preventing large changes that could lead to beam loss or equipment damage. While the primary safety mechanisms rely on external fail-safe systems, the boundedness of the RL agent's actions provides an additional layer of protection.

4.5.3. Limitations

Despite these advantages, TD3 RL agents exhibit significant limitations:

- **No historical memory**: Each decision is based solely on the current state, ignoring valuable information from previous corrections. Unlike the CNN it can be modelled to possess convergence behaviour, but it still has no memory of the previous states, which could prove troublesome for generalizing to different impedances and accelerators.
- **Sample efficiency**: RL agents are famously very sample inefficient[52]. They require extensive training data to learn effective policies and explore different routes to the desired outcome.
- **Stability concerns**: Policy learning can be unstable, particularly in complex environments, and the hyperparameter sensitivity of the final performance of the agent is very high, even with the improvements

Algorithm 4 TD3 Training for Double and Triple-Harmonic Phase Optimization

Initialize actor network $\pi_\theta(s)$, critic networks $Q_{\phi_1}(s,a)$, $Q_{\phi_2}(s,a)$
Initialize target networks with same weights: $\theta' \leftarrow \theta$, $\phi'_1 \leftarrow \phi_1$, $\phi'_2 \leftarrow \phi_2$
Initialize Adam optimizer with constant learning rate
Initialize replay buffer $\mathcal{D}$
for episode = 1 to max_episodes **do**
    Reset accelerator simulation and get initial bunch profile $\mathbf{p}_0$
    Define initial state $s_0 \leftarrow (\mathbf{p}_0, \phi_{s_{approx}})$)
    for t = 0 to max_steps_per_episode **do**
        ▷ *Action Selection and Execution*
        Sample action from policy with exploration noise: $a_t \leftarrow \pi_\theta(s_t) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma)$
        Apply correction a_t to the simulated RF system
        Observe the new bunch profile $\mathbf{p}_{t+1}$ and calculate the reward r_t
        Define next state $s_{t+1} \leftarrow (\mathbf{p}_{t+1}, \phi_{s_{approx}})$
        Store transition (s_t, a_t, r_t, s_{t+1}) in replay buffer $\mathcal{D}$
        if update step **then** ▷ Perform updates every few steps
            ▷ *Sample and Compute Target*
            Sample a random minibatch of B transitions (s_j, a_j, r_j, s_{j+1}) from $\mathcal{D}$
            Select target action with clipped noise: $\tilde{a}_{j+1} \leftarrow \pi_{\theta'}(s_{j+1}) + \text{clip}(\epsilon', -c, c)$
            Compute the target Q-value using the clipped double-Q trick:
            $y_j \leftarrow r_j + \gamma \min_{i=1,2} Q_{\phi'_i}(s_{j+1}, \tilde{a}_{j+1})$
            ▷ *Critic Updates*
            Update critic networks by minimizing the MSE loss:
            $\mathcal{L}_{Q_i} \leftarrow \frac{1}{B} \sum_j^B (Q_{\phi_i}(s_j, a_j) - y_j)^2$
            $\phi_i \leftarrow \text{Optimizer.step}(\phi_i, \nabla_{\phi_i} \mathcal{L}_{Q_i})$ for $i = 1, 2$
            if policy update step **then** ▷ Delayed policy and target updates
                ▷ *Actor Update*
                Compute actor loss (policy gradient): $\mathcal{L}_\pi \leftarrow -\frac{1}{B} \sum_j^B Q_{\phi_1}(s_j, \pi_\theta(s_j))$
                $\theta \leftarrow \text{Optimizer.step}(\theta, \nabla_\theta \mathcal{L}_\pi)$
                ▷ *Target Network Updates (Polyak Averaging)*
                $\phi'_i \leftarrow \tau\phi_i + (1-\tau)\phi'_i$ for $i = 1, 2$
                $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$
            end if
        end if
    end for
end for

of TD3 over DDPG.

The lack of historical memory represents a critical limitation in accelerator applications. Human operators naturally consider the sequence of previous adjustments and their outcomes when making tuning decisions. This historical context provides crucial information about system behaviour, and implicitly handles varying impedances and convergence trends that pure Markovian policies cannot capture.

4.6. Memory-Augmented Reinforcement Learning

4.6.1. Theoretical Foundation

The limitations of memoryless RL agents motivate the integration of memory mechanisms that can capture and utilize historical information from previous optimization steps. Long Short-Term Memory (LSTM) networks provide a natural solution by maintaining internal state representations that evolve over the course of an optimization episode [36]. It was initially formulated to deal with Partially Observable Markov Decision Processes (POMDP) where sensors are required for state formulation and they can break or malfunction. In the current case, given that there are some incomplete parameter groups as explained in Section 4.1.4, this will help when the environment finds itself in one of these unmatchable situations as it will yield a bunch profile that is identically 0 everywhere. This is the partial observability behaviour that we are dealing with in the current case.

LSTM Architecture

The core of the LSTM architecture is its cell, which maintains a *cell state* C_t to act as a long-term memory and a *hidden state* h_t as a working, short-term memory. The flow of information is controlled by three primary

gating mechanisms: the forget gate, the input gate and the output gate.

The first step is to determine which information from the previous cell state C_{t-1} should be discarded. This decision is made by the forget gate, a sigmoid layer that takes the previous hidden state h_{t-1} and the current input x_t as inputs. It outputs a number between 0 and 1 for each element in the cell state, where 1 means "keep this" and 0 means "forget this".

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{4.26}$$

Next, the network decides what new information to store in the cell state. This process has two parts. First, an input gate layer (a sigmoid function) decides which values will be updated. Second, a tanh layer creates a vector of new candidate values, $\tilde{C}_t$, that could be added to the state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4.27}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{4.28}$$

The old cell state C_{t-1} is updated into the new cell state C_t. The previous state is multiplied element-wise by the forget gate vector f_t, discarding the selected information. Then, the new candidate values $\tilde{C}_t$ are scaled by the input gate vector i_t and added to the result. This additive interaction is important for mitigating the vanishing gradient problem that RNNs .

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{4.29}$$

Finally, the network determines its output, which is a filtered version of the cell state. The output gate, a sigmoid layer, decides which parts of the cell state will be included in the output. The cell state is passed through a tanh function to squash the values between -1 and 1, and this is then multiplied by the output of the sigmoid gate to produce the new hidden state h_t.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{4.30}$$

$$h_t = o_t \odot \tanh(C_t) \tag{4.31}$$

In these equations, $[h_{t-1}, x_t]$ denotes the concatenation of the two vectors, W_f, W_i, W_C, W_o are the weight matrices, and b_f, b_i, b_C, b_o are the bias vectors for each respective layer. As before, the symbol $\odot$ represents element-wise multiplication.

LSTM-TD3 Integration

The integration of LSTM memory with TD3 creates a hybrid architecture where the policy and value functions have access to historical information. The LSTM processes sequences of states and actions, maintaining an internal representation of the optimization trajectory, much like a human would.

The modified actor network becomes:

$$a_t = \pi_\theta(s_t, h_{t-1}) \tag{4.32}$$

where h_{t-1} represents the LSTM hidden state encoding historical information. Similarly, the critic networks also incorporate memory:

$$Q_\phi(s_t, a_t, h_{t-1}) \tag{4.33}$$

Learning with Memory

The learning process in LSTM-TD3 requires careful handling of temporal dependencies. The replay buffer stores sequences rather than individual transitions, and training involves processing these sequences to maintain the temporal structure necessary for LSTM operation.

Using this historical information LSTM learns to identify patterns in the optimization trajectory (also referred to as the history buffer), such as:

- Convergence trends and oscillatory behaviour.
- Effectiveness of different parameter adjustment strategies and system response characteristics under various conditions, especially important for triple-harmonic operation.
- Correlation between consecutive adjustments and outcomes, with them being the rewards obtained or the actions derived.

As a result of this, the amount of information the LSTM has to work with, meaning the length of the history buffer h_{len}, becomes an important hyperparameter that is added on top of TD3 hyperparameters. Furthermore, the usage of historical information implies that, in the programming sense, the replay buffer has to have considerably more complex to be able to handle only taking as much information as there is available and considering the termination signal of a given episode, as we don't want to include information in the history that is coming from another episode. Additionally, this becomes even more intricate when considering the possibility of parallelizing environments to speed up data collection, and also the possibility of using a replay buffer different from the traditional First-In-First-Out (FIFO) buffer. This will be discussed next.

4.6.2. Replay Buffers with Memory

The traditional experience replay mechanism in deep reinforcement learning employs a First-In-First-Out (FIFO) buffer that stores transition tuples (s_t, a_t, r_t, s_{t+1}) and samples them uniformly during training. While this approach successfully breaks temporal correlations and enables stable learning, it treats all experiences equally, regardless of their learning potential. This uniform treatment becomes particularly limiting when dealing with sparse rewards or when certain transitions contain significantly more information about optimal behaviour than others, as we have when we want convergence behaviour [33].

Prioritized Experience Replay

Prioritized Experience Replay (PER) addresses this limitation by assigning importance weights to stored transitions based on their temporal difference (TD) error. The core insight is that transitions with larger TD errors represent situations where the agent's current value estimates are most inaccurate, and therefore offer the greatest learning potential. The sampling probability for transition i is defined as:

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha} \tag{4.34}$$

where p_i is the priority of transition i (typically the absolute TD error plus a small constant ϵ to ensure non-zero probability), and α determines how much prioritization is used, with $\alpha = 0$ corresponding to uniform sampling. To compensate for the bias introduced by non-uniform sampling, importance sampling weights are computed as:

$$w_i = \left(\frac{1}{N \cdot P(i)}\right)^\beta \tag{4.35}$$

where N is the buffer size and β controls the amount of importance sampling correction, typically annealed from an initial value to 1. By annealing, initially the weighting is less based on the sampling which makes sense given that the networks are still in the early learning stages. Later on, it gets more and more confident in the TD errors as more information is already in the buffers.

The efficient implementation of prioritized sampling relies on a sum tree data structure, a binary tree where leaf nodes store transition priorities and internal nodes store the sum of their children's priorities. This enables $O(\log n)$ sampling and priority updates, making PER computationally feasible for large buffers [32].

Memory-Augmented Prioritized Replay

Extending PER to LSTM-based agents introduces more challenges. Rather than sampling individual transitions, the buffer must maintain and sample sequences of experiences while respecting episode boundaries. The complexity arises from several factors:

1. The notion of "history" in LSTM-TD3 requires extracting not just a single transition but a sequence of preceding transitions up to length h_{len}. This historical context enables the LSTM to build its hidden state representation, capturing temporal patterns that inform the current decision.
2. Episode boundaries must be carefully handled. When extracting history for a sampled transition, the sequence must not cross episode boundaries, as this would mix information from unrelated trajectories

and possibly different parameter combinations $(V_{total}, r_2, \dot{\mathcal{B}}, FF)$ in double-harmonic and $\dot{\mathcal{B}}$ in triple-harmonic. The implementation employs an iterative backward search from each sampled index:

$$\mathcal{H}_t = \{(s_{t-k}, a_{t-k}) : k \in [1, \min(h_{len}, t - t_{episode\_start})]\} \tag{4.36}$$

where $\mathcal{H}_t$ is the history buffer at step t and $t_{episode\_start}$ denotes the first step of the current episode.

While there are studies that implement PER buffers with LSTM-TD3 agents (see [32]), they do not provide their code. Also, the code provided by the original LSTM-TD3 paper (see [36]) is considerably unoptimised (mainly due to the logging mechanism of keeping all details in memory) and made training very slow, besides only using FIFO buffers. To top all of it off, there was no compatibility with parallelised environments which would speed up data collection, so this implied taking the original LSTM-TD3 code from the authors, optimising it, adding PER buffers and also adding compatibility for parallelisation.

Parallelized Environment Considerations

As mentioned, modern reinforcement learning often employs multiple parallel environments to accelerate data collection. In the current case, in order for the parallelisation to benefit data collection, each environment has to have a different shuffling order of the parameters used ($(V_{total}, r_2, \dot{\mathcal{B}}, FF)$ for double-harmonic and $\dot{\mathcal{B}}$ for triple-harmonic), otherwise we would just be repeating the same order across all environments and it would mean repeating almost the same corrections (with different noise) across many environments. This is handled by making each environment have its own Random Number Generator (RNG) using a seed dependent on the process ID.

This parallelisation introduces even more complexity to the replay buffer design. Two primary architectures emerge:

The **Separate Buffer Architecture** maintains independent prioritized replay buffers for each environment with each buffer providing a fixed amount of samples to the complete buffer. This approach ensures that experiences from different environments remain isolated, preventing potential interference when adding samples to the buffers and also when sampling history buffers. However, it may lead to imbalanced sampling if certain environments generate more informative experiences than others because the TD errors may be higher in some environments than others, but they all produce the same amount of samples for the complete buffer.

The **Unified Buffer Architecture** employs a single prioritized replay buffer shared across all environments, with an additional integer environment identifier stored with each transition. This approach provides several advantages:

- Global prioritization ensures that the most informative experiences are sampled regardless of their source environment.
- Better memory utilization as the buffer capacity is not pre-divided among environments, which reduces the overhead in the amount of memory taken up by the objects themselves.
- Simplified priority management with a single sum tree structure rather than individual sum trees.

The unified architecture's history extraction mechanism employs an iterative search that skips experiences from different environments while continuing to look backward for valid history and the algorithm is shown in Algorithm 5. This iterative approach maximizes history utilization by continuing the search beyond experiences from other environments, resulting in more complete historical context for the LSTM networks. Nevertheless, it does imply more computational effort to iteratively look for relevant samples to add to the history buffer.

Another important consideration that must be made with parallelisation is the modification of the *replay ratio* r_{replay}, which is defined as the number of gradient steps done for each transition collected in the replay buffer. It can be seen as how many times a sample in the replay buffer is used to update the networks before it is kicked out of the buffer because new transitions enter. So when it is very high, the RL agent becomes more sample efficient, but it is focusing a lot on reusing the samples in the replay buffer and it can lead to overfitting and vice versa. With PER this equation changes a bit, as we remove samples from the buffer that have the lowest priority, so it could be that a sample stays there for longer if it still represents a lot of information to the learning in the critic networks. However, it still represents the same thing on average, only that some transitions might stay for longer and others for a shorter number of updates.

As a result it is a very important value in the training that can lead to completely different results. To mitigate this, we set a certain replay ratio and use it as a hyperparameter of the training, and then calculate the amount of gradient updates that should be performed given that replay buffer, the number of parallelized environments, and the batch size B of the updates (which is another hyperparameter as it changes the amount of samples drawn from the buffer):

Algorithm 5 Iterative History Extraction for Parallel Environments

Input: Sampling index in the buffer idx, environment ID n, maximum history length h_{len}
Initialize history buffer filled with zeros $\mathcal{H} \leftarrow zeros(h_{len}, dim(s))$
Set search index $j \leftarrow idx - 1$
while $|\mathcal{H}| < h_{len}$ and $j \geq 0$ **do**
    if $env\_id\_buffer[j] = n$ **then** ▷ Only consider the transitions within one environment
        if $done\_buffer[j] = 1$ **then**
            break ▷ Episode boundary reached
        end if
        Append j to $\mathcal{H}$
    end if
    $j \leftarrow j - 1$ ▷ Keep going backwards in the buffer
end while
Return: Reversed $\mathcal{H}$ for chronological order

$$\text{Number of gradient steps / updates} = \frac{n_{\text{env}} \times r_{replay}}{B} \tag{4.37}$$

In practice we add this number of gradient steps to a "update credit" counter, which is considering how many updates should be performed in that given step according to Equation 4.37. Then in the update itself, we round down this update credit and perform that amount of gradient updates, subtracting the amount of times we update (which is necessarily an integer) from the update credit. Like this we are sure to always update only the amount of times indicated by the combination of the replay ratio r_{replay}, the batch size B and the constant amount of n_{env} for a given training (which depends on the availability of cores/threads in the computer or node where we are running the training).

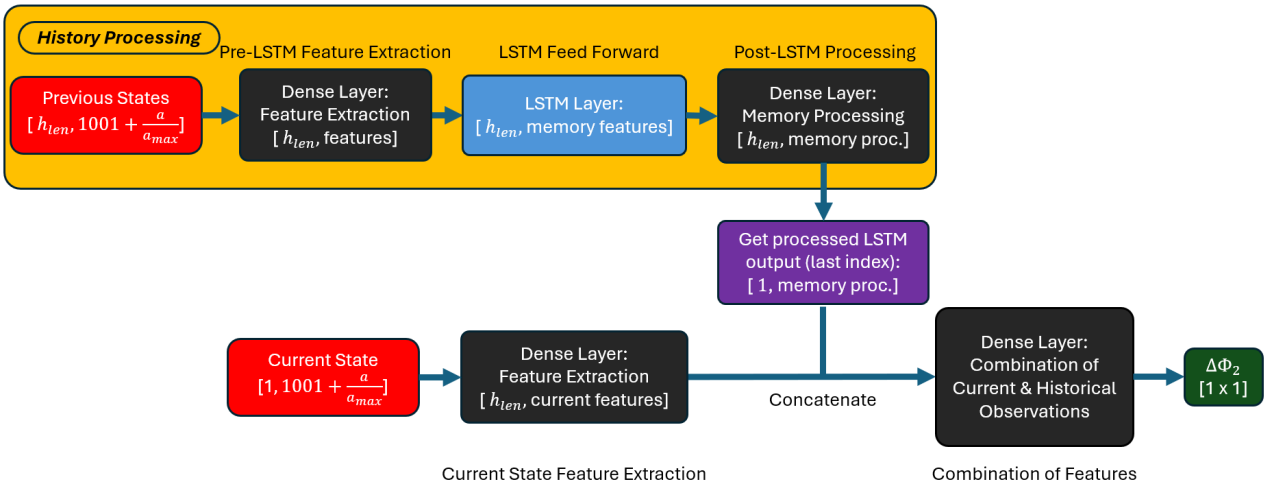


Figure 4.10: LSTM-TD3 model forward pass

The training procedure is now considerably modified from the TD3 variant with the introduction of the LSTM network, the history buffers $\mathcal{H}$, the PER buffer and the parallelisation, and the general procedure is shown in Algorithm 6, while the graphical depiction of the forward pass for the model is shown in Figure 4.10. Note that the action is now contained within the state to give even more context, but it is normalised by the maximum action, a_{max}, such that the network always receives a value in the range $[-1, 1]$. This would help it to generalise to even bigger actions in the case that this is desired.

Algorithm 6 LSTM variant of TD3 with Prioritized Experience Replay (PER)

Replace actor/critics with LSTM variants: $\pi_\theta(s, \mathcal{H})$, $Q_{\phi_i}(s, a, \mathcal{H})$; clone targets as in TD3
Initialize PER buffer $\mathcal{D}$ with $(\alpha, \beta_0) \rightarrow (0.6, 0.4)$)
Initialize n_{env} environments with their RNGs and history buffers $\mathcal{H}^{(n)} \leftarrow zeros(h_{len}, dim(s))$
Initialize update credit $\leftarrow 0$
for episode = 1 to max_episodes **do**
  Reset environments and define initial states $s_0^{(n)} \leftarrow (\mathbf{p}_0, \phi_{s_approx}))$
  for t = 0 to max_steps_per_episode **do** ▷ *Data collection (parallel)*
    for each env n in parallel **do**
      ▷ *Sample/Compute Action and Apply*
      $a_t^{(n)} \leftarrow \begin{cases} \text{random}, & t < startup\_steps \\ \pi_\theta(s_t^{(n)}, \mathcal{H}^{(n)}) + \mathcal{N}(0, \sigma^2), & \text{otherwise} \end{cases}$
      Execute action $a_t^{(n)}$, observe the reward $r_t^{(n)}$, get the next state $s_{t+1}^{(n)}$
      ▷ *Store Transition with Max Priority*
      Store the transition in the buffer (with maximum priority P_{max} so it is used at least once)
      ▷ *Update History Buffer*
      Update histories $\mathcal{H}^{(n)} \leftarrow$ append $(s_t^{(n)}, a_t^{(n)})$
      if max steps reached **then**
        Reset history $\mathcal{H}^{(n)} \leftarrow zeros(h_{len}, dim(s))$
      end if
    end for
    if $t_{total} \geq$ update_after **then**
      Add update credit $+= \frac{n_{\text{env}} \times r_{replay}}{B}$
      number of updates $\leftarrow int$(update credit)
      for $k = 1$ to number of updates **do**
        ▷ *Anneal β*
        Update $\beta \leftarrow \beta(t_{total})$
        ▷ *Sample from $\mathcal{D}$ using priorities $j \sim P(j) \propto p_j^\alpha$*
        Sample batch of B transitions with histories $(s_j, \mathcal{H}_j, a_j, r_j, s_{j+1}, \mathcal{H}_{j+1})$ from $\mathcal{D}$:
        ▷ *Compute Update Weights*
        Compute update weights per sequence j: $w_j \leftarrow \left(\frac{1}{N} \frac{1}{P(j)}\right)^\beta$, normalize by max
        ▷ *Compute Target and TD errors*
        Target smoothing: $\tilde{a}_{j+1}{=}\pi_{\theta'}(s_{j+1}, \mathcal{H}_{j+1})) + \mathcal{N}(0, \sigma^2_{\text{targ}})$
        Choose the minimum of the target twin critics $y_j{=}r_j + \gamma \min_{i=1,2} Q_{\phi_i'}(s_{j+1}, \mathcal{H}_{j+1}, \tilde{a}_{j+1},)$
        Compute the TD error for each critic $\delta_{j,i} \leftarrow |Q_{\phi_i}(s_j, \mathcal{H}_j, a_j){-}y_j|$ for $i = 1, 2$
        ▷ *Critic Updates*
        Update critic networks by minimizing the weighted MSE loss:
        $\mathcal{L}_{Q_i}{=}\frac{1}{B} \sum_j^B w_j \cdot (\delta_{j,i})^2$ for $i = 1, 2$
        $\phi_i \leftarrow$ Optimizer.step$(\phi_i, \nabla_{\phi_i} \mathcal{L}_{Q_i})$ for $i = 1, 2$
        ▷ *Priority Updates with small ϵ_{pri} so $p_j > 0$*
        Update PER priorities per sequence: $p_j \leftarrow \max|\delta_{j,i}|{+}\epsilon_{\text{pri}}$ for $i = 1, 2$
        if policy update step **then** ▷ Delayed policy and target updates
          ▷ *Actor Update*
          Compute actor loss (policy gradient): $\mathcal{L}_\pi \leftarrow -\frac{1}{B} \sum_j^B Q_{\phi_1}(s_j, \mathcal{H}_j, \pi_\theta(s_j, \mathcal{H}_j))$
          $\theta \leftarrow$ Optimizer.step$(\theta, \nabla_\theta \mathcal{L}_\pi)$
          ▷ *Target Network Updates (Polyak Averaging)*
          $\phi_i' \leftarrow \tau\phi_i + (1 - \tau)\phi_i'$ for $i = 1, 2$
          $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$
        end if
      end for
      ▷ *Remove the amount of updates from the credit*
      update credit $-=$ number of updates
    end if
  end for
end for

Another element introduced is the random sampling of the action space for a given number of startup_steps such that random data is used in the buffers before updating. As a result, startup_steps < update_after. Finally, there is also saving logic and a testing sequence on a separate test environment with its own RNG and rendering features showing the profile, the action taken, the error in phase and the reward at a given step (it renders to an animation) to be able to monitor and confirm the progress of the learning, but it is not included in Algorithm 6.

4.6.3. Applicability and Advantages

Memory-augmented RL provides several critical advantages:

- **Historical context**: Incorporation of previous corrections and their outcomes in decision-making.
- **Trajectory optimization**: Ability to plan sequences of adjustments rather than individual steps allowing for more complex strategies.
- **Convergence acceleration**: Learning from patterns in successful optimization sequences.
- **Robustness**: Better handling of noisy or inconsistent system responses, which could very well occur when there are inconsistencies in the data acquisition.

The historical context capability addresses the primary limitation of memoryless approaches. By maintaining information about previous states, actions, and rewards, the agent can make more informed decisions that consider the broader optimisation context and generalise to other accelerators and impedances which can shift the phase further from the ideal value used in the dataset.

4.6.4. Computational Considerations

While LSTM-TD3 provides enhanced capabilities, it introduces additional computational complexity:

- **Memory overhead**: Storage of hidden states and longer sequence buffers.
- **Training complexity**: Sequential processing requirements for LSTM components.
- **Inference time**: Additional computation for memory state updates.

Despite these considerations, the inference time remains suitable for real-time accelerator applications, as the primary computational bottleneck lies in data acquisition and preprocessing to match the simulated training distribution.

4.7. Feature Extraction and Preprocessing

4.7.1. CNN-based Feature Extraction

To improve learning efficiency and robustness, a variant of the LSTM-TD3 training incorporates control-relevant Convolutional Autoencoder (CAE) that extracts compact, physically meaningful features rather than using raw profiles directly. This has been proved to help in visual learning tasks, so the attempt is also made here, with the possibility of also validating the CNN approach by visualizing if profiles can be clustered by the necessary $\Delta\Phi_2$ phase correction[7]. The CAE architecture employs hierarchical convolutional layers with progressively larger receptive fields to capture multi-scale beam characteristics—from local peak shapes to global profile width and shape. The architecture is summarized in Table 4.3, but it is not fundamental as it just contains many learnable kernels and pooling layers to capture different features of the profile.

Table 4.3: Control-Relevant Autoencoder Architecture Details

Layer	Channels	Kernels	Pooling	Information Scale
		Encoder (Profile Processing)		
Conv1	$1 \rightarrow 64$	[3, 5, 7]	MaxPool(2)	Fine details: peak shapes, local variations
Conv2	$64 \rightarrow 128$	[9, 13, 17]	MaxPool(2)	Regional patterns: peak clusters, segments
Conv3	$128 \rightarrow 256$	[19, 25, 31]	MaxPool(2)	Global structures: symmetry, envelope

Continued on next page

Table 4.3 – continued from previous page

Layer	Channels	Kernels	Pooling	Information Scale
Conv4	256 → 512	[35, 45, 55]	None	Full context: long-range dependencies
		Context Encoding		
ϕ_s Encoder	1 → 32	—	—	Phase sensitivity information
		Feature Fusion		
Fusion Layer	544 → 512	—	—	Combined profile + ϕ_s features
Projection	512 → 32/64	—	—	Control-relevant latent space
		Decoder (Reconstruction)		
FC Layers	32/64 → 512	—	—	Latent expansion
DeConv1	512 → 256	7	Upsample(2)	Coarse reconstruction
DeConv2	256 → 128	7	Upsample(2)	Medium details
DeConv3	128 → 64	7	Upsample(2)	Fine details
DeConv4	64 → 1	7	None	Final reconstruction
Output	1 → 1	-	—	Positive beam profile (Softplus)
		Auxiliary Head (Phase Prediction)		
MLP	32/64 → 2	—	—	Predicts [cos($\Delta\phi$), sin($\Delta\phi$)]

The training follows a *transfer learning* approach, meaning that it is first trained on one task and then it moves on to the next. Thus, the training is performed in two phases:

1. **Initial training (epochs 0-99)**: The model optimizes a combined loss function:
   - Reconstruction loss to preserve profile information.
   - Control relevance loss with Variance-Invariance-Covariance (VICReg) regularization that clusters profiles requiring similar phase corrections in the latent space, using the cosine similarity of phase errors (similar to how the cosine loss was implemented for the CNN) [11].
2. **Phase-aware refinement (epochs 100+)**: After the initial reward plateaus, an auxiliary phase head is introduced that predicts signed phase errors as 2D unit vectors $[\cos(\Delta\phi), \sin(\Delta\phi)]$ from the latent representation. This additional supervision helps the encoder identify control-relevant features more precisely, albeit with the possibility of losing precision in the reconstruction task if it finds an avenue for steeper loss reduction through the phase-aware refinement.

This approach yields a latent representation organized by "what correction is needed" rather than visual similarity, providing the RL agent with features that are compact (reduction from 1001 to 32-64 dimensions), and control-relevant as the features are forced to relate to required phase corrections. Furthermore, two possibilities arise for how the CAE can be used in the training of the RL agent:

1. **Reduce Dimensionality of RL Input**: Use the CAE to encode raw profiles into a latent space which has richer features for the RL agent to use and converge quickly to the ideal corrections.
2. **Augment the Raw Profiles**: Use the CAE to encode the raw profiles that are not augmented by any means, and then decode them again to feed into the RL agent. By doing this, one would assume that because the CAE has been conditioned into looking at the most relevant features for phase correction, that the profiles that would be fed into the RL agent are more refined in the portions that contain information relevant for corrections, while being noisier or simply 0 in the portions where there is no phase information. Additionally, when applied in the PSB, this might yield better results as the profiles would be reconstructed to profiles that the RL has seen before, which might not be the case when using preprocessed WCM signals. However, this also depends on the reconstruction quality of the CAE, which might deteriorate after the phase-aware loss comes into play (epochs 100+).

Both of these approaches will be considered for the training to see how much performance can be increased.

On a final note, the same type of stratified sampling described in Section 4.4 is employed in the training of the CAE.

4.8. Hyperparameter Optimization

To maximize model performance, Optuna, a Bayesian optimization framework, is used to systematically explore the hyperparameter space. This framework was chosen due to its simplicity and performance in both accuracy and computation time, offering a good compromise between them in comparison to other highly similar frameworks like HyperOpt[26]. This is a sample-efficient way to map out the validation loss (for the CNN) or mean training reward (for the RL agent), which is used as the value to minimize, as a function of the hyperparameters. This means that the hyperparameter optimization **will only be performed for the double-harmonic optimizers**. This is because only RL is considered for triple-harmonic optimization and the main driving hyperparameters parameters are expected to be the training parameters as the quality of training is heavily influenced by them. Thus, the same parameters or similar ones with slight modifications should be fit for the triple-harmonic RL agent as well.

The validation loss is used because the generalisation of the model to untrained datasets is desirable, and given how well defined the correct second harmonic phase is (provided the voltage and voltage ratio is high enough) and the convolutional nature of the model, there is a high chance that the geometric understanding of the profiles can be extrapolated to unseen scenarios. This is essentially required as operators are still able to correctly phase a profile even when new voltage or momentum programmes. Meanwhile, the mean training reward is used because it is representative of the reward across several parameter combinations due to the parallelisation and the independent RNGs of each environment. However, a small change is made to the objective value for the RL agents given the noisiness of the reward curves, which can be very large. This change is to use the following objective function for the RL agents, where y is the objective value:

$$y = mean(r_{\text{final epoch}}) - std(r_{\text{final epoch}}) \tag{4.38}$$

where r is referring to the episode returns for each of the n_{env} the parallelized environments, and std refers to the standard deviation of those returns (σ). This is also because as the model gets better it should be better across all parameter combinations, and not the majority. So if it does not manage to perform at least somewhat well on all parameter combinations, the standard deviation should be very large.

4.8.1. Optimization Procedure

The hyperparameter optimization process can be formalized as finding the set of hyperparameters λ that minimizes some loss $\mathcal{L}_{\text{training}}$, which in this case is either the validation loss or the mean training loss:

$$\lambda^* = \underset{\lambda}{\operatorname{argmin}}\, \mathcal{L}_{\text{training}}(f_\lambda(X)) \tag{4.39}$$

where f_λ is the model configured with hyperparameters λ, and X is the data used to train each model.

Optuna employs Tree-structured Parzen Estimators (TPE) to efficiently explore the hyperparameter space [60]. The TPE algorithm models the conditional probability of hyperparameters given the observed objective value y:

$$p(\lambda|y) = \begin{cases} \ell(\lambda) & \text{if } y < y^* \\ g(\lambda) & \text{if } y \geq y^* \end{cases} \tag{4.40}$$

where y^* is a threshold value that is automatically computed from the samples taken, $\ell(\lambda)$ is the probability density of hyperparameters that yield good performance, and $g(\lambda)$ is the density of hyperparameters that yield poor performance. For each trial, Optuna selects the hyperparameters that maximize the expected improvement given the TPE:

$$\lambda_{\text{next}} = \underset{\lambda}{\operatorname{argmax}}\, \frac{\ell(\lambda)}{g(\lambda)} \tag{4.41}$$

Thus, it is expected that as the number of times that the objective value is probed increases, the better the models are of the probability densities $l(\lambda)$ and $g(\lambda)$.

4.8.2. Functional Analysis of Variance (fANOVA)

Functional ANOVA (fANOVA) is a variance decomposition technique that quantifies the importance of hyperparameters by analysing their contribution to the variance of the objective function. Unlike traditional ANOVA, fANOVA leverages Random Forests to handle both continuous and categorical variables while capturing non-linear relationships and interactions, while ANOVA can only capture linear dependencies with variables [23].

Random Forests in fANOVA

Random Forests are ensemble learning methods that construct multiple decision trees and aggregate their predictions. Each tree T_b in the forest is trained on a random sample of the data, and at each split in the decision trees, only a random subset of features is considered, which in this case are the hyperparameters. At the output level, the predictions of all trees are averaged and the Random Forest is trained to minimize the MSE loss from the observed data, which in this case is the objective value. Essentially, it is building a function estimator through the use of Random Forests, but the beauty lies in that it can exclude hyperparameters once the model is trained, allowing interactions to be assessed and the individual variance contribution of a certain parameter to be ascertained. This approach provides several advantages for fANOVA:

- **Non-linearity**: Trees naturally capture non-linear relationships through their hierarchical splitting structure.
- **Mixed variables**: Can handle both continuous and categorical hyperparameters without preprocessing.
- **Interactions**: Automatically model hyperparameter interactions through the tree structure.
- **Robustness**: Ensemble averaging reduces overfitting, crucial for small sample sizes.

The Random Forest approximates the objective function as:

$$\hat{f}(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} T_b(\mathbf{x}) \tag{4.42}$$

where B is the number of trees and $T_b(\mathbf{x})$ is the prediction of the b-th tree for hyperparameter configuration $\mathbf{x}$.

Mathematical Formulation

Given the hyperparameter space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_d$ and the Random Forest approximation $\hat{f}$, fANOVA performs a functional decomposition:

$$\hat{f}(\mathbf{x}) = f_0 + \sum_{i=1}^{d} f_i(x_i) + \sum_{i<j} f_{i,j}(x_i, x_j) + \cdots + f_{1,2,\ldots,d}(x_1, \ldots, x_d) \tag{4.43}$$

where:

- $f_0 = \mathbb{E}[\hat{f}(\mathbf{x})]$ is the mean prediction.
- $f_i(x_i) = \mathbb{E}[\hat{f}(\mathbf{x})|x_i] - f_0$ is the main effect of hyperparameter i.
- $f_{i,j}(x_i, x_j) = \mathbb{E}[\hat{f}(\mathbf{x})|x_i, x_j] - f_i(x_i) - f_j(x_j) - f_0$ is the interaction effect.

Efficient Computation via Tree Structure

The key insight is that these expectations can be computed efficiently using the tree structure. For a single tree T, the marginal prediction for a subset S of hyperparameters is:

$$\mathbb{E}[T(\mathbf{x})|\mathbf{x}_S] = \sum_{\ell \in \text{leaves}(T)} v_\ell \cdot P(\mathbf{x} \in \ell|\mathbf{x}_S) \tag{4.44}$$

where v_ℓ is the prediction value at leaf ℓ, and $P(\mathbf{x} \in \ell|\mathbf{x}_S)$ is the probability of reaching leaf ℓ given the values of hyperparameters in subset S. This essentially means that we can quickly compute the marginalization of a certain hyperparameter and see its contribution to the objective value.

Variance Decomposition and Importance

The total variance is decomposed as:

$$\mathrm{Var}(\hat{f}) = \sum_{S \subseteq \{1,\ldots,d\}} \mathrm{Var}(f_S) \tag{4.45}$$

The importance of hyperparameter i, including all its interaction effects, is achieved by averaging over the effect of all other variables (using the efficient marginalization described previously) and computing the variance associated to that variable. Thus, it can be written as:

$$I_i = \frac{\mathrm{Var}(\mathbb{E}[f_i|x_i])}{\mathrm{Var}(\hat{f})} \tag{4.46}$$

This quantifies the fraction of the objective function's variance that can be attributed to hyperparameter i and its interactions, which tell us how important its value is.

Practical Considerations

For hyperparameter optimization with limited trials (e.g., $n < 20$), the Random Forest parameters should be adjusted to reduce the number of trees ($B \approx 16 - 32$) to prevent overfitting, limit the tree depths (depth = 2-3) for better generalization and to use multiple random seeds to assess stability of importance estimates. Furthermore, this can be coupled with other estimates such as the Mean Decrease in Impurity (MDI) importance estimator to confirm the results of the analysis. MDI is not explained as it is out of the scope, but it uses a similar approach as fANOVA, just a different objective function to optimize the random forests.

The Random Forest-based approach makes fANOVA particularly suitable for hyperparameter analysis as it naturally handles the mixed continuous-categorical nature of hyperparameter spaces and captures the complex, non-linear relationships between hyperparameters and model performance.

4.9. Summary

This chapter has presented a comprehensive analysis of three ML frameworks suitable for accelerator parameter optimisation: CNN with attention mechanisms, TD3 reinforcement learning, and memory-augmented LSTM-TD3. Each approach addresses different aspects of the optimisation challenge, with a clear progression from static pattern recognition to dynamic policy learning with historical context. The dataset creation methodology, constrained by practical limitations of accelerator operations, fundamentally shaped the choice for these ML approaches. The adoption of static profile datasets eliminates temporal methods while enabling the development of robust optimisation policies through simulation-based training.

The progression from CNN to TD3 to LSTM-TD3 represents a natural evolution driven by the requirements of RF system optimisation presented in Chapter 3. The frameworks presented in this chapter form the foundation for addressing both the 1D phase optimisation problem in double-harmonic operation and the more complex 5D parameter space control required for triple-harmonic systems. The theoretical understanding developed here enables the analysis, evaluation and comparison of these approaches in the context of their training, simulated PSB cycles and experiments in the PSB itself, as detailed in the verification and validation procedures in Table 3.2.

5

Double-Harmonic Operation: Φ_2 Optimisation

This chapter presents the development and evaluation of three distinct machine learning approaches to automate the second harmonic phase correction: a Convolutional Neural Network with attention mechanisms (CBAM) and a memory-augmented reinforcement learning agent (LSTM-TD3). Each approach offers unique advantages and faces specific challenges in learning the complex relationship between beam profiles and optimal phase settings, as detailed in Chapter 4. The CNN-based approach leverages spatial pattern recognition to directly map beam profiles to phase corrections, while the RL agents learn through interaction with the accelerator environment, potentially discovering better and more robust optimisation strategies.

We begin by examining the CNN with CBAM architecture, focusing on critical training considerations such as data normalisation strategies. Subsequently, we analyse the performance of the RL agents, comparing their ability to adapt to varying beam conditions and their robustness to operational uncertainties. Additionally, we apply the verification and validation principles outlined in Section 3.3.1 in order to evaluate the ML frameworks and their applicability.

5.1. CNN with Attention Mechanisms

5.1.1. Training Considerations

The training of convolutional neural networks for beam profile analysis presents unique challenges due to the nature of the longitudinal beam profiles and their temporal dependency which is not immediately modelled. This section examines critical aspects of the training procedure that significantly impact model performance.

Normalisation Strategy: Sum versus Maximum

A fundamental preprocessing decision involves the normalisation strategy for bunch profiles. Initially, sum normalisation appears attractive, as it preserves information about charge concentration. This is a key indicator of whether the system is phased in Bunch Shortening Mode (BSM) or Bunch Lengthening Mode (BLM), since the main goal here is to minimise the maximum longitudinal charge density. As illustrated in Figure 2.26, sum-normalised profiles reveal how charge clustering varies with different second harmonic phases, providing clear visual cues that ideally should guide the model toward homogenising the distribution. Proceeding with max normalisation eliminates this information when it is most critical, as seen in Figure 5.1, where the profiles matched at different phases show practically the same bunch profile with minor differences. This will make it difficult for the model to distinguish them.

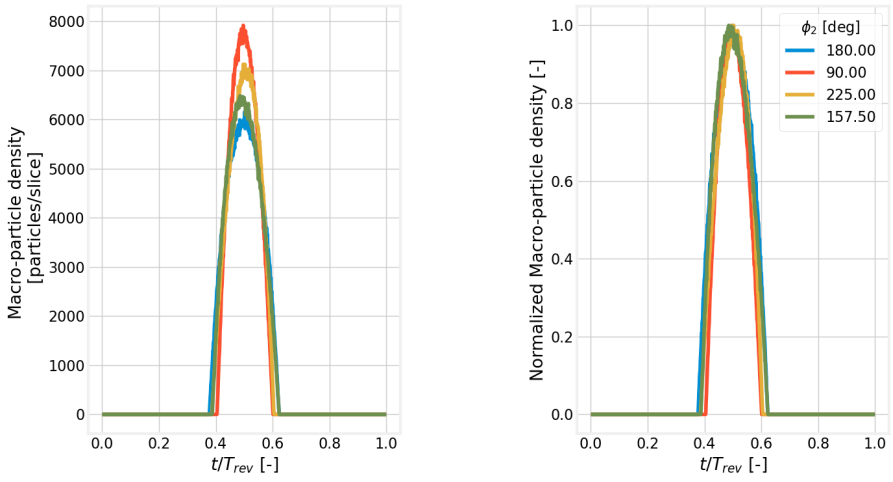


Figure 5.1: Comparison of beam profiles with different no normalisation (left) and max normalisation (right) at low total voltages, voltage ratios and emittances

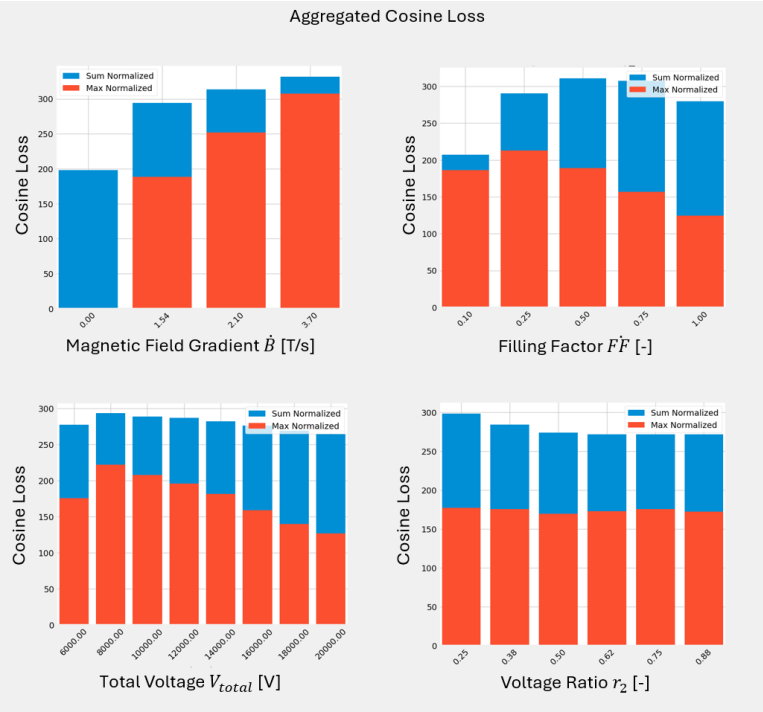


Figure 5.2: Aggregated cosine loss error for the reduced dataset using the CBAM model, comparing maximum and sum normalisation strategies across different beam parameters.

However, results using a reduced dataset without augmentation (no noise and no baseline droop) revealed a counter-intuitive outcome when training the CNN with CBAM attention mechanisms: sum normalisation consistently yielded higher prediction errors compared to maximum normalisation. Figure 5.2 demonstrates this performance gap by showing the aggregated cosine loss error across various parameters $V_{total}, r_2, \dot{\mathcal{B}}$ and FF. The aggregated cosine loss shows approximately 20 to 40% higher errors when using sum normalisation across all parameters. It is even more interesting to see that when there is no acceleration ($\dot{\mathcal{B}} = 0$), there are practically no errors for the max normalised model, while there is considerable error for the sum normalised model.

This unexpected behaviour stems from a fundamental characteristic of the CNN architecture. The convolutional layers naturally extract spatial features through their kernels, making them sensitive to relative patterns rather than absolute magnitudes. When sum normalization is applied, the varying charge density across different beam conditions introduces a scaling factor that interferes with pattern recognition. The model becomes confused between changes due to charge density variations and changes due to phase relationships. This interferes with updates to the network parameters as this additional factor negatively affects feature recognition.

Additionally, the approximate synchronous phase $\phi_{s_{approx}}$ in the profile location becomes more dominant with sum normalization. Since the beam's temporal position within the RF bucket correlates strongly with the optimal second harmonic phase as seen in Figure 2.23 where the higher ϕ_s, the larger the deviation from π required for optimal BLM phasing. As a result, the CNN learns to rely heavily on this positional information rather than on the distribution shape. This creates a fragile predictor that fails in the simplest of cases for $\dot{\mathcal{B}} = 0$ as $\phi_{s_{approx}} = 0$. This theory is supported by the plots seen in Figure 5.2.

The CNN's susceptibility to this issue relates to its hierarchical feature extraction process. Early convolutional layers detect edges and peaks in the profile, while deeper layers combine these into higher-level features. Under sum normalisation, the varying baseline levels caused by different heights of the profile create inconsistent edge responses as the kernels are learning parameters that are not generalisable across different heights, disrupting the feature hierarchy. Meanwhile, max normalisation effectively removes this confounding factor by ensuring that all profiles have the same peak height, forcing the model to focus on shape characteristics. Although this sacrifices some information about charge concentration, it creates a more robust feature space in which the CNN can reliably learn the relationship between the bunch shape and the optimal phase settings. This trade-off proves worthwhile in practice, as the normalised shape contains sufficient information for phase optimisation while avoiding the pitfalls of intensity-dependent features.

Furthermore, in anticipation of the MLP network structure of the RL agents, it becomes important to consider whether the sum normalisation can aid there, as it is incompatible with the convolutional layers but perhaps

not with MLP. However, since this test is performed before the stratified dataset split was incorporated, instead of only looking at the validation loss, it is more indicative of performance to take a weighted average of the training and validation losses, in the percentages they represent in the dataset, with a training/validation split of 85:15 in this case. This is done because the validation set could contain entire groups that are not seen by the model, so it could yield varying validation losses that indicate that a certain model is better, when it is entirely up to chance due to the randomised split of data.

The weighted loss is shown in Table 5.1 when training for 200 epochs (sufficient for both losses to have plateaued) on the reduced problem for the CNN with CBAM (now referred to as CBAM to distinguish it from no attention mechanisms), the CNN without attention mechanisms, and only the MLP-based phase computation head shown in Figure 4.9. The max normalized MLP model still performs better on both the validation and training dataset, and there is a higher percentage increase in the loss in this case compared to the others. This process was repeated in order to observe whether the results were consistent, and also repeated once the hyperparameter optimisation was performed. As will be discussed in the following sections, the results were consistent, and the same conclusion was reached.

As such, only max normalization is considered as reasoning and empirical data seem to support it as being the normalization technique yielding the highest performance. On a final note, even though it might seem that Table 5.1 is performing an ablation study, this is not the case because this is reliant on unoptimised network parameters, a reduced dataset as indicated in the horizontal axis labels in Figure 5.2, and a random split in the dataset (compare with Figure 4.4). For example, if we were to use this as the ablation study, it would indicate that because the validation loss is higher for the CNN than the MLP model, that there is no benefit to having convolutional layers, when this is not the case.

Table 5.1: Weighted average of training and validation cosine loss comparison for different models and normalization strategies (85/15 Train/Val Split, 200 Epochs)

Model	Norm.	Train Loss	Val Loss	Weighted Average	Δ (%)
CBAM	Max	1.03×10^{-3}	4.72×10^{-2}	7.95×10^{-3}	–
	Sum	7.84×10^{-4}	5.09×10^{-2}	8.30×10^{-3}	+4.40
CNN	Max	2.11×10^{-3}	8.73×10^{-2}	1.49×10^{-2}	–
	Sum	1.96×10^{-3}	8.92×10^{-2}	1.51×10^{-2}	+1.34
MLP	Max	5.62×10^{-3}	7.14×10^{-2}	1.55×10^{-2}	–
	Sum	6.72×10^{-3}	7.39×10^{-2}	1.68×10^{-2}	+8.39

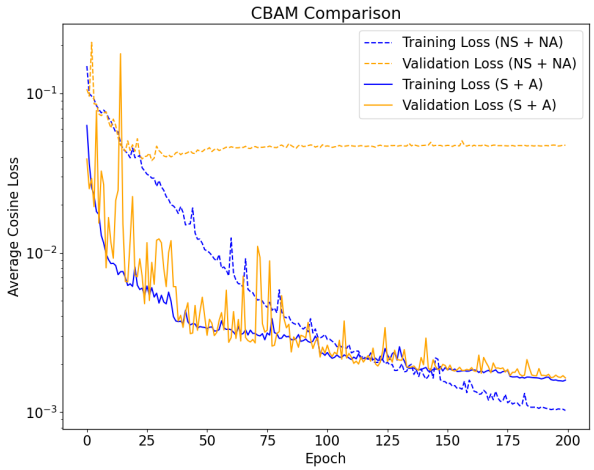
Effect of Stratified Sampling and Data Augmentation

The performance of neural networks critically depends on the quality and distribution of the training data. To investigate this dependency, we compare models trained on randomly split datasets against those using stratified sampling combined with data augmentation, implying a more difficult learning task to be invariant to the augmentation. The stratified approach ensures balanced representation across all operational parameters while the augmentation introduces realistic measurement artifacts. Figure 5.3 reveals this because all three architectures show significantly better validation losses when trained on the stratified dataset, despite the added difficulty from augmentation. This improvement manifests itself differently across architectures, providing insight into their learning characteristics.

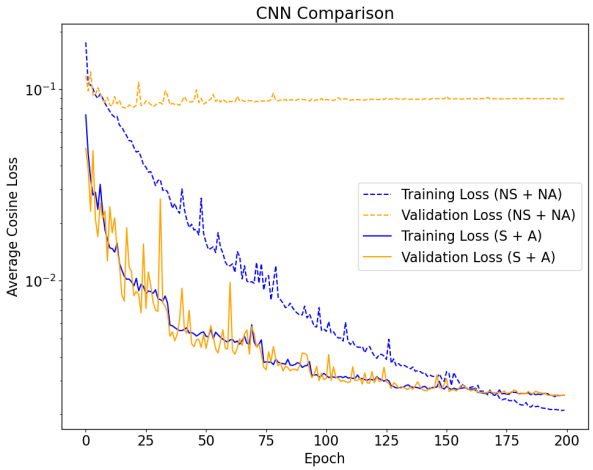
The CBAM model (Figure 5.3a) demonstrates the most pronounced benefit from stratified sampling. Although the non-stratified training achieves a lower final training loss (approximately 10^{-3}), this apparent advantage masks severe overfitting and is only due to the simpler task due to lack of augmentation. The validation loss remains stubbornly high around 5×10^{-2}, creating a generalization gap of nearly two orders of magnitude. In contrast, the stratified dataset yields more balanced learning curves, with the validation loss stabilizing near 1.6×10^{-3}, representing an improvement of over 90%. The higher training loss in the stratified case (2.8×10^{-3}) actually indicates better learning, as the model must generalize across augmented profiles with noise and baseline variations rather than memorizing clean examples.

The standard CNN (Figure 5.3b) exhibits similar behavior. The non-stratified validation loss plateaus around 8.9×10^{-2}, while stratified sampling reduces this to approximately 2.5×10^{-3}. Interestingly, the CNN shows more stable training dynamics than CBAM, with fewer oscillations in the loss curves. This stability suggests that the simpler architecture is less prone to overfitting, although it still benefits substantially from proper dataset preparation, and achieves a validation loss roughly 50% larger than that of the CBAM model.

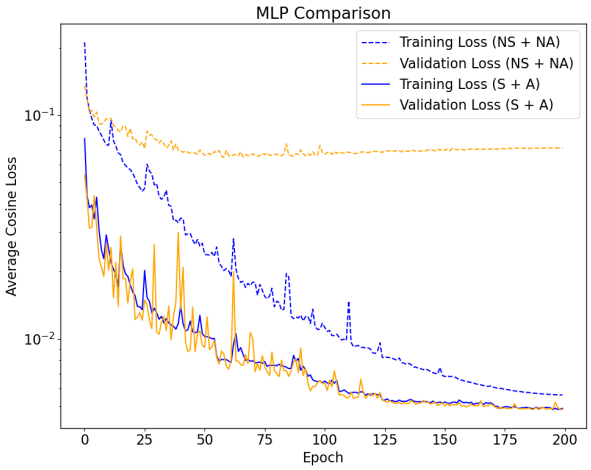
The MLP results (Figure 5.3c) provide an important baseline, showing that even without convolutional struc-



(a) CBAM



(b) CNN



(c) MLP

Figure 5.3: Training and validation loss evolution comparing non-stratified (NS) without augmentation (NA) versus stratified (S) with augmentation (A) datasets. All models use maximum normalization and were trained for 200 epochs with identical hyperparameters.

ture, stratified sampling improves generalization. The improvement in validation loss from 7.1×10^{-2} to 4.8×10^{-3} demonstrates that the benefits extend beyond spatial feature extraction. What is also interesting to see is that for the MLP model, the stratified dataset and the augmentation lead to quicker learning than without it, and a lower training loss even when considering the noise and baseline droop. This is an important result, as it indicates that by augmenting the dataset the model focuses on the more important aspects of the profiles, rather than "cheating" by using ideal profiles and using less relevant features that lead to the same correction. This result indicates that when training the RL models, the profiles should be augmented to lead to better performance.

Furthermore, when comparing the accuracy of each model, the loss must not be compared at face value because the proportionality between losses changes when the cosine is applied in the cosine loss. The difference in accuracy (in degrees) is lower than the difference implied by the losses of each model:

$$\mathcal{L}_{CBAM} = 1.6 \times 10^{-3} \rightarrow \Delta\Phi_{2_{error}} = 3.24^\circ \tag{5.1}$$

$$\mathcal{L}_{CNN} = 2.8 \times 10^{-3} \rightarrow \Delta\Phi_{2_{error}} = 3.97^\circ \tag{5.2}$$

$$\mathcal{L}_{MLP} = 4.8 \times 10^{-3} \rightarrow \Delta\Phi_{2_{error}} = 5.61^\circ \tag{5.3}$$

In summary, the superior performance of stratified sampling stems from its systematic approach to dataset construction. By ensuring equal representation across the five-dimensional parameter space (total voltage,

voltage ratio, magnetic field gradient, filling factor, and augmentation type), stratified sampling eliminates accidental biases that arise in random splits. For instance, a random split might place most high-voltage or low-filling-factor examples in either training or validation, creating distribution mismatches that impair generalization. This robustness is particularly important for the CBAM architecture, whose attention mechanisms might otherwise focus on noise-free features that do not exist in reality.

These results carry important implications for deployment. Models trained on non-stratified data may perform well on beam conditions similar to their training set but fail catastrophically when encountering underrepresented parameter combinations. The stratified approach, despite requiring more careful dataset preparation, ensures consistent performance across the full operational envelope of the PSB and also results in more reproducible training results as well. The additional computational cost of augmentation during training is negligible compared to the operational benefits of robust phase correction.

5.1.2. Autoencoder Analysis for Feature Extraction

Training and Auxiliary Phase Head

To explore alternative representations of bunch profiles for phase correction, we investigated a convolutional autoencoder (CAE) architecture with VICReg regularization. As mentioned previously, the approach aims to learn a compressed latent representation that captures the essential features determining optimal phase corrections or enabling data augmentation such that the RL can train on more consistent data from reconstructions rather than the raw profiles. Applying this to the PSB would technically have the benefit of converting the WCM-acquired bunch profiles into something that the RL has seen.

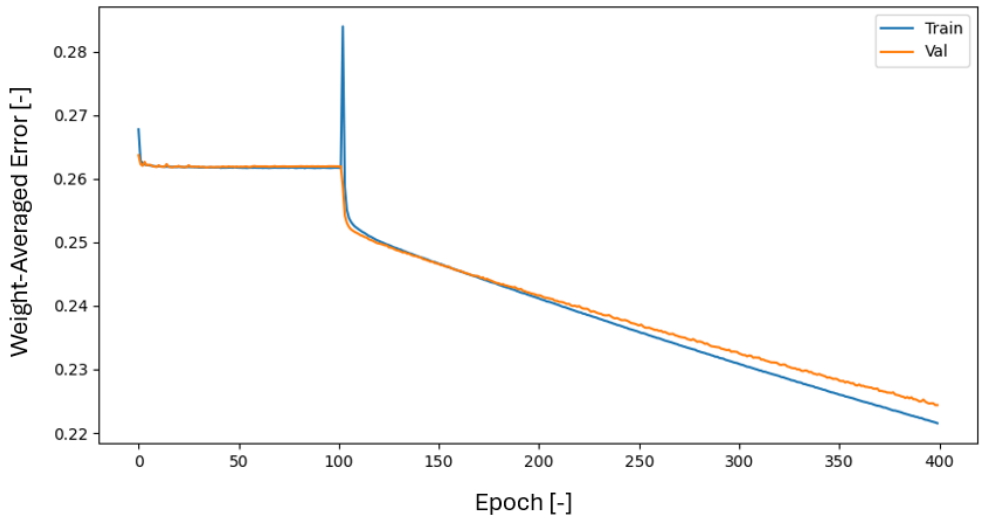


Figure 5.4: Training and validation loss evolution for the CAE with 64-dimensional latent space

Figure 5.4 shows the training progression of the CAE. With a 64-dimensional latent space, both training and validation losses plateau after approximately 40 epochs at a relatively high value of 0.26. This early saturation suggests the model struggles to compress the "1000-dimensional" beam profiles effectively into such a compact representation while maintaining reconstruction fidelity (as the loss is dominated by the reconstruction error).

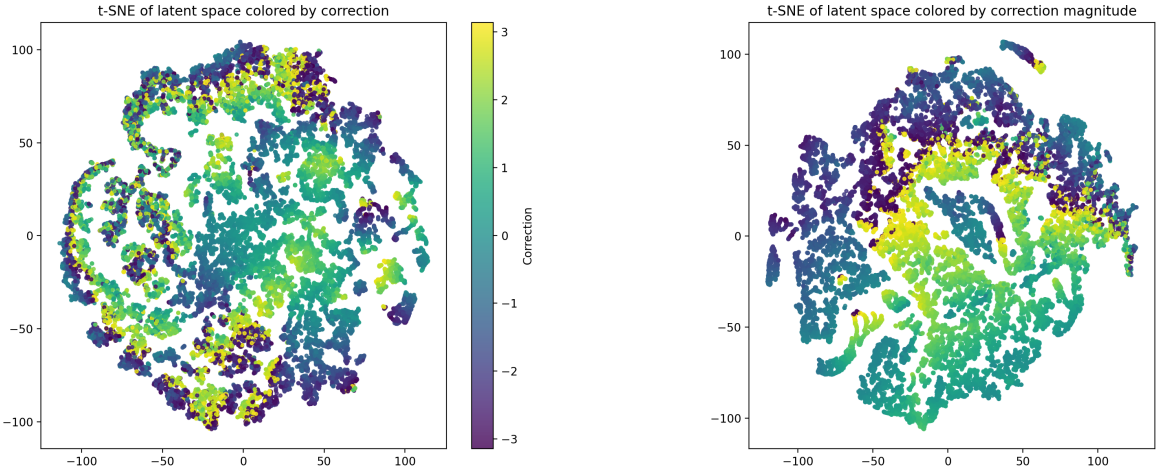
The introduction of an auxiliary phase prediction head at epoch 100 marks change in the training. This additional component predicts the optimal phase correction directly from the latent representation, adding a supervised signal to the otherwise unsupervised learning process. The combined loss function then becomes (whereas before $\lambda_{phase} = 0$):

$$\mathcal{L}_{total} = (\mathcal{L}_{reconstruction} + \lambda_{VICReg}\mathcal{L}_{VICReg} + \lambda_{phase}\mathcal{L}_{phase})/(1 + \lambda_{VICReg} + \lambda_{phase}) \tag{5.4}$$

where $\mathcal{L}_{phase}$ measures the cosine distance between predicted and true phase corrections. This multi-tasking learning approach provides dual benefits: it directly optimises the latent space for phase prediction relevance, and it helps overcome the clustering constraints imposed by VICReg regularization that may not align naturally with the phase correction task. The resumed decrease in loss after epoch 100 validates this hypothesis, especially since the loss is divided by the weighting of the individual loss components, suggesting the phase supervision guides the latent space toward more meaningful phase-correction representations. Also, since the losses did not plateau, it would have been ideal to keep training the CAE, but by this time it was already training for a week on a NVIDIA H100 GPU (80 GB of VRAM, and high transfer speeds), one of the most capable GPUs in the market. Thus, due to time and computational constraints, the training ended at this point.

To validate that the learned representations capture phase-relevant information, we employed t-distributed

Stochastic Neighbours Embedding (t-SNE) to visualise the 64-dimensional latent space. t-SNE is a non-linear dimensionality reduction technique that preserves local neighbourhood structures by minimising the KL divergence between probability distributions in high and low-dimensional spaces (for exact details see [59], here only the visualisation component of the technique is relevant). The algorithm constructs probability distributions over pairs of samples, where similar samples have high probability of being picked as neighbours, then finds a low-dimensional embedding that preserves these relationships (in this case compressing the 64 dimensions into 2 to show clustering).



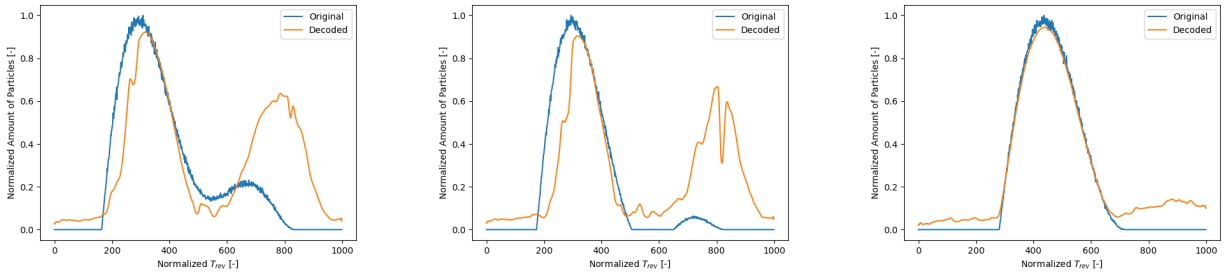
(a) Before auxiliary phase head conditioning **(b)** Final plot after conditioning

Figure 5.5: t-SNE visualisation of the CAE's latent space for 50,000 bunch profiles before and after auxiliary phase conditioning.

Figure 5.5 shows the structure in the learned representations. The learnt clustering improves considerably under the auxiliary phase conditioning, which is evident when comparing the t-SNE distribution before and after conditioning in Figure 5.5a and Figure 5.5b, respectively. The latent space exhibits clear clustering based on optimal phase corrections, with smooth colour gradients indicating continuous transitions between different correction values. Particularly noteworthy is the sharp boundary between π and $-\pi$ corrections (the yellow-to-purple transition in Figure 5.5b). This discontinuity is not an artifact but rather physically meaningful: corrections of $+\pi$ and $-\pi$ are equivalent, as they both represent a 180° phase shift. The CAE correctly learns this periodicity without explicit instruction, providing strong evidence that convolutional architectures can extract phase-relevant features from beam profiles.

Reconstruction Quality and Implications

Despite the encouraging latent space organization, the CAE's reconstruction capability proves severely limited, as was suggested by the stagnated training and validation loss in the reconstruction-dominated portion of the training. Figure 5.6 shows representative examples across different profile types, revealing reconstruction failures.



(a) Double-peaked profile **(b)** Separate Charge Concentrations **(c)** Single-peaked, asymmetric profile

Figure 5.6: CAE reconstruction quality across different beam profile types. Blue: original profiles, Orange: reconstructions.

The reconstruction lacks in multiple aspects:

- **Discontinuities**: Profiles with separate charge concentrations like in Figure 5.6b are the worst reconstructions overall, not only having an incorrect reconstruction, but also introducing discontinuities in the profile that are completely unphysical.

- **Profile Shifts**: The reconstruction introduces a shift to the double-peaked profiles secondary peaks (Figure 5.6b), potentially disrupting the RL actor's capabilities, given that it does not have convolutional layers and, thus, does not have translational invariance.

- **Adding Non-Existent Features**: Besides the shift in the horizontal axis, the profile's smaller peaks are made larger, which could incorrectly indicate that there are smaller corrections to be made than what is actually required.

These limitations likely arise from the combination of parameters that the CAE has to map to the same phase correction, a task on which it focuses more after the introduction of the auxiliary phase head. The reconstructed profiles in Figure 5.6a and 5.6b look similar to $r_2 > 1$ profiles that are offset in Φ_2 by the same amount, which likely means that the CAE is understanding that because they map to the same correction, their reconstructions should be similar. This is desired for the feature extraction and reducing the dimensionality for the RL input, but not for augmenting the raw profiles and providing more robustness when deploying the RL agent.

Training an RL agent on such corrupted data would teach it to respond to impossible scenarios, potentially leading to degraded performance when deployed on real beam profiles. The agent might learn to exploit these non-physical artifacts rather than understanding genuine beam dynamics. As a result, this analysis forces neglecting the data augmentation approach and, rather than using the CAE for profile augmentation, two approaches arise for the RL agent:

1. **Direct latent space input**: Feed the 64-dimensional latent vectors directly to the RL agent, leveraging the phase-relevant clustering shown in the t-SNE analysis and avoid reconstruction entirely.

2. **Traditional augmentation**: Apply physics-preserving augmentations (noise, baseline droop) directly to original profiles, maintaining data fidelity while increasing training diversity.

The comparative training results presented later will determine which approach yields superior phase correction performance. However, the autoencoder analysis already provides valuable insights: it confirms that beam profiles contain learnable phase information (partially validating the CBAM and CNN approach) while highlighting the challenges of dimensionality reduction in this domain.

5.1.3. Hyperparameter Optimisation

Initial Optimisation

In the first phase, all hyperparameters were optimised jointly in a single Optuna study comprising 190 trials. The learning rate was sampled once per trial and then reduced with a quadratic schedule for the entire training run of that trial. As the learning rate directly controls the magnitude of the weight updates, it dominates the loss and consequently explains its large contribution of the variance in the results. This is reflected in Table 5.2, where fANOVA attributes 93.64% of the total variance to the learning rate, while MDI assigns it 73.16%. Note that because we are working with stochastic regressors (Random Forests are inherently stochastic, as explained in Section 4.8), the importances were averaged over 1000 runs with randomized seeds.

Despite the dominance of the learning rate, the remaining parameters still receive non-negligible importance scores, indicating that they influence the model when the learning rate is fixed. For instance, the convolution kernel size, the widths of the two fully-connected layers, and the number of training epochs each explain between 1.41%, 1.29% and 1.36% of the variance. This makes sense as the how much information from the profile is considered in each convolution is determined by the size of these kernels (besides the parameters that define the kernel itself), and the fully-connected layers combine all the feature maps' information into the predicted correction $\Delta\Phi_2$. Meanwhile, the training epochs influence the final validation loss as it defines how many gradient updates are performed. These small but important contributions are meaningful because they reveal which architectural choices are robust to variations in the learning rate.

The insight gained from this first analysis motivated a change of focus. The overwhelming influence of the learning rate suggested that a more sophisticated learning-rate schedule could reduce its apparent importance and allow finer control over other factors such as kernel size and network depth. Consequently, we introduced a scheduler that reduces the learning rate on plateau and expanded the search to include a third convolutional layer and a broader range of kernel sizes.

Table 5.2: Parameter importance (% variance) for the Attention CNN Optuna study (averaged over 1000 runs)

Parameter	fANOVA (%)	MDI (%)
Learning rate	93.64	73.16
$\text{Conv}_{1/2}$ kernel size	1.41	2.60

Continued on next page

Table 5.2 – continued from previous page

Parameter	fANOVA (%)	MDI (%)
FC layer 1 width	1.03	1.50
Training epochs	1.36	3.86
Batch size	0.93	1.44
Conv2 channels	0.70	1.50
FC layer 2 width	0.26	2.14
Conv_1 channels	0.45	1.50
Channel-attention reduction ratio	0.12	1.01
Weight decay (L2)	0.09	11.28

Final Optimisation

In the second phase, we replaced the constant learning rate with the scheduler described in the methodology: reducing the learning rate when validation loss stagnated. The scheduler automatically decays the learning rate when the validation loss stops improving, thereby diminishing the role of the initial learning rate in overall performance. To give every configuration enough time to converge despite the initial learning rate, the maximum number of epochs per trial was fixed at 700. Because the scheduler adjusts the learning rate dynamically, the epochs and initial learning rate hyperparameters became far less decisive; they now serve only as upper bounds rather than as dominant factors.

The search space was also enriched:

- Kernel sizes for both convolutional layers were expanded to $\{3, 5, 7, 9, 11, 13, 15\}$ and $\{21, 23, 25, 27, 29, 31, 51\}$, respectively.
- A third convolutional layer (optional) was introduced, with its own kernel size and channel count.
- Dropout rates for the fully-connected part of the network were added to reduce overfitting.

A total of 66 trials were completed, limited by the computational budget. The additional convolutional layer did not yield a reduction in validation loss; consequently, the best configurations omitted this extra layer. The optimisation was therefore concentrated on kernel size tuning and dropout regularisation, with those yielding the highest contributions to the variance: 26% and 20%, respectively.

The initial and final set of hyperparameters that emerged from this stage is listed in Table 5.3. Compared with the initial optimisation, the kernel sizes for both Conv_1 and Conv_2 increased substantially: from 7 to 15 and from 7 to 31, respectively. Also, the number of filters grew from 16 to 24 for Conv_1 and 32 to 48 for Conv_2, and the channel-attention reduction ratio was relaxed from 4 to 6. The learning rate was reduced by roughly a factor of four, indicating that it did not need to be as high as it was before with a different scheduler. These changes reflect the shifted emphasis toward expressive convolutional kernels and stronger regularisation, now that the learning rate no longer dominates the loss landscape.

Table 5.3: Best hyperparameters shared between initial and final optimisations, with descriptive names. Conv2 kernel size is included for completeness (not tuned in initial).

Parameter	Description	Initial	Final
Learning rate	Step size for gradient-based optimisation (optimiser LR).	8.15×10^4	2.0×10^4
Conv_1 filters	Number of output channels in the 1st convolutional layer.	16	24
Conv_2 filters	Number of output channels in the 2nd convolutional layer.	32	48
Conv_1 kernel size	Kernel size for the 1st convolution.	7	15
Conv_2 kernel size	Kernel size for the 2nd convolution.	–	31
FC1 width	1st hidden layer size in the dense layer after feature extraction.	192	512
FC2 width	2nd hidden layer size in the dense layer after feature extraction.	128	64

Continued on next page

Table 5.3 – continued from previous page

Parameter	Description	Initial	Final
Channel-attention reduction	Squeeze ratio in channel attention (smaller means stronger compression).	4	6
Weight decay (L2)	L2 regularization coefficient applied to weights for the AdamW optimiser.	2.3824×10^5	1.3276×10^6
Batch size	Samples per optimisation step (mini-batch size).	64	64

Validation Loss Comparison

The ultimate metric of interest is the validation loss achieved by each model after the respective optimisation phases. Table 5.4 reports the loss for three architectures: the Convolutional Block Attention Module (CBAM), a plain CNN, and a multilayer perceptron (MLP). The CBAM benefits most from the refined hyperparameter search, showing a relative improvement of +6.33% (i.e., a lower absolute loss) compared with the initial optimisation. The CNN and MLP exhibit only marginal changes, which is expected because the hyperparameter optimisation was performed for the CBAM model knowing that it would probably have better performance due to the initial results presented. This highlights that while the optimisation does improve performance for the individual parts of the model, it is catering much more to the collective improvement of the model and the interactions between its parameters, which is exactly what is desired.

Table 5.4: Validation loss for initial and final optimisations

Model	Initial Validation Loss	Final Validation Loss	Δ Improvement (%)
CBAM	1.4804×10^{-3}	1.5740×10^{-3}	+6.33
CNN	2.4387×10^{-3}	2.4606×10^{-3}	+0.90
MLP	4.3544×10^{-3}	4.4097×10^{-3}	+1.27

Statistical Reliability of the Importance Measures

The importance percentages reported in Table 5.2 are derived from a small Monte-Carlo estimate (using 1000 random runs) of the variance decomposition. This repeated analysis is fundamental given that the Random Forests used in either estimate are stochastic in nature, so using different seeds and repeating the estimate. Both estimators are unbiased under the assumption that the sampled hyperparameter space is representative of the true underlying distribution, which in turn depends on the amount of area covered in the hyperparameter space (so the number of trials performed). However, by repeating the importance calculation 1000 times and reporting the mean, we reduce the variance of the estimator by a factor of 1000, and we use shallow decision trees to avoid overfitting to the data. Furthermore, the resulting standard deviations (not shown to keep the table concise) were all below 0.5%, indicating that the reported percentages are stable.

Furthermore, the TPE sampler used in Optuna guarantees that the distribution of sampled points asymptotically converges to regions of high expected improvement, which aligns with the goals of both exploration (wide coverage) and exploitation (focus on promising configurations). This property ensures that the importance analysis is not biased towards a narrow subset of the space, but rather reflects the behaviour of the model across the most relevant hyperparameter configurations. In summary, the combination of a representative sampling strategy for the data, the repeated importance estimation, and a clear reduction in validation loss gives us confidence that the hyperparameter configuration identified in the final optimisation is statistically robust. Nevertheless, more trials could be performed to lower the loss even further and increase performance.

5.1.4. Model Analysis

This section presents a comprehensive analysis of the CBAM model performance for correcting Φ_2 in a simulated environment with BLonD. We begin with an ablation study to evaluate the contribution of different architectural components, followed by an in-depth analysis of the attention mechanisms to understand how the model identifies relevant features in the input data. Finally, we present a verification analysis that examines the model's performance across all operational beam configurations.

Ablation Analysis

Following hyperparameter optimisation, another comparative training was performed using the exact same dataset with the same stratified split to conduct an ablation study to assess the contribution of each architectural component to the model's overall performance. Figure 5.7 presents the training and validation loss evolution for the three model variants: the complete CBAM architecture, the CNN without attention modules, and the baseline MLP model.

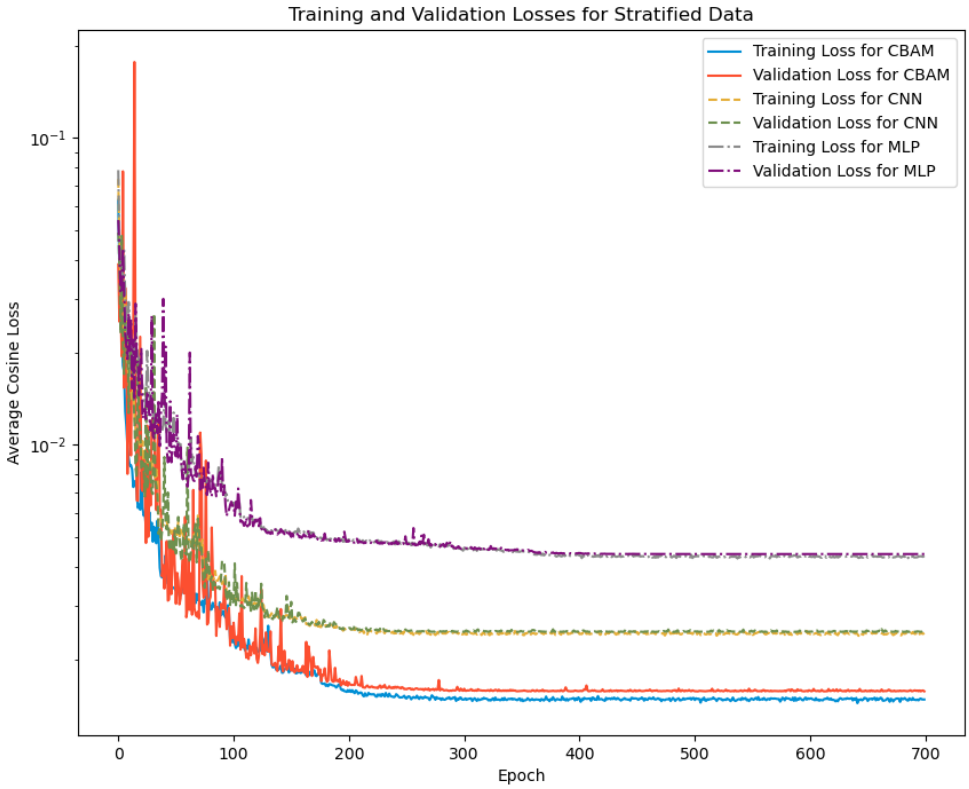


Figure 5.7: Training and validation loss evolution for the CBAM, CNN and MLP models after hyperparameter optimisation with the same training and validation datasets

The results demonstrate a clear performance hierarchy among the three architectures. The complete CNN-CBAM model achieves the lowest validation loss, converging to approximately 1.5×10^{-3} after 700 epochs. The CNN without attention modules shows intermediate performance with a validation loss around 2.5×10^{-3}, while the baseline MLP exhibits the highest validation loss at roughly 4.5×10^{-3}.

This progressive degradation in performance when removing architectural components validates the design choices. The superior performance of the CNN over the MLP can be attributed to the convolutional layers' ability to capture local patterns and dependencies in the bunch profiles through their inherent signal processing capabilities. The further improvement achieved by incorporating CBAM modules demonstrates the value of the attention mechanism in selectively focusing on the most informative features for phase correction. This also benefits understanding the model as will be shown in later discussions.

The consistent gap between model performances throughout the training process and also the different points at which they plateau with the adaptive scheduler chosen indicates that these architectural advantages are fundamental rather than just affecting convergence speed. The attention mechanisms appear to play a crucial role in identifying and prioritizing the most relevant spatial and channel-wise features for accurate phase prediction, as will be further demonstrated in the following section through detailed attention map analysis.

Attention Mechanisms

To understand how the CBAM model learns to predict the optimal second harmonic phase for bunch lengthening in the PSB, we analysed the attention weights generated by both channel and spatial attention modules. This analysis provides crucial insights into the features that the model prioritizes for different types of bunches and beams.

The channel attention mechanism, applied after the first convolutional layer, exhibits relatively subtle modulation of feature importance. As shown in Figure 5.8, the channel attention weights μ^j remain remarkably close to a uniform baseline of 0.5 across all 24 channels, with only minor variations depending on the characteristics of the input profile. This near-uniform weighting suggests that the initial convolutional features extracted from the longitudinal bunch profile and its derivative all contribute approximately equally to the phase prediction

task.

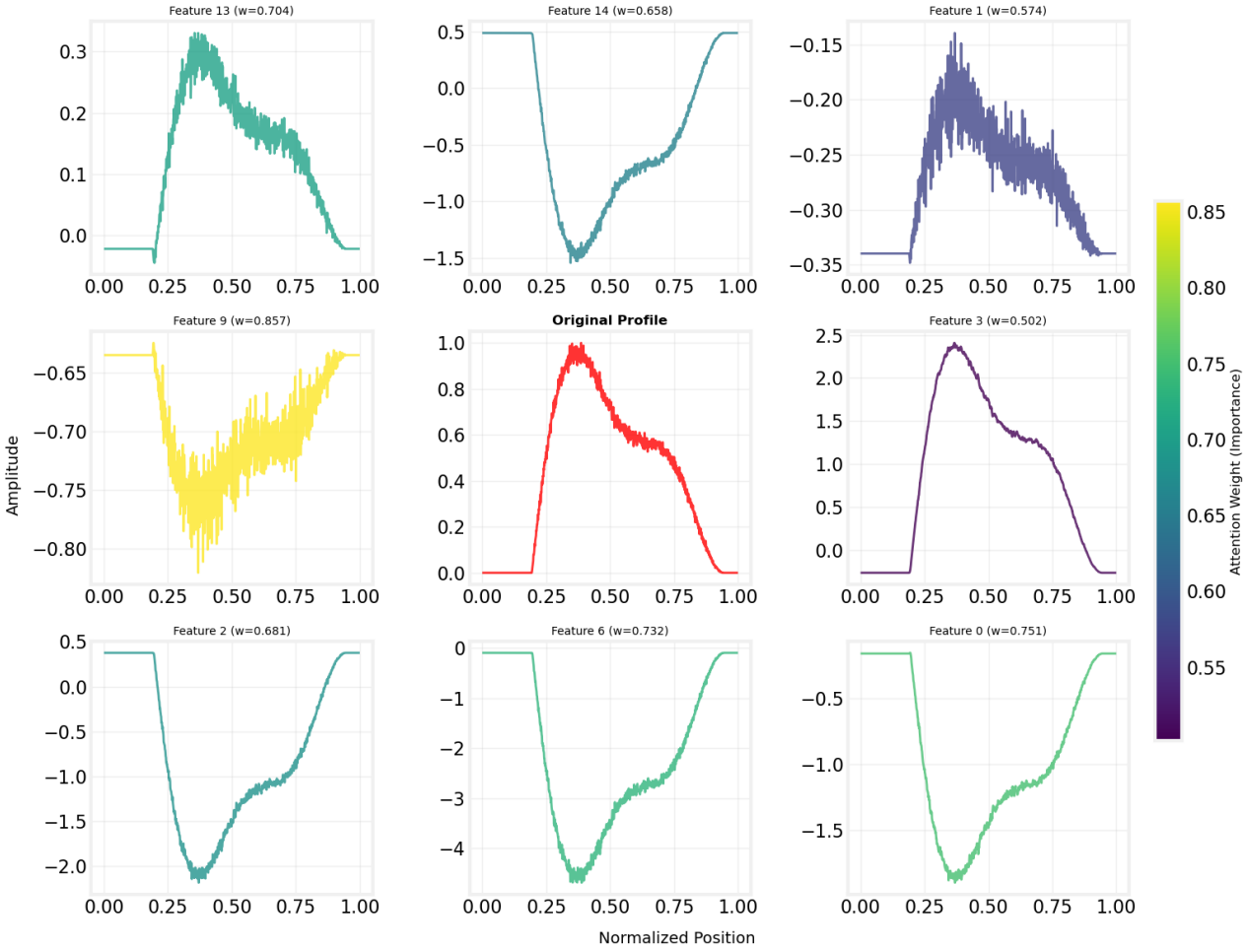


Figure 5.8: Channel attention weights μ^j applied after Conv_1, visualised through coloured features for 8 randomly selected features.

The subtle variations in channel weights (ranging from approximately 0.52 to 0.85) indicate that while certain features may be slightly more informative for specific profile types, the model benefits from maintaining access to the full spectrum of learned features coming from the 1st convolution. This behaviour aligns with what is expected, as the features from the 1st convolution will then be processed again into more complex features after passing through the 2nd convolutional filter.

Meanwhile, the spatial attention mechanism demonstrates far more discriminative behaviour, particularly when analysed separately for different emittances. For high emittance beams, the spatial attention weights α_i exhibit a pronounced focusing effect on specific spatial locations within the bunch profile.

As illustrated in Figure 5.9, the spatial attention effectively acts as a soft gating mechanism, applying weights as low as 0.0005 to suppress less relevant spatial locations while emphasizing critical regions with weights up to 0.004. This represents nearly an order of magnitude difference in importance attribution. Furthermore, the attention consistently highlights the edges of the bunch profile when looking at high emittance bunches, precisely the regions that determine the bunch length and are most affected by the Φ_2 adjustment.

This edge-focusing behaviour is particularly significant given that the primary objective of the phase correction is bunch lengthening and that the bunch lengthening is most noticeable at high emittances. The model has autonomously learned to prioritize the spatial locations that directly correlate with the control objective, demonstrating that the attention mechanism successfully captures the underlying physics of the beam dynamics problem.

The spatial attention mechanism exhibits different behaviour when processing low emittance beam profiles, showing an adaptive strategy that accounts for the limitations of max-normalization discussed in Figure 5.1.

For low emittance cases, where the normalised profile width becomes less representative of the actual bucket

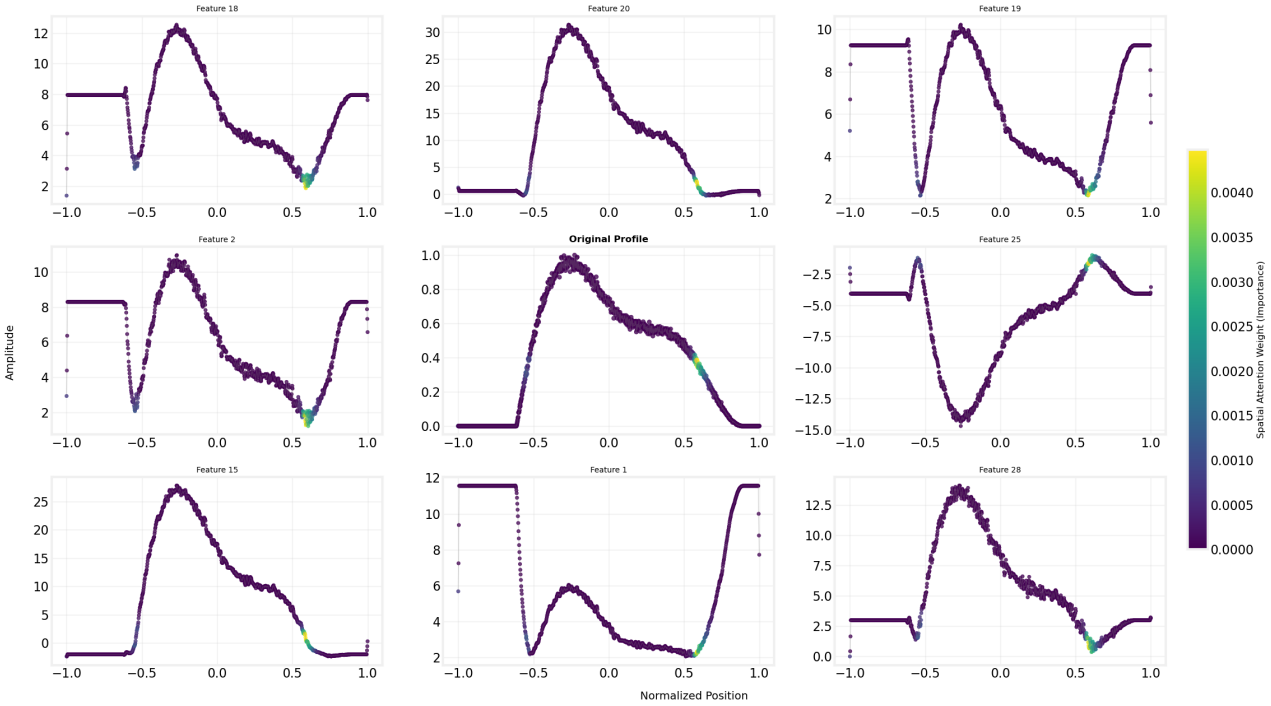


Figure 5.9: Spatial attention weights α_i applied after Conv_2, for high emittance visualised through coloured markers for 8 randomly selected features.

length and of how well phased Φ_2 is, the spatial attention shifts its focus from the edges to the peak regions and asymmetries in the centre of the profile. As shown in Figure 5.10, the attention weights now emphasise the central peak and any asymmetric features in the profile shape, with attention values ranging from 0.002 to 0.010,notably higher than in the high emittance case.

This adaptive behaviour is particularly encouraging as it mirrors the exact approach used during dataset generation, where asymmetries in the bunch profile were key indicators for determining the optimal phase correction when at low emittances (this was the fine" mode in the dataset generator). The model's ability to independently discover and utilize these same features validates both the architecture design and the physical relevance of the learned representations.

Furthermore, the visualisation of these attention mechanisms reveals insights into the model's learning process and its alignment with beam phasing principles. The model demonstrates remarkable physical consistency by focusing its attention on bunch edges for length control in high emittance beams while shifting to profile asymmetries for low emittance beams, mirroring the strategies used during dataset generation described in Section 4.1. This adaptive capability, where the spatial attention mechanism automatically adjusts its focus based on the emittance regime without explicit programming nor knowing the emittance, significantly enhances trust in the model's predictions. Rather than merely fitting statistical patterns in the data, the CBAM-enhanced CNN has demonstrably learned physically meaningful representations that correspond to our theoretical understanding of the bunch lengthening process.

5.1.5. Verification Analysis

To comprehensively evaluate the performance of the trained CBAM model, we conduct an extensive verification analysis across all operational beam types in the PSB. Although complete verification plots for all operational beams are provided in Appendix B, and errors are presented for all beam types, this section focuses on four representative beam cycles: BCMS, ISOLDE, TOF, and MTE. The analysis demonstrates the model's ability to generalize across different operational conditions, which is particularly important given that verification in simulation can be performed inexpensively compared to machine time.

Phase solutions are computed only where the harmonic-voltage ratio is physically meaningful for shaping $0.2 < r_2 < 4$, since outside this window the second harmonic either has negligible leverage on flattening the potential well (low r_2) or dominates bucket shape and changing it just shifts the bucket making the phase change irrelevant (high r_2).

Global Error Summary

The aggregate metrics in Figure 5.11 should be interpreted as *indicative* rather than absolute for two reasons:

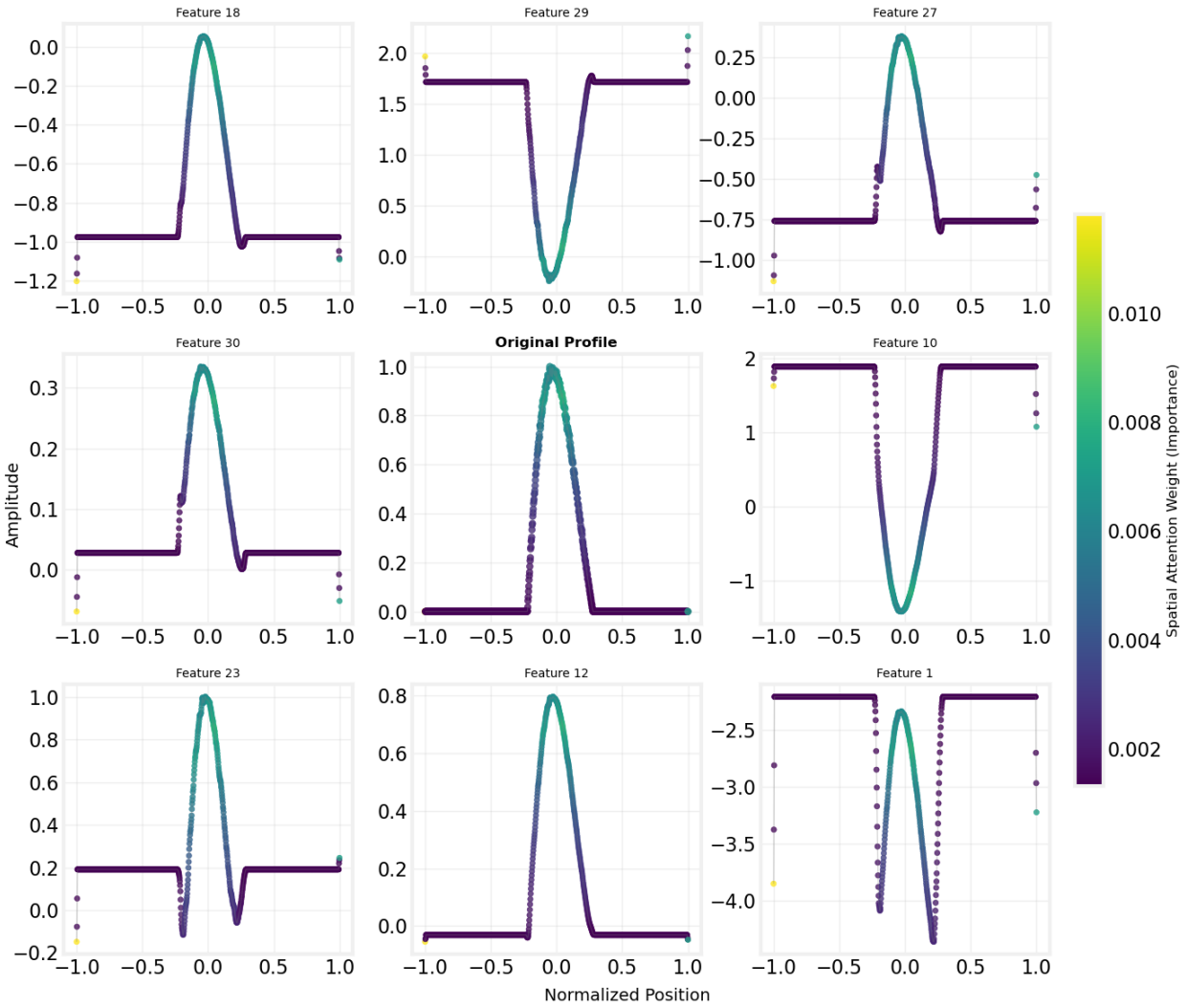


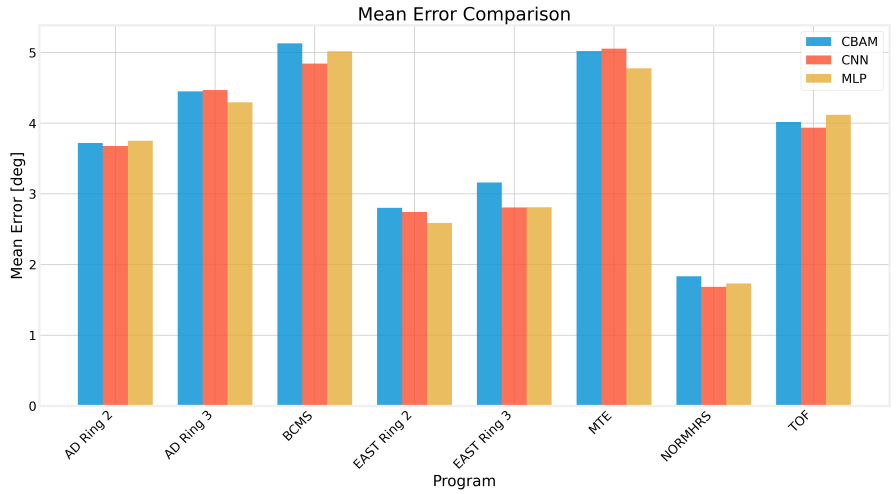
Figure 5.10: Spatial attention weights α_i applied after Conv_2, for low emittance visualised through coloured markers for 8 randomly selected features.

1. They are computed in BLonD using a model of the PSB impedance that differs from the actual machine impedance and, as shown later in Section 5.1.7, the operational profiles in the machine are not perfectly phased either. Hence, numerical error bars are useful for relative comparison but not as definitive performance bounds.

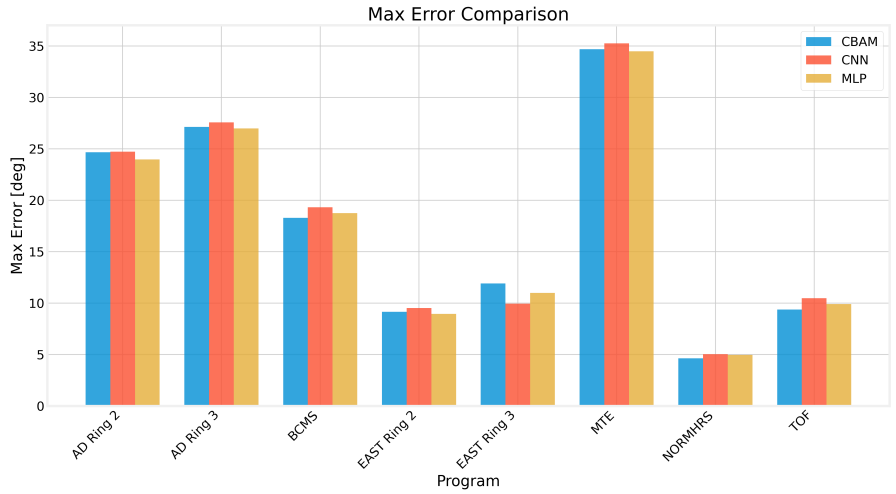
2. For programs such as AD (Ring 3) and MTE, the late part of the cycle exhibits very high r_2 in order to produce two bunches. In that regime V_2 takes over and due to how the solution is computed by matching, and not tracking particles, the maximum error increases for these cycles. The matching process is not considering the temporal aspect of the splitting where the distribution of particles as the profiles are being split is important to determine how many particles will lie in each bunch, and it will be shown in detail for MTE.

Viewed through the operational intent of each program, the trends are consistent. For programs seeking compact bunches at extraction (which is basically all cycles extracted to the PS, which excludes ISOLDE-type beams) show comparable mean and variance across models. The practically relevant metric is whether bunch lengthening is correctly applied during the acceleration phase and if the model can cope with the big jumps in V_1 (as is done for TOF). Interestingly, the lowest errors are achieved for the ISOLDE cycle, which is the only cycle that extracts at a low energy (1.4 GeV) and where V_2 is sufficiently high at extraction such that the bunch is still flattened in the correct phasing.

To show this analysis in a more detailed way, the following presents a profile-based verification for each program, emphasizing bunch-length evolution and symmetry rather than looking solely at the absolute phase error. We employ a decaying correction sequence (10 steps linearly weighted from 1.0 to 0.1) to emulate convergence, given that the model inherently lacks this due to its regressive nature. Additionally, all initial conditions start



(a) Mean error across programs.



(b) Maximum error across programs.

Figure 5.11: Summary of phase-error metrics for CBAM, CNN, and MLP models across all operational PSB cycles

from a constant phase throughout the cycle of $-90°$.

BCMS

The corrections applied at each C-Time for 10 iterations and the final Φ_2 solution are shown in Figure 5.12. The phase set in the machine is also shown and the difference between them is highlighted for the relevant Φ_2 computation points. Meanwhile, the initial and final longitudinal profile evolutions are compared in Figure 5.13. It should also be noted that the voltage programs for the 2025 BCMS beam were used, which uses triple-harmonic operation at injection, which is why the profiles are split at injection (as V_3 is missing), which would be significantly detrimental to space charge mitigation. Regardless, it is still relevant to see if the system can phase it correctly as the computation of Φ_3 is independent of the second harmonic (but the correct flattening depends on the combination of V_1, V_2 and V_3) so this serves to verify that it can still be used to aid even in triple harmonic optimisation. Also, the fact that the bunches are split implies a higher charge density, which should shift the phase through the space charge impedance, and it will be fundamental to the performance in the PSB if the model can cope with it in simulation.

The correct bunch lengthening is achieved throughout the cycle, with the peaks being matched in the cases that it is required (up to C-700). After that, the phase remains constant at $-90°$ as V_2 is too low to have an effect, and even though the phase at C-700 is not set at $-90°$, it still achieves the correct flattened profile. This shows that the phase set in the machine is not necessarily the most optimised one in simulation, so as long as a similar shape is achieved in the final Φ_2 program and the correct final waterfall is achieved, the model can be considered to meet the verification requirement. This could be due to slight differences in the impedance model or in the way that the servoloop behaviour is modelled by simply dividing the impedances by a factor of 50. Additionally, it is important that no abrupt changes are performed in the phase program as it would imply that the bucket shifts quickly and unwanted oscillations would appear in the profile.

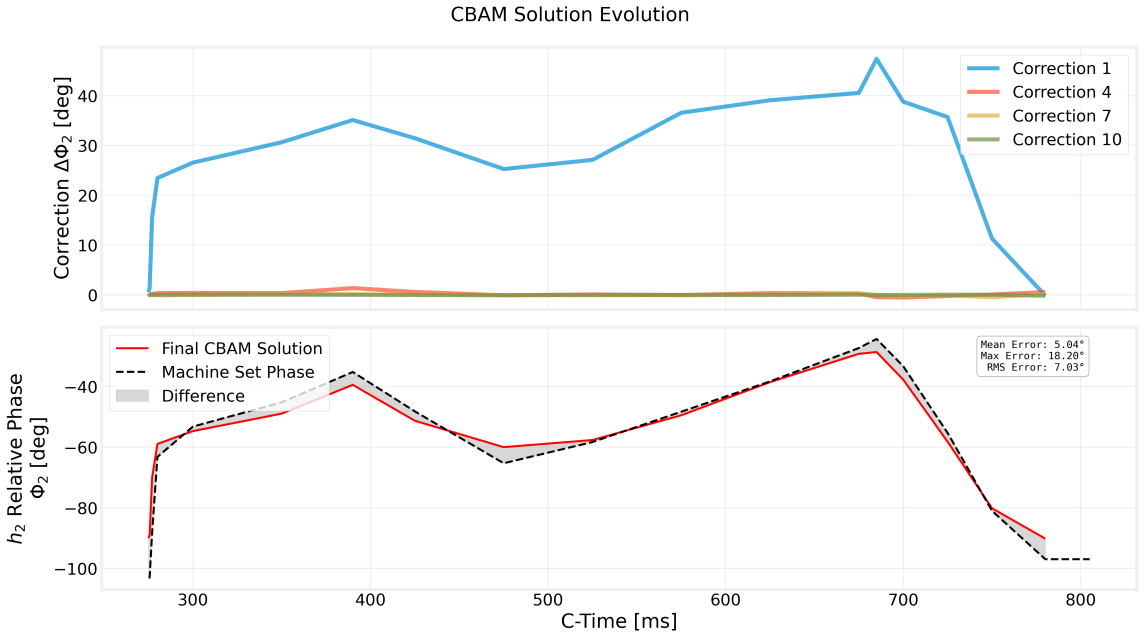


Figure 5.12: BCMS (2025) phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

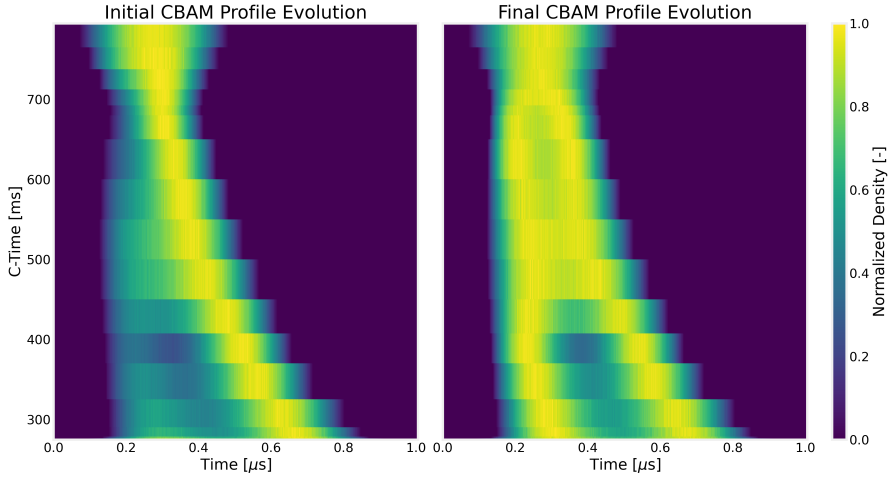


Figure 5.13: BCMS (2025) normalised longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

ISOLDE

The corrections applied at each C-Time for 10 iterations and the final Φ_2 solution are shown in Figure 5.14 with the initial and final waterfall plots in Figure 5.15. In this cycle the extraction energy is lower, so V_2 remains comparatively high up to extraction since microwave instabilities at extraction are not an issue at these energies, as analysed in Chapter 3. The phase therefore maintains relevance in the profile throughout the entire ramp, allowing full-cycle optimisation.

The desired bunch lengthening is consistently achieved throughout the cycle. The final Φ_2 program remains in a BLM configuration with smooth variations and no abrupt changes, maintaining a flat, widened profile. Where the model departs from the machine set phase, the corrected profiles still exhibit the intended flattened shape, confirming that the learned solution is physically consistent as long as the final Φ_2 program yields the profiles throughout the cycle. Furthermore, lengthening is the operative goal at all C-Times and r_2 remains within the actionable range until extraction, the corrections are effective everywhere. This is also why ISOLDE attains the smallest error bars among the programs, as the highest errors are usually attained near extraction for the higher energy cycles (see Appendix B to confirm this over the full range of operational cycles) where r_2 is reduced in order to increase the synchrotron frequency spread, increase the energy spread and reduce the chance of microwave instabilities occurring. This makes the phase less determined than for high r_2 values.

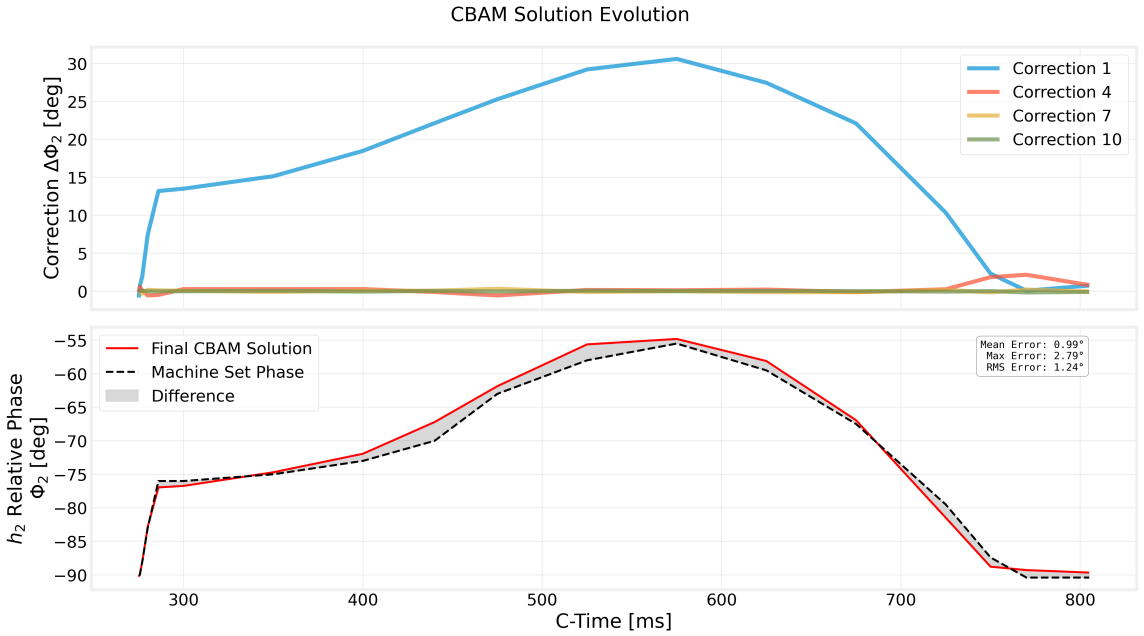


Figure 5.14: ISOLDE phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

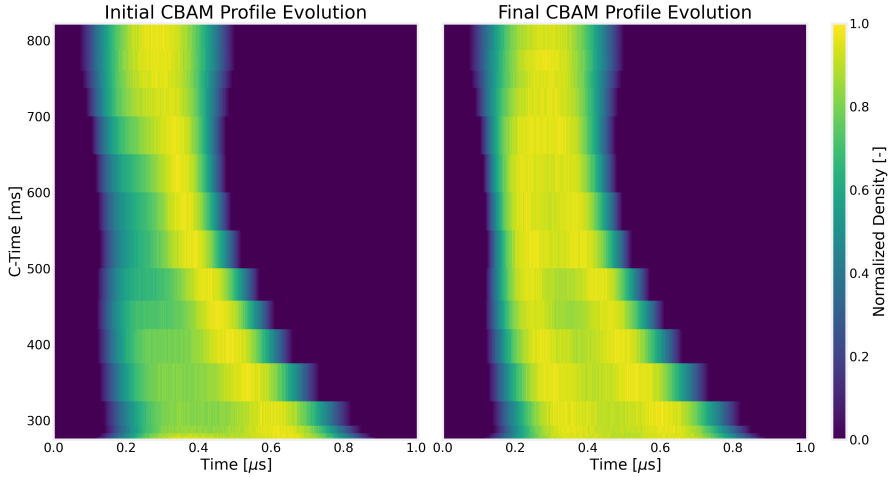


Figure 5.15: ISOLDE normalised longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

MTE

As before, the corrections and solution for Φ_2 are shown in Figure 5.16 with the initial and final waterfall plots in Figure 5.17. In this cycle the main objective is the controlled splitting; consequently Φ_2 remains relevant throughout the cycle.

The waterfalls confirm that the splitting is performed correctly, with two buckets with a balanced charge distribution after the splitting is performed. Near extraction, when r_2 becomes large and V_2 dominates, Φ_2 diverges considerably from the value manually optimised in the machine because at this point in the cycle $V_1 << V_2$, then changes in Φ_2 are mainly shifting the bucket position, and not affecting the charge distribution as much. This explains the large late-cycle errors in Figure 5.11b despite the profiles being correct, and it is also visible in Figure 5.16 since the first correction performed is the dominant one. Furthermore, it is also visible in the left panel of Figure 5.17 for the initial waterfall, since the splitting there is almost optimal, yet the correction performed completely evens out the amount of charge in both bunches. This suggests that because we are merely matching the distributions and not tracking the particles as they are getting split into the different bunches, that the way the particles lie on either potential well is dictated by the splitting process and not the potential well depth as this is what is considered when they are being matched (see Figure 2.24 and 2.25).

In addition, since we are using the same computation points as in the manually phased solution in the PSB, a

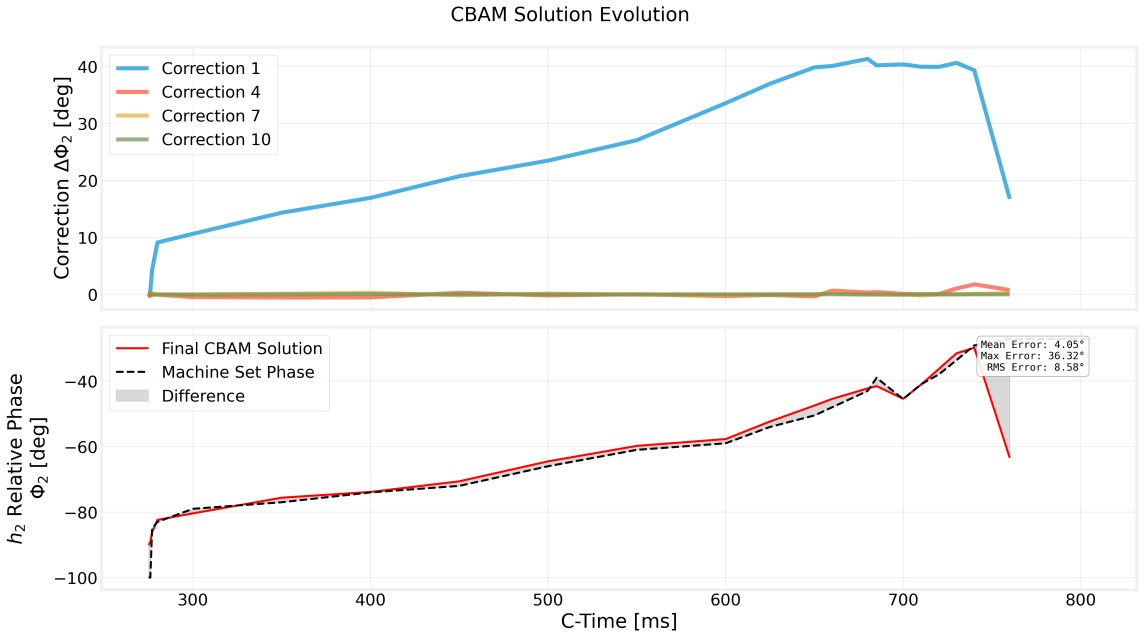


Figure 5.16: MTE phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

higher density of Φ_2 computation points is used near extraction to ensure equal partitioning of charge between the two buckets, which is required by the splitting process and is reflected in the solution granularity in Figure 5.17 from C-720 to C-770. However, unlike previously, abrupt phase jumps are not entirely avoided, especially at the last computation point, so it is yet to be seen if this will cause issues in the actual machine implementation. As a result, the model is verified only tentatively for MTE beams until the machine solution confirms it.

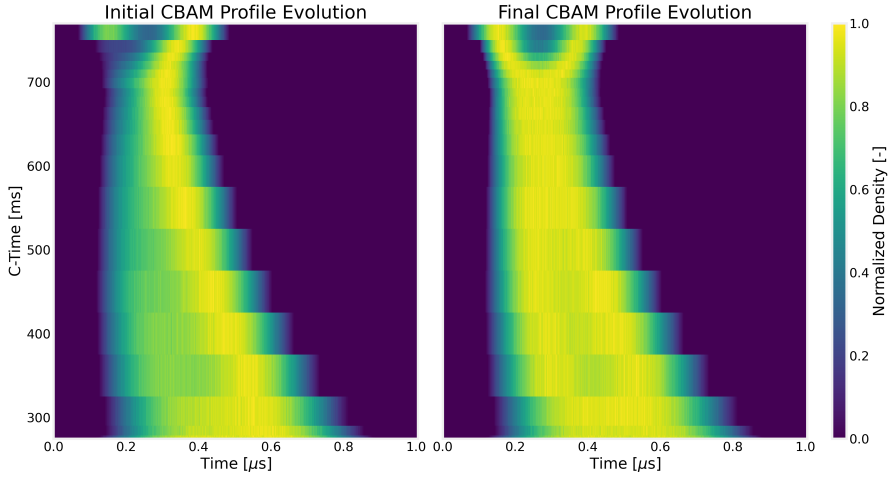


Figure 5.17: MTE normalised longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

TOF

Finally, the corrections and Φ_2 solution for the TOF cycle are shown in Figure 5.18 with the initial and final waterfall plots in Figure 5.19. For this high-intensity cycle, bunch lengthening is required in order to mitigate the transversal tune spread induced by space charge. However, because of the combination of this high intensity and high extraction energy the cycle is particularly susceptible to instabilities, unlike the ISOLDE cycle. Thus, in order to mitigate this and increase the synchrotron frequency spread, a large increase in V_1 is applied at C-720 in the voltage programs, and the Φ_2 solution is only computed up to this point.

Despite a pronounced difference between the computed solution and the phase set in the machine, the final profiles remain symmetric and lengthened as required. This discrepancy likely reflects a larger mismatch for this particular cycle between the PSB impedance used in simulation and the actual machine impedance. It could be due to a combination of the control loops used, or due to the amount of manipulations involved, but

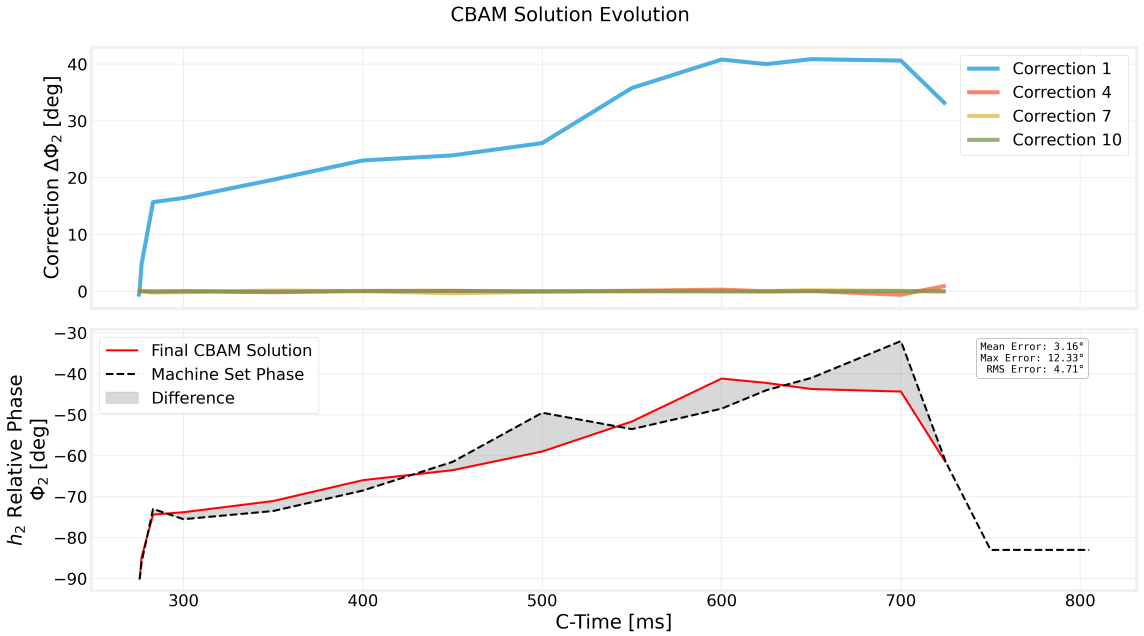


Figure 5.18: TOF phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

Φ_2 is practically changing in the opposite direction in the model-computed solution. The increase in V_1 near C-720 also reduces the importance of Φ_2 exactly where the largest deviations are observed. Nevertheless, the corrected waterfalls satisfy the operational objective: sustained bunch lengthening with stable symmetry up to the computation limit of C-720. Whether the machine will follow the same Φ_2 trajectory remains to be established; if not, the origin of the discrepancy must be understood (e.g., impedance modelling, profile errors in the machine itself, or servoloop behaviour). As with the other cycles, abrupt changes in the phase program are avoided to prevent bucket translation and profile oscillations.

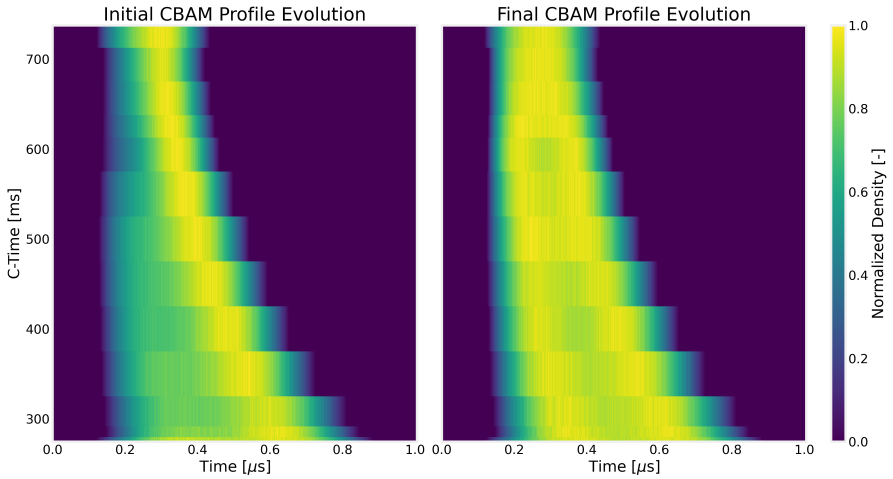


Figure 5.19: TOF normalised longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

In conclusion, all cycles achieve the desired behaviour in terms of the profiles produced, yet there are considerable discrepancies with the solutions set in the PSB and those of the model. This difference is also maintained for the other models, with them also achieving the desired profiles, which is shown in Figure 5.11a with the different models having almost the same mean error. This is something that will need to be analysed when implemented in the PSB itself to determine whether it is an error in the impedance model or if it is because the profiles in the machine are not optimal. Furthermore, it is also important to determine the behaviour of the decaying corrections in the presence of an increased impedance to characterize its generalization properties.

Effect of Initial Phasing & Varying Impedance

We study the sensitivity of the solution to the starting Φ_2 for the optimisation when the PSB Finemet longitudinal impedance is increased by a factor of two in both its real and imaginary parts. However, the considerations

on the starting point for the solution are still relevant in general, not only to higher impedances, only that this case was found to show large sensitivity to the starting conditions. The ISOLDE cycle will be analysed, as this is the case where the CBAM solution is the closest to the machine set solution, so that the effect of the increased impedance can be discerned and evaluated in the solution.

Two cases are considered: (i) initialization at $\Phi_2 = 90°$ and (ii) initialization at $\Phi_2 = -90°$. The corresponding correction evolutions and final phase programs are shown in Figure 5.20 and Figure 5.21, with associated waterfalls in Figure 5.22 and Figure 5.23.

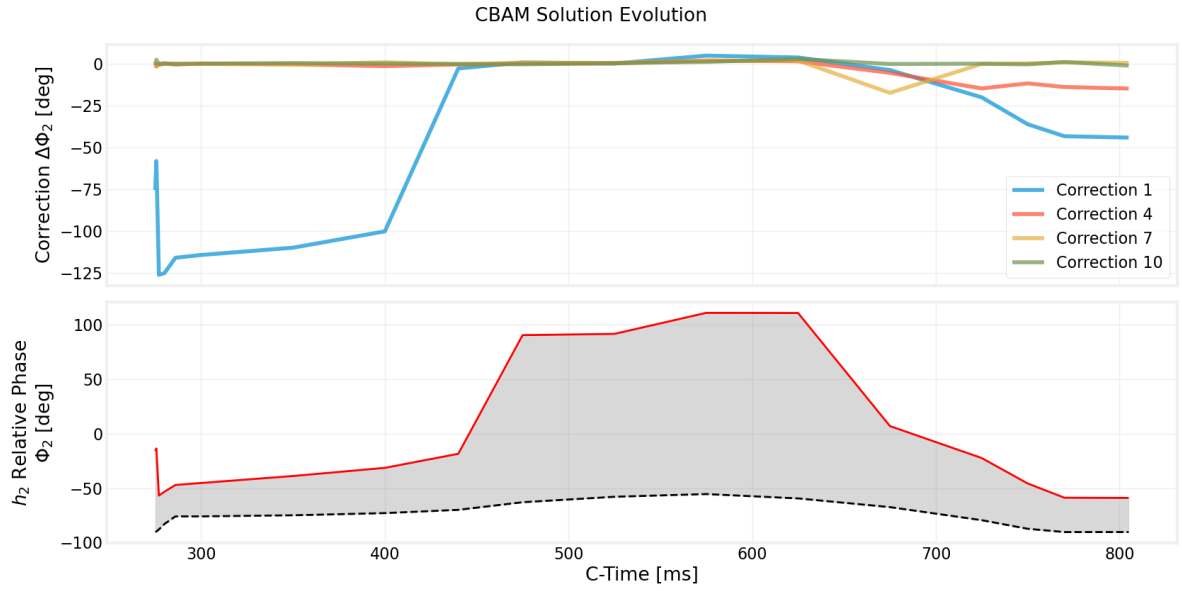


Figure 5.20: CBAM solution for ISOLDE cycle under twice the PSB impedance: correction evolution and final Φ_2 when initialized at 90° (CBAM solution in red as before)

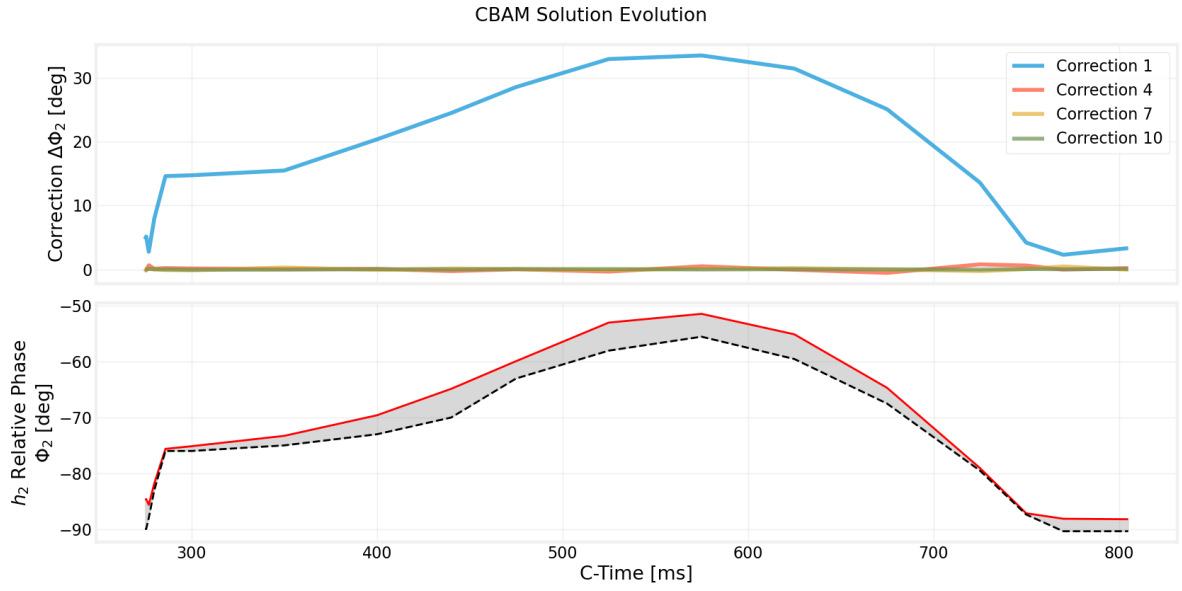


Figure 5.21: CBAM solution for ISOLDE cycle under twice the PSB impedance: correction evolution and final Φ_2 when initialized at −90° (CBAM solution in red as before)

When starting at 90°, the CNN interprets the profiles near the middle of the cycle as a low-emittance bunch that is already in or close to BLM. As a result, the proposed corrections remain very small and the solution stays near the initial trajectory. The waterfalls confirm limited shaping change and a tendency to preserve shorter, more peaked bunches in the middle of the cycle. In contrast, initializing at −90° seeds the optimisation in the correct Φ_2 region for BLM and the model then converges to the BLM phase. The final waterfalls show the expected broadened, symmetric bunches across the cycle achieving the desired shape throughout the cycle even in the presence of this higher impedance. These results highlight a key limitation: the current CNN lacks

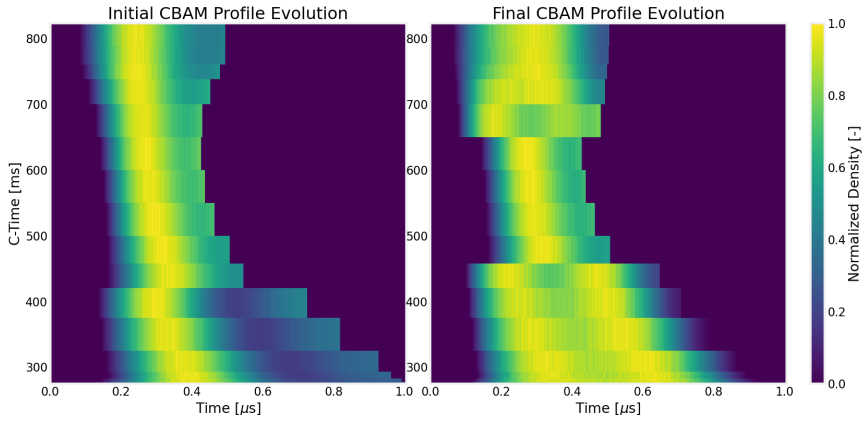


Figure 5.22: CBAM initial and final waterfalls for ISOLDE cycle under twice the PSB impedance when initialized at 90°.

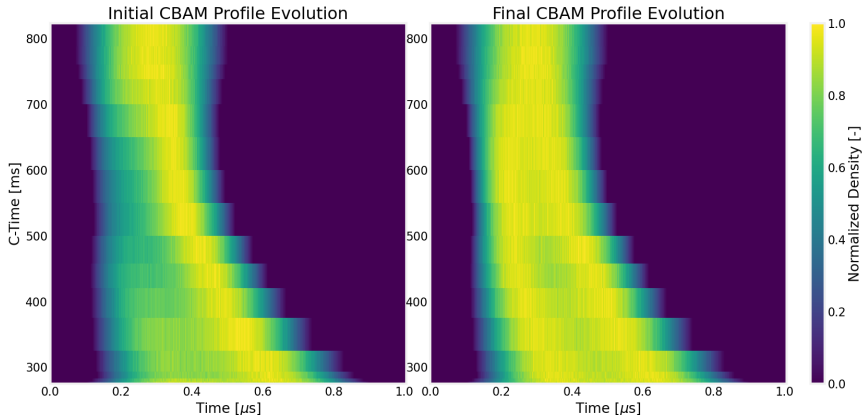


Figure 5.23: CBAM initial and final waterfalls for ISOLDE cycle under twice the PSB impedance when initialized at −90°.

exploratory capability and relies on a reasonable first estimate to deliver effective corrections. With a poor initial estimates for Φ_2 the model can fail to transition to BLM.

This is also due to the fact that the model has less information on the correct phase corrections to make when faced with a symmetric profile in BSM, as it has no asymmetry information to tell the model to increase or decrease Φ_2. A very simple, yet practical mitigation for this is to set the initial phase to the approximate synchronous phase defined in Chapter 4:

$$\phi_{s_{approx}}(t) \approx \arcsin\left(\frac{\Delta E_s(t)}{V(t)}\right), \quad \text{where } V(t) = \max(V_1(t), V_2(t)) \tag{5.5}$$

This seeds the optimiser close to the optimal phase with the correct relative shape. As a result, this is the technique used when implementing both the CBAM and LSTM-RL model in the PSB. However, before discussing the results, it is important to first discuss how the profiles are acquired and processed into the format that the CBAM model expects: centred profiles interpolated to one RF period.

5.1.6. Data Acquisition & Preprocessing

This section describes the acquisition code developed to provide robust, cycle-wide input data for the double and triple-harmonic models in the form of centred longitudinal bunch profiles that are interpolated to one RF period T_{rev}. The pipeline optimises the computation points (meaning the points in C-Time where the RF parameters are computed), the acquisition timing, centres the bunch in the acquisition window across the cycle, and delivers normalised, single-bucket bunch profiles that match the model's training data.

Computation Point Selection & Acquisition optimisation

The choice of where computation points are placed is of utmost importance, as operators select this depending on how accurate the second harmonic phase must be over the cycle. Thus, to place computation points where

corrections are most impactful, we combine a uniform time grid with inserted computation points at sudden changes of the approximate synchronous phase:

1. **Base grid.** Construct a uniform array of candidate times
$$\mathcal{T}_{\text{base}} = \{t_k\}_{k=0}^{K}$$
from $t_{\min}$ to $t_{\max}$ (defining the optimisation interval) with step Δt (usually 20 to 30 ms).

2. **Approximate synchronous phase.** Again using the magnetic field program and the machine parameters to compute the energy gain of the synchronous particle per turn $\Delta E_s(t)$. This energy as a function of time $E_s(t)$ is computed and interpolated on a turn-by-turn basis using the magnetic field ramp $\mathcal{B}(t)$ and machine geometry using Equation 2.8, the parameters in Table 2.1 and the rest energy of a proton. Furthermore, combining this with the RF voltages from the machine, we get $\phi_{\text{s}approx}(t)$ and take the magnitude of its time derivative $|\text{d}\phi_{\text{s}approx}/\text{d}t|$.

3. **Peak insertion.** Detect peaks of $|\text{d}\phi_{\text{s}approx}/\text{d}t|$ above a threshold. These times mark sudden bucket shifts where the second harmonic phase is expected to change significantly. Let $\mathcal{T}_{\text{peaks}}$ be the set of these peak times.

4. **Merge, de-duplicate, and sort.** Finally, we form the complete set of computation points
$$\mathcal{T} = \mathcal{T}_{\text{base}} \cup \mathcal{T}_{\text{peaks}}.$$
and cluster any times closer than a small window ϵ_t ($\epsilon_t \approx 5\,\text{ms}$) and replace each cluster by its mean time. Sort the result to obtain the final, well-spaced set of computation points $\mathcal{T}^\star$, and also store the values of $\phi_{\text{s}approx}(t)$ for the initial estimates of the solution for $\Phi_2(t)$.

This strategy ensures baseline temporal coverage via $\mathcal{T}_{\text{base}}$ while adaptively increasing resolution where $\phi_{\text{s}}(t)$ exhibits large variations, leading to more effective and meaningful Φ_2 updates in the same way that an operator would add computation points. An example of additional computation points is shown in Figure 5.24 for the TOF cycle, where the red dashed lines indicate automatically detected sampling points.

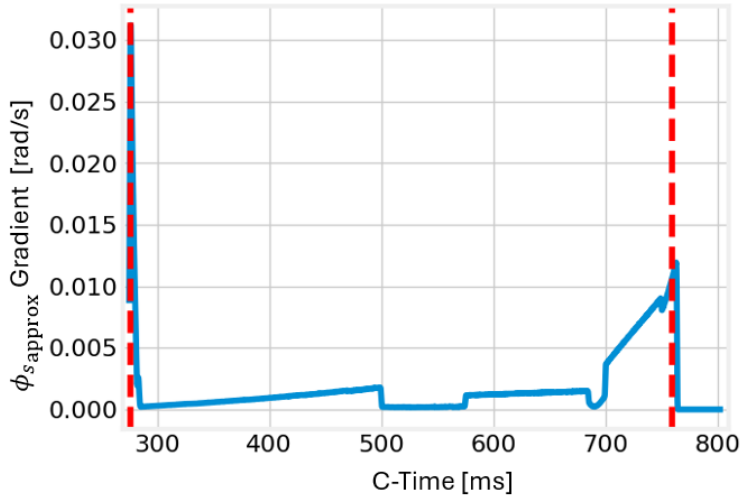


Figure 5.24: Automatic additional sampling using the gradient of (red dashed lines indicate additional computation points)

Furthermore, each burst is defined by a starting time c_{start}, a certain amount of traces (meaning the amount of acquisitions) acquired n_{traces} and a given amount of turns between each trace Δ_{turns}. The starting times for each of the 4 bursts are set with logarithmic spacing from $t_{\min}$ to $t_{\max}$ (without including $t_{\max}$) to be consistent with the fact that any more turns are performed per millisecond near extraction (this can be seen with the curve elongated on the right panel of Figure 5.26 in comparison to the curve on the left panel). Then, the for the fixed amount of 400 traces per burst (maximum allowed by the cards), the Δ_{turns} are iteratively optimised such that the distance of the traces to the computation points in $\mathcal{T}^*$ is minimized. Using this procedure, the bursts are defined to acquire longitudinal bunch profiles over the entire cycle.

Trigger Delay Compensation

During acceleration, the revolution period shrinks since the particles are both accelerating and not yet ultra-relativistic, so particles gain more velocity per increase in energy in comparison to accelerators further down

the chain (i.e. $\frac{dv}{dE}_{(PSB)} > \frac{dv}{dE}_{(PS)} > \frac{dv}{dE}_{(LHC)}$). This is shown in the left panel of Figure 5.26. So, a triggering system to acquire the profiles set at a fixed frequency would acquire profiles with a certain delay that would increase as the cycle time (C-Time) does. This produces a "tilt" in the waterfall plot and does not centre the bunch with respect to the acquisition window, possibly not capturing the entire profile at later points in the cycle. The acquisition pipeline first compensates this effect at injection and then propagates a per-trace delay schedule up to extraction to maintain centring.

In order to do so, we first define $T_{rev}(t)$ as the revolution period as a function of the cycle time computed by knowing the velocity of the particle with its $E_s(t)$ defined previously, and the circumference of the accelerator. Then, we consider a conservative initial delay (in ns) to begin the delay compensation:

$$\Delta t_{\min} \;=\; k_{\text{safety}}\,(T_{\text{rev}}(t_{\text{inj}}) - T_{\text{rev}}(t_{\text{ext}})) \times 10^9, \tag{5.6}$$

with $k_{\text{safety}} = 1.5$ being a safety factor used in operation. This initial delay is used because the delay has to be positive throughout the cycle in the parameters that we are feeding to the digital acquisition cards (that manage this triggering and acquisition process). As a result, the initial minimum delay has to be larger than the expected change in revolution period because otherwise the delay might become negative at some point in the cycle, which would give off errors in the pipeline.

Then, a single burst acquisition (in the PSB one can have up to 4 burst acquisitions), the calibration burst, is acquired at injection to capture around 100 traces over the first 3000 turns after injection using the delay $\Delta t_{\min}$. From the mean profile $\bar{p}$ of those 100 traces (we use the mean in order to average out injection transients), a Region of Interest (ROI) containing the bunch is extracted using the gradient of a smoothed trace:

1. Smooth $\bar{p}$ using a smoothing spline with the regularization selected via generalized cross-validation (GCV, see [**smooth_spline**] for more information), yielding $\tilde{p}$.

2. Compute $g = \nabla\tilde{p}$ and a normalised gradient g_{n} (scale positive and negative part of g to $[0, 1]$ independently, so g_{n} has the range $[-1, 1]$).

3. Define ROI bounds as the first index above a positive threshold and the last index below a negative threshold (the threshold was set to avoid gradients introduced by baseline noise):

$$i_{\text{L}} = \min\{i \mid g_{\text{n}}(i) \geq \tau\}, \quad i_{\text{R}} = \max\{i \mid g_{\text{n}}(i) \leq -\tau\}, \quad \tau \approx 0.7\text{–}0.8.$$

   where $i_{\text{L/R}}$ are the left and right bounds of the ROI.

As for centering algorithm, let N be the number of acquisition points (usually 1000) and $i_{\text{C}} = (i_{\text{L}} + i_{\text{R}})/2$ the ROI center (the approximated center of the bunch). The centering error in samples is

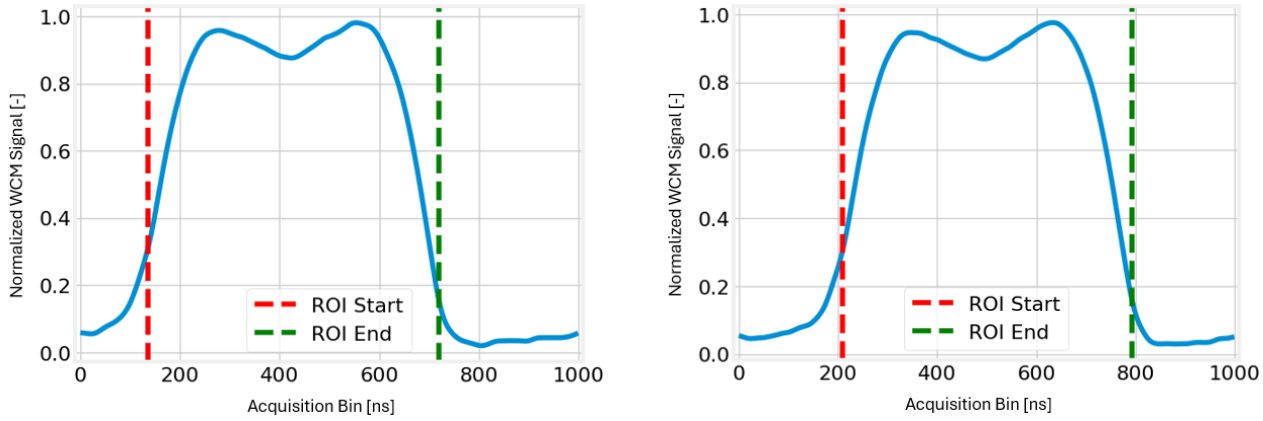
$$e \;=\; \frac{1}{2}\,(i_{\text{L}} - (N - i_{\text{R}}))\,, \tag{5.7}$$

which converts to a nanosecond trigger update by multiplying with the sampling interval (ns per point, which is usually 1 ns). The delays for the calibration burst are shifted by e (with safeguards to perform ± 500 half-window shifts when the ROI is wrapped), and acquisition is repeated until $|e|$ is small (typically < 10 samples). The beauty and robustness of gradient-based ROI detection is that it localizes the bunch edges even when the internal shape changes (e.g., when we use triple-harmonic operation), so re-centring via e consistently centres the bucket in the window. The process, as it occurs in the script, is shown in Figure 5.25 where the left and right ROI bounds can be seen, and the algorithm converges from Figure 5.25a to 5.25b in a single iteration.

Consequently, once the profiles at injection are centred with their initial delay Δt_0, per-trace delays for the entire cycle are propagated by compensating the local change of T_{rev} between traces:

$$\Delta t_i \;=\; \Delta t_0 \;-\; k_{\text{compensation}}\,(T_{\text{rev}}(t_{\text{inj}}) - T_{\text{rev}}(t_i)) \times 10^9, \tag{5.8}$$

where t_i is the C-Time of the i-th acquired trace within its burst and $k_{\text{compensation}}$ is a factor which changes depending on the ring that is acquired. For some reason that has not been identified in the CCC, the delays have to be scaled by a factor in order to properly remove the tilt in the waterfall plots. This could potentially be due to different path lengths for the acquisition signals for different cards that are somehow corrected by using an empirical factor. Also, if the sampling interval must change to keep a full bucket in view, the initial delay for the next trace is shifted by half the window difference to preserve centring. Finally, the acquisition delays for each trace are shown in the right panel of Figure 5.26, where they approximately follow the inverse of the shape of the revolution frequency ω_0 due to the relationship $T_{rev} \propto \frac{1}{\omega_0}$. An example of the delay corrected acquisition for a full cycle is shown in Figure 5.27



(a) Initial uncentred bucket ($e = 78$)

(b) Final centred bucket ($e = 1$)

Figure 5.25: Initial bucket centring process to compute the required initial delay Δt_0

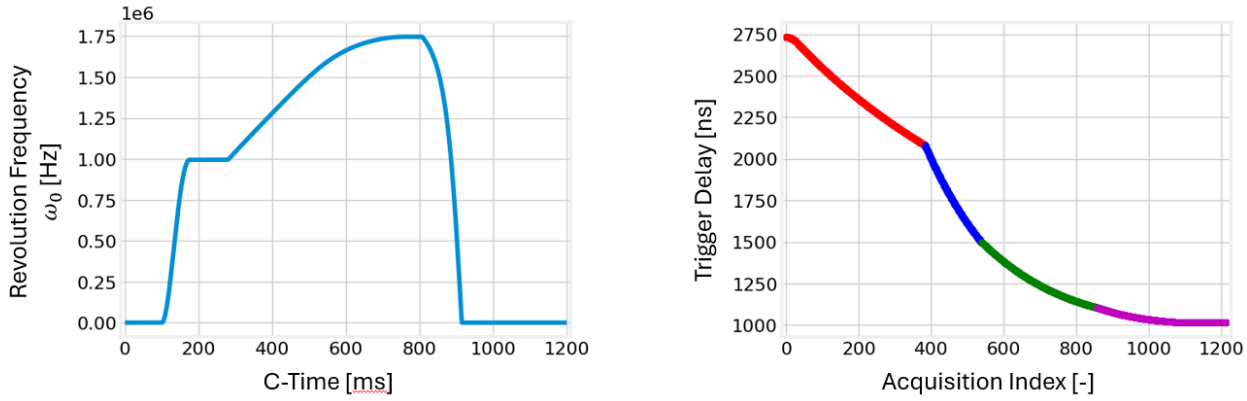


Figure 5.26: Revolution frequency ω_0 as a function of cycle time (left) and the trigger delays per trace (right). Each burst is shown in a different colour.

Profile Selection Around Computation Points

After acquiring the centred bunch profiles throughout the cycle, it is important to consider how the data will be selected for each computation point, and if it will be averaged or not.

Now onto the final selection and preprocessing of the traces acquired. Let $t_j^\star \in \mathcal{T}^*$ be the computation points, for the selection of traces to include in the inference of the RF parameters at each computation point:

- Injection mitigation: for the first computation point(s), if they are near injection, average all traces in a short window (e.g. $t_1^\star$ to $t_1^\star$+2 ms) to suppress injection transients that make the profiles have high-frequency features.
- For subsequent C-Times, select the single trace with acquisition timestamp closest to each $t_j^\star$ ("best" profile).

This is done such that averaging (and, thus mixing information) is only done where transients dominate and therefore it improves stability, while nearest-neighbour selection elsewhere preserves temporal fidelity to the computation points. Then, for the smoothing and reinterpolation:

1. We apply a smoothing spline (GCV-chosen smoothing) on the selected profiles to obtain $\tilde{p}$.
2. Using the ROI bounds, we compute the centre of the bucket and select a symmetric window of one RD period $\pm T_{rev}(t_j^\star)/2$ around the centre.
3. Then we reinterpolate onto a uniform grid of N=1000 points spanning one RF period.
4. Finally, we shift the minimum to 0 and max-normalise so that the final profile has the range $[0, 1]$.

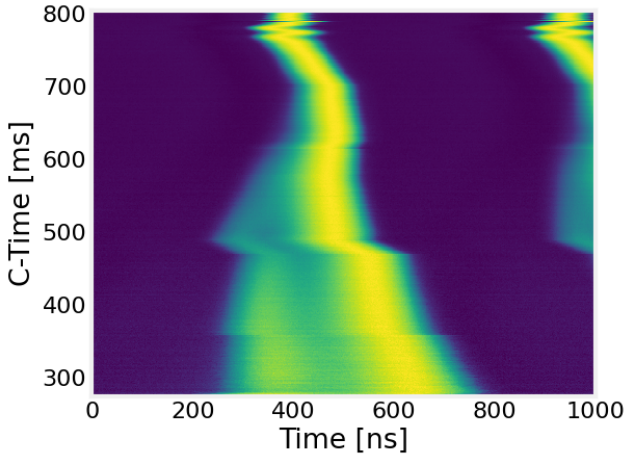
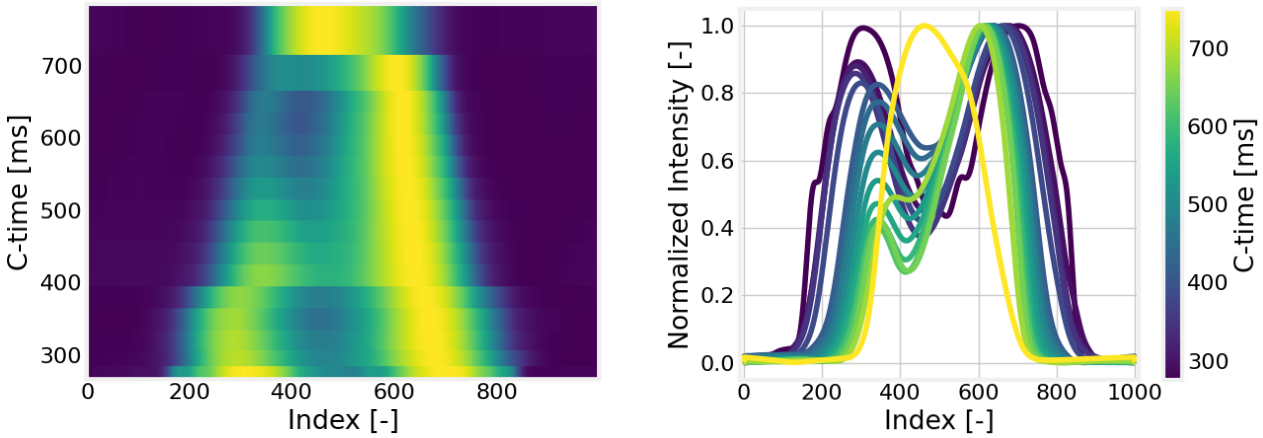


Figure 5.27: Waterfall plot showing the entire ramp acquired and delay corrected such that the acquisition window is centred.

An example of two views of the reinterpolated waterfall plot are shown in Figure 5.28. Figure 5.28a shows the waterfall plot of the centred and normalised profiles, while Figure 5.28b shows a view which is more indicative of how the model sees each C-Time.



(a) Waterfall plot

(b) Individual Profiles

Figure 5.28: Complementary views of reinterpolated profiles after profile selection/averaging

A final point to mention is the fact that the blow-up voltage V_{10} must be set to 0 during optimisation. This is because the blow-up voltage is used to generate high-frequency modulations in the bunch profile, and therefore increase the emittance ϵ and enhance Landau damping and stability. While useful and necessary for some beam types, these modulations in the bunch profile completely distort and deteriorate the quality of the profiles before and after smoothing. This effect is seen in Figure 5.29.

5.1.7. Validation Procedure: PSB Results

This section validates the proposed CBAM phase correction tool using raw longitudinal bunch profile measurements from the Proton Synchrotron Booster (PSB). The data acquisition and preprocessing procedure and the preprocessing required to match the training domain were presented previously; here we apply the trained model directly to measured longitudinal profiles and assess the outcome in the machine.

In line with the V&V methodology defined in Chapter 3, we restrict the validation to four representative operational programs: BCMS, ISOLDE, MTE, and TOF. Together, these cover the range of longitudinal manipulations achievable under double-harmonic operation in the PSB: sustained bunch lengthening (ISOLDE, TOF, BCMS), and splitting (MTE). Our evaluation emphasizes correctly phasing for bunch lengthening, the absence of oscillations, and the comparison with the manually phased solution. However, since the manually phased Φ_2 which could be affected by the intensity used, the correct phasing of the profiles was also verified for the operational beams using operational intensity to make sure the analysis presented here is still valid.

A specific point for the BCMS cycle is that two variants are in use: (i) BCMS 2024, operated with double harmonic throughout, and (ii) BCMS 2025, which employs triple-harmonic operation at injection before reverting

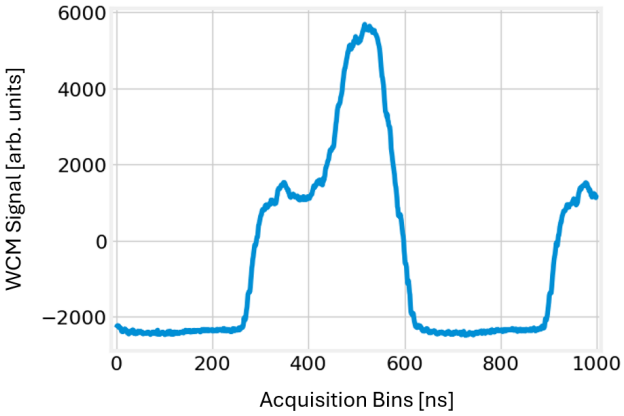


Figure 5.29: Modulations introduced in bunch profile due to blow-up voltage (h_{10})

to double harmonic later in the cycle. Both variants are validated for three reasons:

1. **Operational relevance.** BCMS is a key production beam; validating both program years ensures robustness to program updates that could occur later on. An example would be the possibility of adding triple harmonic to other cycles like TOF, and since the correct Φ_2 for BLM is independent of the addition of V_3, it can still be used to optimise Φ_2 once r_2 is roughly known.

2. **Generalization across harmonic configurations.** Comparing BCMS 2024 (pure double harmonic) with BCMS 2025 (triple at injection, double thereafter) probes the model's sensitivity to the higher voltage ratio r_2 which is required for the correct triple harmonic operation.

3. **Completeness with respect to verification.** The verification study considered only the 2025 variant; validating both cycles in the machine closes this gap and isolates any differences that could arise.

The timing requirements proposed in Chapter 3 were met by the CBAM model, as it was able to take less than 30 seconds to acquire, preprocess, infer and correct the profile. The largest proportion of the time was spent on the reinterpolation and smoothing process, which took roughly 1 second per computation point, while the inference took less than a second to perform and send the phase setting to the PSB. However, the more computation points used, the more time the process takes. This could be improved by using a more efficient smoothing (such as is done with the convolutional kernels instead of B-spline smoothing) and reinterpolation mechanism to be able to do this in a tensorized manner for computational efficiency. Regardless, since the timing requirements were met for all programs and varied Δt (ranging from 10 to 30 ms), the requirements are considered to be met. As for the methodology, for each program we follow a consistent procedure:

1. Inject protons with an intensity of 80×10^{10} ppb.

2. Acquire a reference waterfall under the current machine phasing.

3. Compute the model's Φ_2 updates only where the voltage ratio is actionable. Thus, we optimise from C-275 to C-750, except for MTE which is optimised until extraction (C-805).

4. Apply iterative corrections multiplied by a decaying weight to emulate convergence.

5. Compare initial and final waterfalls and profiles at selected C-Times against the manually optimised phase for that program.

BCMS 2025 (Triple-Harmonic at Injection)

For the BCMS 2025 validation we disable the blow-up voltage and the third harmonic voltage, so that the measured profiles are driven only by V_1 and V_2 during the test. The initial waterfall corresponding to the machine set solution is shown in Figure 5.30. In this figure, slight oscillations are visible as thin lines riding on top of the main bunch density. These features are consistent with the longitudinal shaving (and additional transverse shaving with beam stoppers) used to trim emittance. Under current phasing an asymmetry is present between C-600 and C-700, with higher density in the right lobe than in the left lobe, even though by this point V_3 is already off in the original cycle (so it is not being influenced by the fact that we disabled V_3). This region provides a clear target for correction.

Starting from the initial acquisition, we apply the first CBAM correction computed from the measured waterfall. The initial and post-correction waterfalls are compared in Figure 5.31, and the phase program after the first

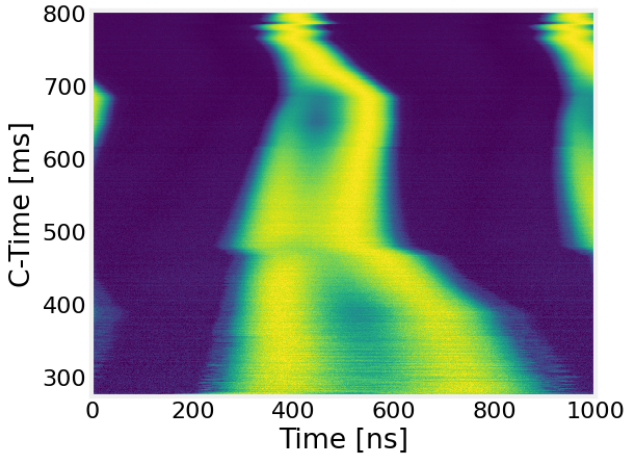
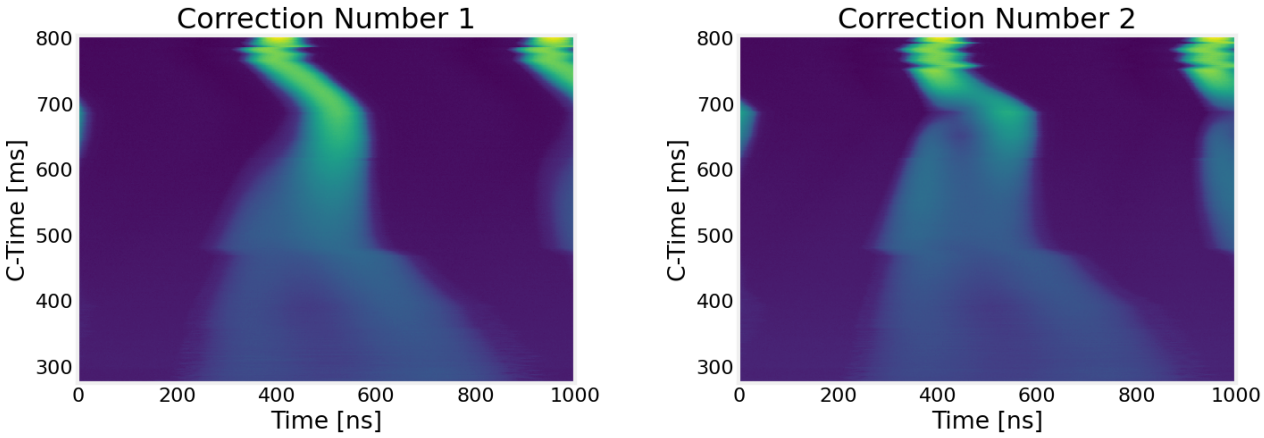


Figure 5.30: CBAM BCMS 2025: measured waterfall under manually optimised phasing

correction is in Figure 5.32. As expected (given impedance differences used in training and in the PSB), the solution does not converge after a single step, but it already moves towards reducing the asymmetry.



(a) Initial BCMS 2025 waterfall

(b) After 1st correction

Figure 5.31: CBAM BCMS 2025: initial estimate waterfall (left) and after-first-correction waterfall (right)

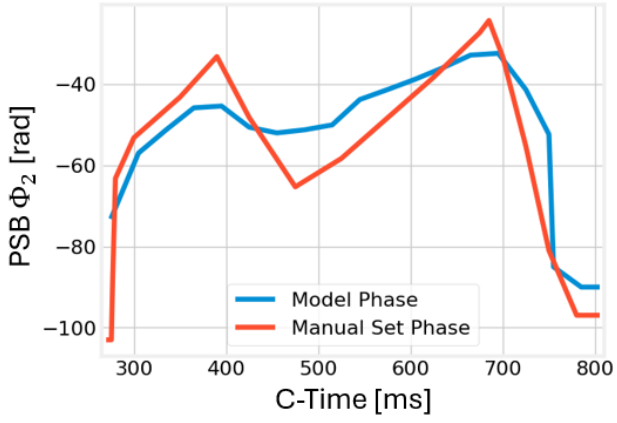


Figure 5.32: CBAM BCMS 2025: Φ_2 program after the first correction compared to the manually phased Φ_2 program

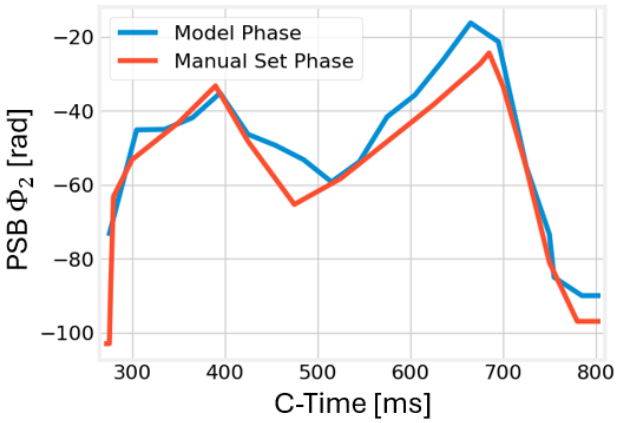


Figure 5.33: CBAM BCMS 2025: final Φ_2 program with the solution compared to the manually phased Φ_2 program

After a few iterations, the solution converges to a trajectory that differs from the machine set phase (Figure 5.33) but yields the desired profile evolution. In particular, the phase becomes larger between C-600 and C-700, consistent with compensating the observed asymmetry in the manually phased waterfall plot. The final waterfall (Figure 5.34) confirms that the left/right particle density is balanced in that region.

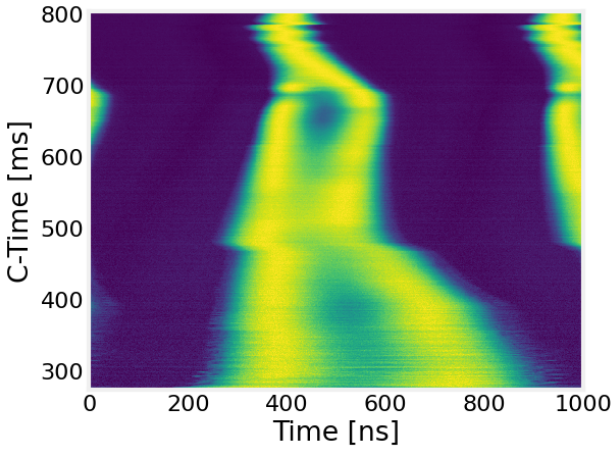


Figure 5.34: CBAM BCMS 2025: final measured waterfall after applying the CBAM solution

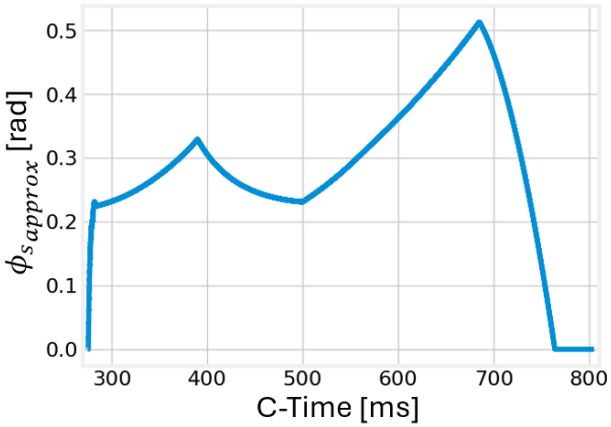
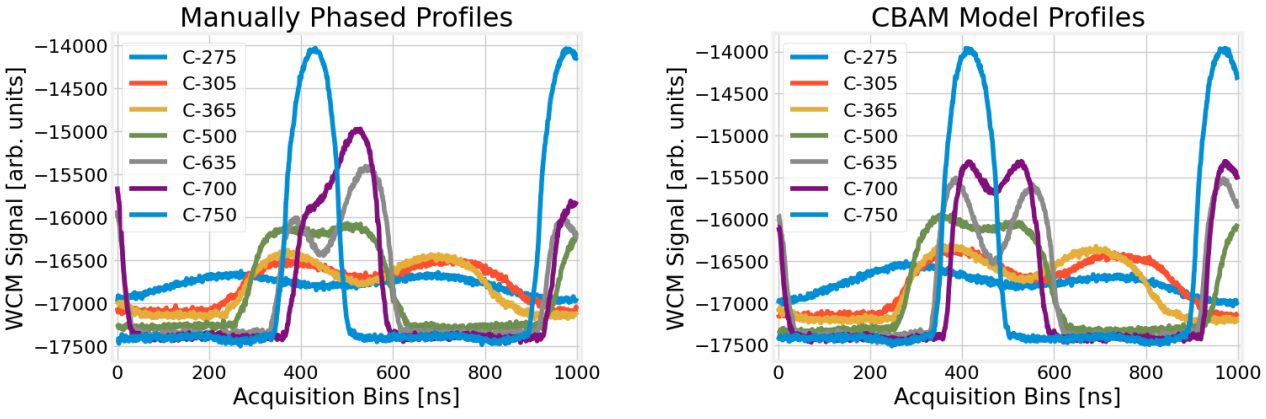


Figure 5.35: CBAM BCMS 2025: approximate synchronous phase across the cycle



(a) Manual phasing

(b) CBAM

Figure 5.36: CBAM BCMS 2025: select profile comparison at certain C-Times for the manually phased and CBAM solution

Around C-680 a small charge shift to the right lobe remains visible in the final waterfall. This coincides with the location of a kink in the manual set phase and in the approximate synchronous phase shown in Figure 5.35, which was not captured precisely by the automatic computation point selection. This suggests that the current computation point pipeline should be replaced by a more robust selector. Elsewhere, the solution is well set; small deviations around C-460 and C-660 may have the same origin, or might simply be the limit of the CBAM's accuracy with respect to phasing.

Nevertheless, to quantify the improvement, we compare acquisitions at discrete C-Times in Figure 5.36. The manually phased profiles are shown in Figure 5.36a and the CBAM-corrected profiles in Figure 5.36b. From C-500 onward the CBAM result is visibly closer to the desired shape, with better symmetry and length control across the selected time slices.

Finally, the per-iteration corrections over C-Time are shown in Figure 5.37. Distinctive oscillations appear after C-700, likely due to the combination of very low V_2 and the relatively low emittance there, which occurs due to *adiabatic damping*. This occurs when at higher energies the oscillations in phase become smaller and smaller due to the denominator in Equation 2.37 increasing, hence causing lower emittances. Furthermore, low emittances are already known to be harder to optimise for given that low emittances do not provide sufficient information on the potential well shape (see the error at low filling factors Figure 5.2). In that region, BLM is not required operationally, and the apparent convergence is only ensured by the decay weighting in the iterative scheme, which sheds light on why a fully optimised LSTM-RL agent is expected to have better performance.

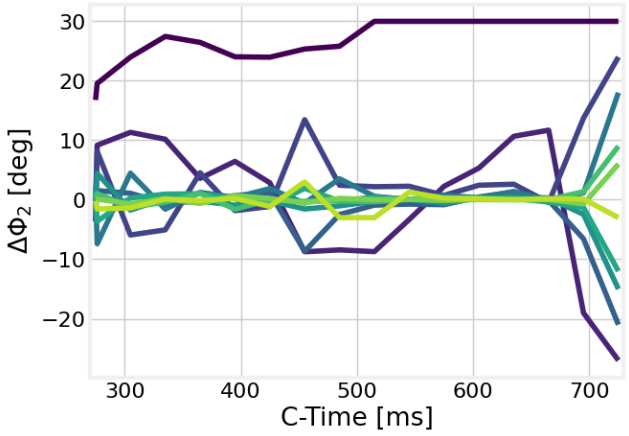


Figure 5.37: CBAM BCMS 2025: correction amplitude $\Delta\Phi_2$ per iteration across C-Time (yellow indicates corrections later in the cycle)

BCMS 2024 (Double-Harmonic)

For the BCMS 2024 validation, the blow-up voltage is disabled as in the BCMS 2025 case. The resulting phase program is notably similar to TOF, which is expected: this cycle predates triple-harmonic operation and aims at space-charge mitigation using only the second harmonic, much like TOF but at lower intensity and reduced transverse emittance, which is reduced even further with triple-harmonic operation in the 2025 variant. This is not so much of an issue for TOF as the beam is extracted after the PS and the transverse emittance doesn't grow considerably, hence why triple harmonic operation is not of primary importance there.

The optimisation performs well, and the final solution is shown in Figure 5.38. The previously mentioned sampling issue still persists: the automatic computation point selection skips C-680 even though the approximate synchronous phase changes there, so the optimiser cannot act exactly at that location. This is a preprocessing limitation rather than a shortcoming of the model and motivates replacing the current selector with a more robust scheme.

The initial and final waterfalls are shown side-by-side in Figure 5.39, where the effect of not computing at C-680 is seen again in the asymmetric profiles at that location. As in BCMS 2025, the CBAM solution improves symmetry and length control after mid-cycle. From C-540 onwards, the profiles are consistently better aligned to the target shape.

Interestingly, a faulty acquisition occurred in correction number 9, which led the optimiser to adjust the phase around C-550 away from its previously stabilized value. The issue is visible in Figure 5.40 and by comparing the two consecutive Φ_2 solutions in Figure 5.41. In the discrete profile comparison, the manually phased peaks appear to be better matched than the CBAM result at C-550. However, this is mainly attributable to the bad input in correction 9, as by correction 8 the solution had already stabilized at the same value as the manually phased solution as shown in the left panel of Figure 5.41. This underlines the sensitivity to acquisition quality, especially early in the iterative sequence: in correction 8 only 30% of the regressed value has been applied, yet the step at C-550 is still about 8°.

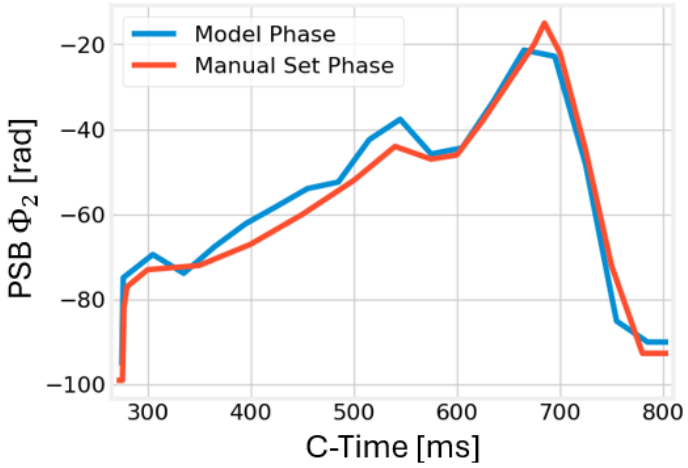
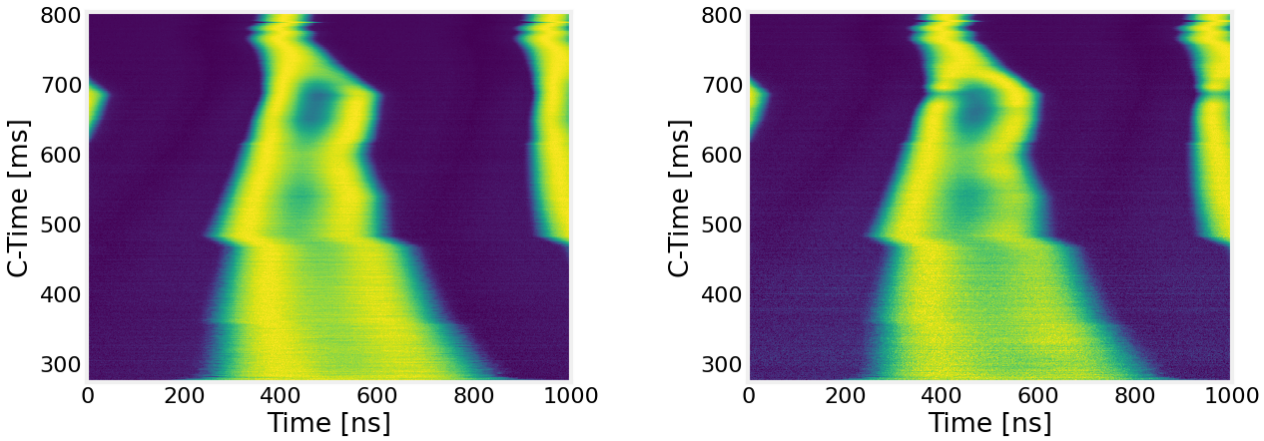


Figure 5.38: CBAM BCMS 2024: final Φ_2 program with the CBAM solution compared to the manually phased Φ_2 program



(a) Manually phased waterfall

(b) CBAM solution waterfall

Figure 5.39: CBAM BCMS 2024: manually phased solution and CBAM solution measured waterfalls shown side by side

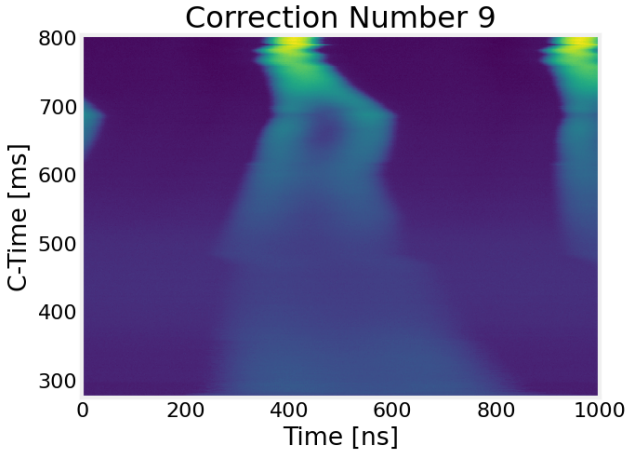
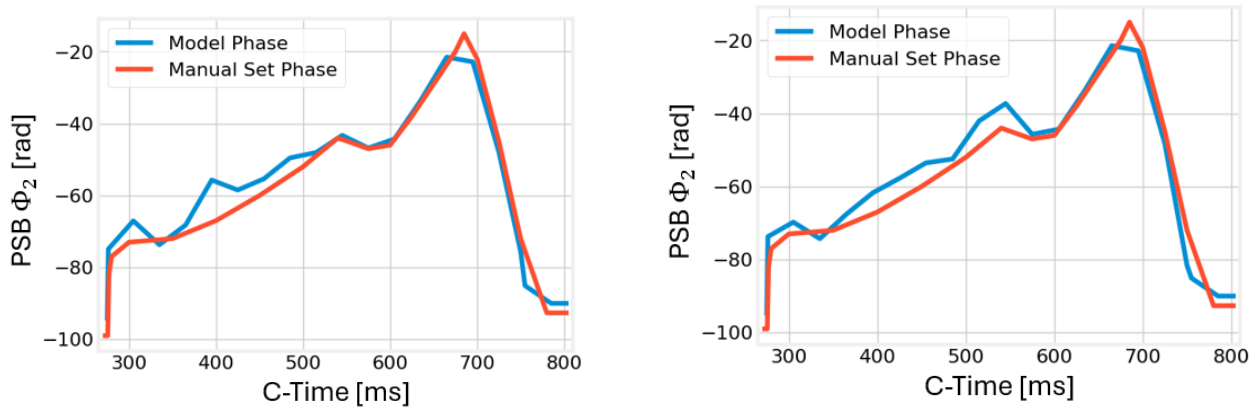


Figure 5.40: CBAM BCMS 2024: faulty acquisition displayed in waterfall plot at correction number 9

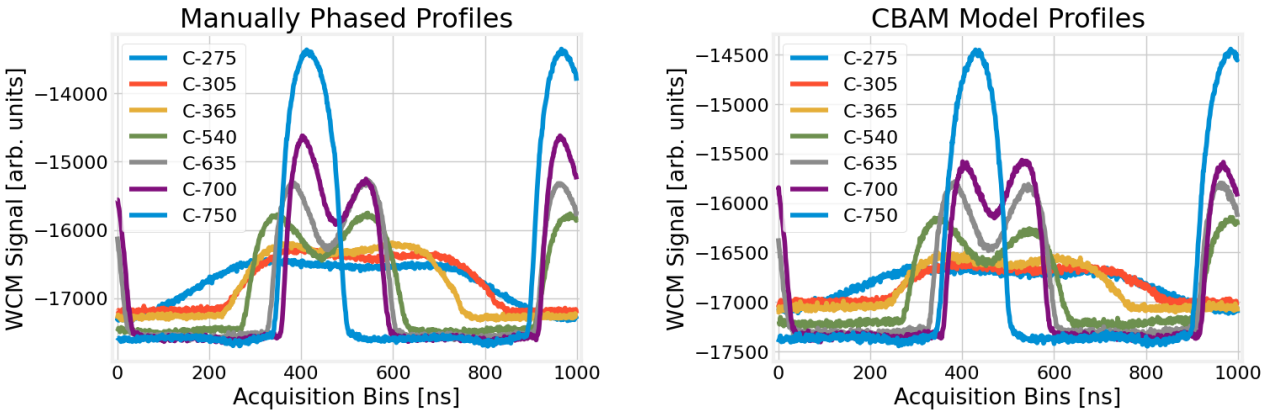


(a) CBAM solution at correction 8

(b) CBAM solution at correction 9 (faulty input)

Figure 5.41: CBAM BCMS 2024: side-by-side comparison of Φ_2 solution after correction 8 and 9

Representative manually phased profiles and CBAM-corrected profiles at selected C-Times are shown in Figure 5.42a and Figure 5.42b. Despite the faulty acquisition, the CBAM result after C-540 remains superior overall, with improved symmetry, albeit with the issue of the large change in C-680, which is bypassed by the model. Nevertheless, from injection to C-540 the manually phased profiles are slightly more symmetric than that of the CBAM model, but the model still achieves almost the perfect phasing. This could be attributed to its attempt to generalize for all emittances and its regressive behaviour.



(a) Manual phasing

(b) CBAM

Figure 5.42: CBAM BCMS 2024: select profile comparison at certain C-Times for the manually phased and CBAM solution

Finally, Figure 5.43 shows the per-iteration corrections versus C-Time. As with BCMS 2025, oscillations appear beyond C-700 and convergence there is only guaranteed by the decay weighting. This effect reoccurs in all cycles, so it will not be shown again. It is still to be investigated why this occurs after C-700, but as mentioned previously, it is likely a mixture of the voltage ratio not being high enough and low emittances near extraction.

ISOLDE

For ISOLDE, the blow-up voltage and V_3 are disabled; operationally this only affects the cycle up to C-400, after which the program runs purely in double-harmonic mode. The optimisation was carried out up to C-750. The CBAM and manually phased waterfalls are shown side by side in Figure 5.44, and the final Φ_2 program used to obtain the CBAM result is shown in Figure 5.45. Note that the machine set phase is different from that used in the verification, as the one in verification was the 2024 variant, which did not yet use triple-harmonic operation, but the one discussed here does. It is more important that the CBAM model is able to cater to this program as well given its relevance in current operations. Meanwhile, the approximate synchronous phase for this cycle is given in Figure 5.46.

Near injection, the CBAM waterfall shows a small shift in bucket location relative to the manual case. This coincides with a kink in the approximate synchronous phase, as shown in Figure 5.46, that lies just below the detection threshold of the current computation point selector and is therefore not sampled. The same undersampling occurs around C-400, where a change in the balance between energy gain and dominant voltages

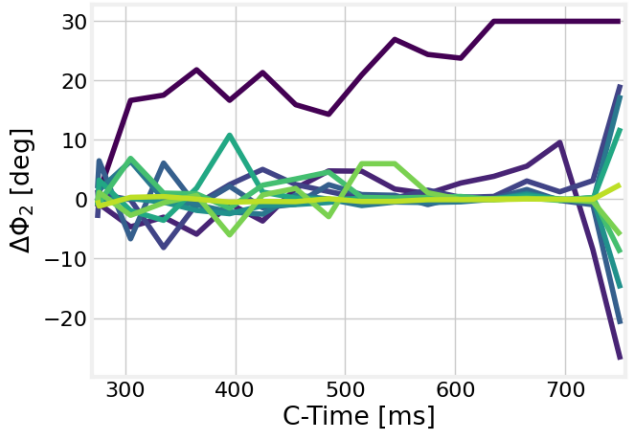
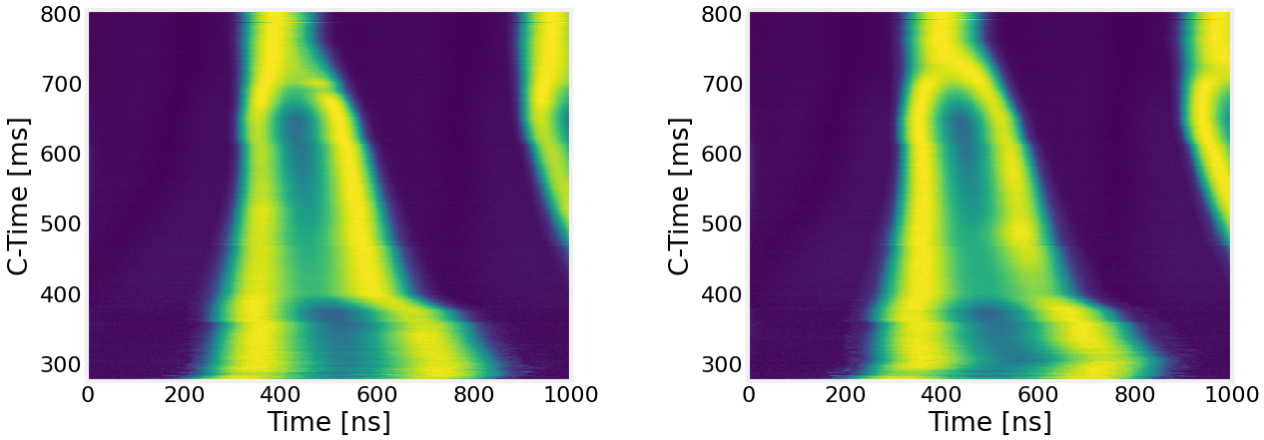


Figure 5.43: CBAM BCMS 2024: correction amplitude $\Delta\Phi_2$ per iteration across C-Time (yellow indicates corrections later in the cycle)



(a) Manually phased waterfall

(b) CBAM solution waterfall

Figure 5.44: CBAM ISOLDE: manually phased solution and CBAM solution measured waterfalls shown side by side

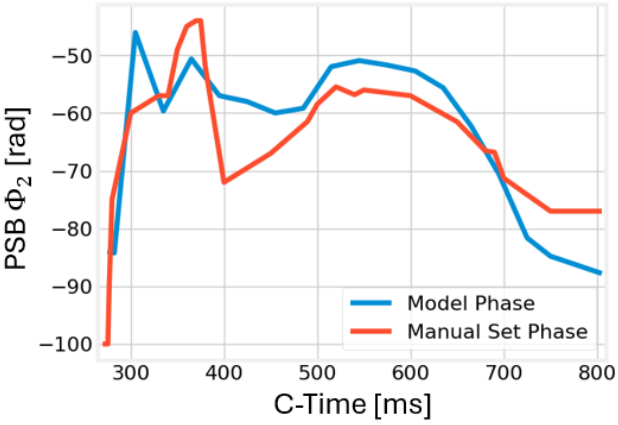


Figure 5.45: CBAM ISOLDE: final Φ_2 program with the CBAM solution compared to the manually phased Φ_2 program

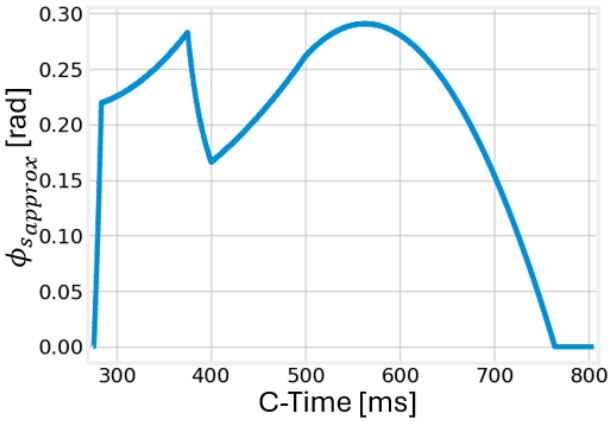
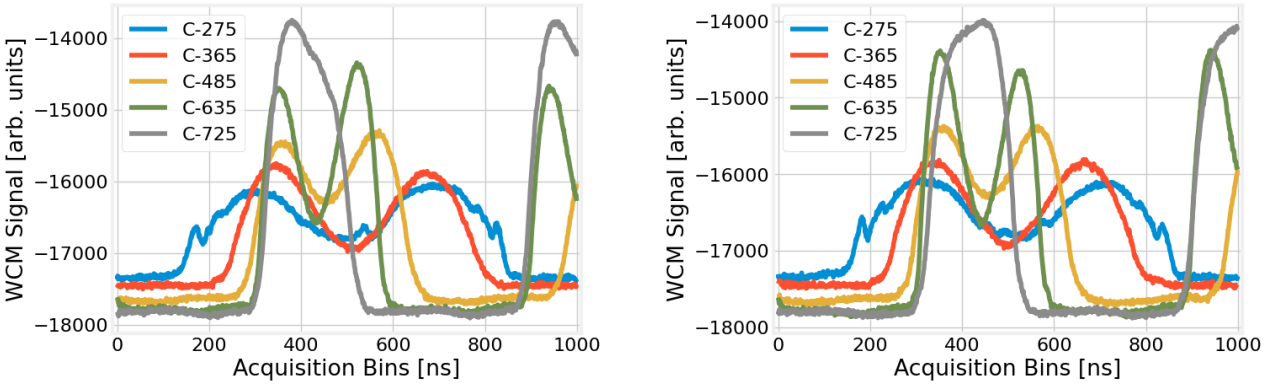


Figure 5.46: CBAM ISOLDE: approximate synchronous phase across the cycle



(a) Manual phasing

(b) CBAM

Figure 5.47: CBAM ISOLDE: select profile comparison at certain C-Times for the manually phased and CBAM solution

alters the synchronous phase, shifting the bucket again. Missing that point leads to a temporary charge imbalance as the left lobe becomes denser, and it could again be avoided with a more robust selector.

However, from C-400 to C-660, in Figure 5.44a the manually optimised case exhibits asymmetries with greater density in the right lobe. This does not occur for the CBAM solution in Figure 5.44b and is a clear improvement in the reduction of the line density. Furthermore, the discrete profile comparisons between Figure 5.47a and Figure 5.47b show that at injection (C-275) the CBAM profile is correctly phased with the peaks having the same height. However, between C-275 and approximately C-340 the CBAM result is not optimal, as shown in Figure 5.44b, and this is mainly due to the under-sampling at the critical computation points which are identified in the manually phased solution. This causes the model to basically interpolate at these critical points and not phase the profiles correctly. For the remaining C-Times, notably from C-635 onward, the CBAM profiles are more symmetric with the desired bunch lengthening.

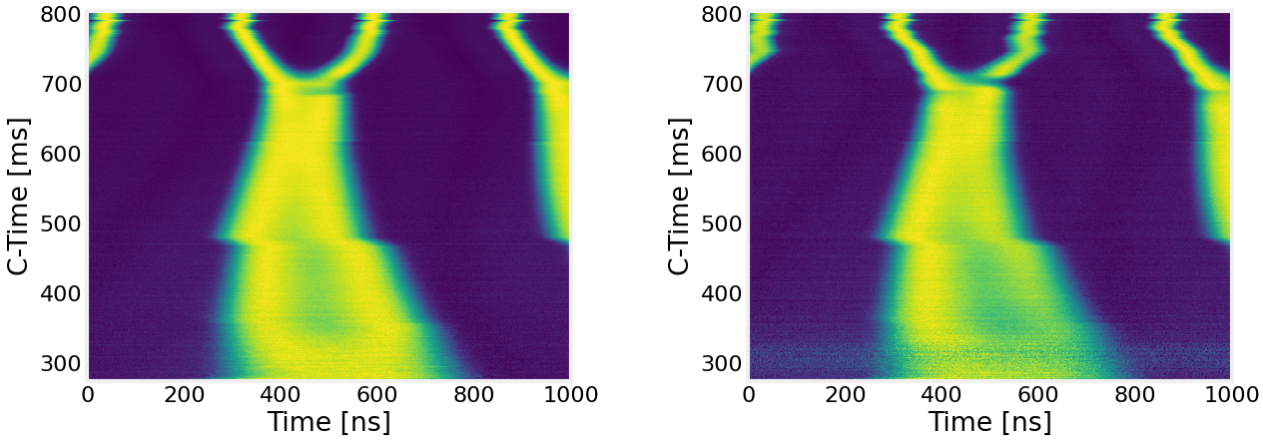
Overall, the only region where the CBAM solution is lacking is the short interval after injection up to about C-400. The most plausible causes are a combination of injection transients (whose spectral content is at frequencies lower than the noise augmentation used during training) and a slight offset between the computation points used in the manually phased solution and those used by the CBAM model. Both observations further motivate replacing the current computation-point selection by a method that is responsive to small but operationally relevant changes in synchronous phase.

MTE

For MTE we remove the blow-up voltage and validate the model during the splitting, which demands very precise second harmonic control. This case makes the impact of computation point selection most visible: although the correct phasing is achieved in a broad sense, the sampling does not align exactly with the true onset of the split, so the final two buckets do not carry identical charge. The initial (manual) and final (CBAM) waterfalls are shown side by side in Figure 5.48, and the final Φ_2 solution is reported in Figure 5.49. In this program the second harmonic phase is manually set with high precision, which is not surprising given that splitting requires the most precise control of Φ_2.

It is interesting to note that the model already interprets that a split is being formed after the first correction, as after the first correction the reinterpolated profiles show a small charge concentration consistent with the two-bucket structure, as shown in the late-cycle (yellow) reinterpolated profiles in Figure 5.50. This early recognition is helpful, but precise balance ultimately depends on sampling the onset of the split. Additionally, two noisy acquisitions (corrections 5 and 8) introduced solution oscillations up to about C-500. This degraded performance precisely where balance is most sensitive. At correction 4 the solution was already satisfactory, the Φ_2 programmes for corrections 4 and 5 are compared in Figure 5.51, and the corresponding reinterpolated profile stacks are shown in Figure 5.52. The latter makes clear that equal charge in both bunches at correction 4 becomes unbalanced at correction 5 mainly because of the faulty input. In the final waterfall in Figure 5.48b this manifests between C-300 and C-500 as a higher density in the left lobe.

A final comparison of discrete profiles indicates that the two bunches reach a similar peak density at extraction, but likely at the expense of some intensity. In the manually phased case the peak at C-800 remains higher than the peaks in C-750, whereas for the CBAM solution this is not observed. The CBAM profile at C-700 (the approximate onset of splitting) is also more asymmetric than the manual one, consistent with the presence of an additional computation point at C-710 in the manual phasing, which matches the kink in the approximate synchronous phase. The discrete profile sets are provided in Figure 5.53.



(a) Manually phased waterfall

(b) CBAM solution waterfall

Figure 5.48: CBAM MTE: manually phased solution and CBAM solution measured waterfalls shown side by side

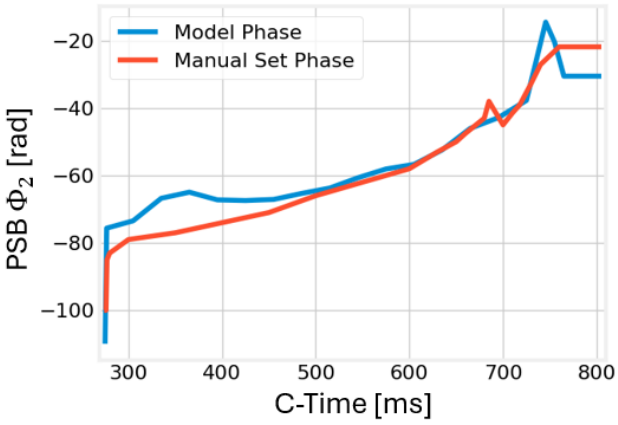


Figure 5.49: CBAM MTE: final Φ_2 program with the CBAM solution compared to the manually phased Φ_2 program

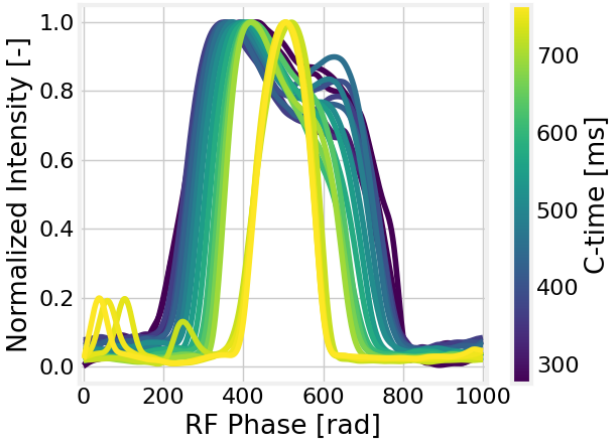
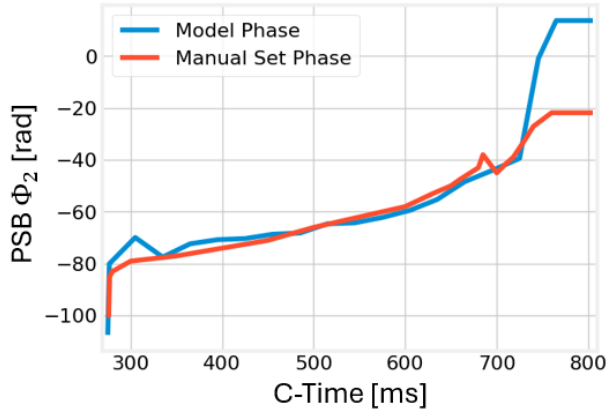
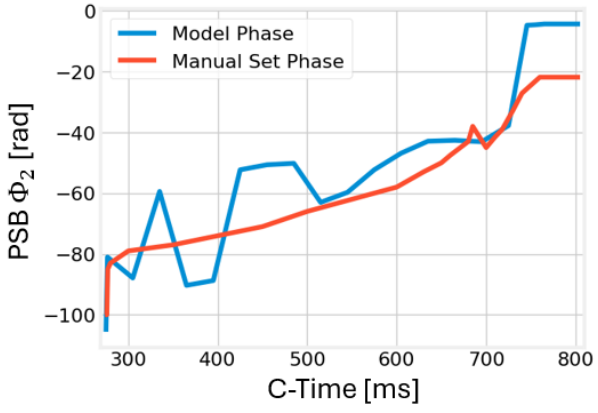


Figure 5.50: MTE: reinterpolated profiles after the first correction

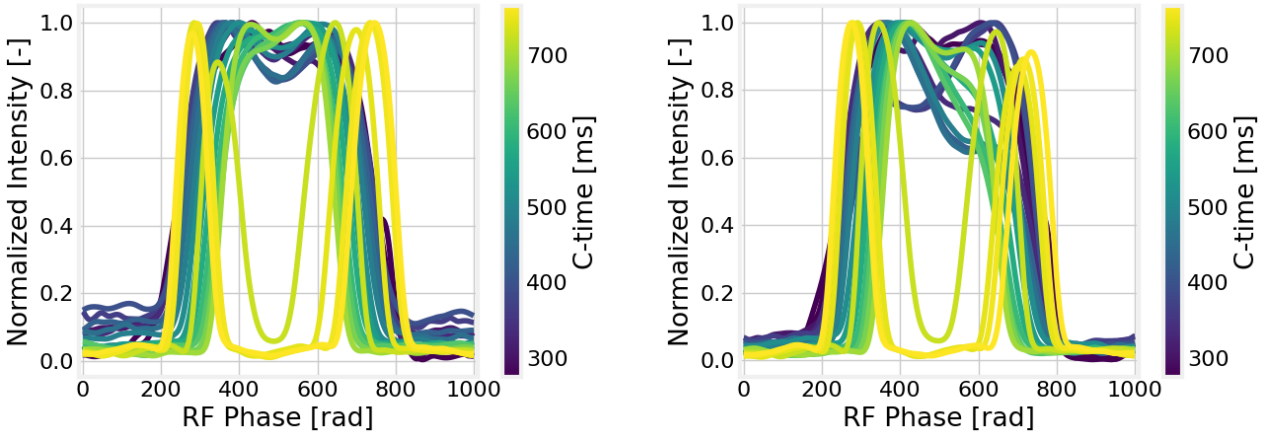


(a) Φ_2 at correction 4



(b) Φ_2 at correction 5 (bad acquisition)

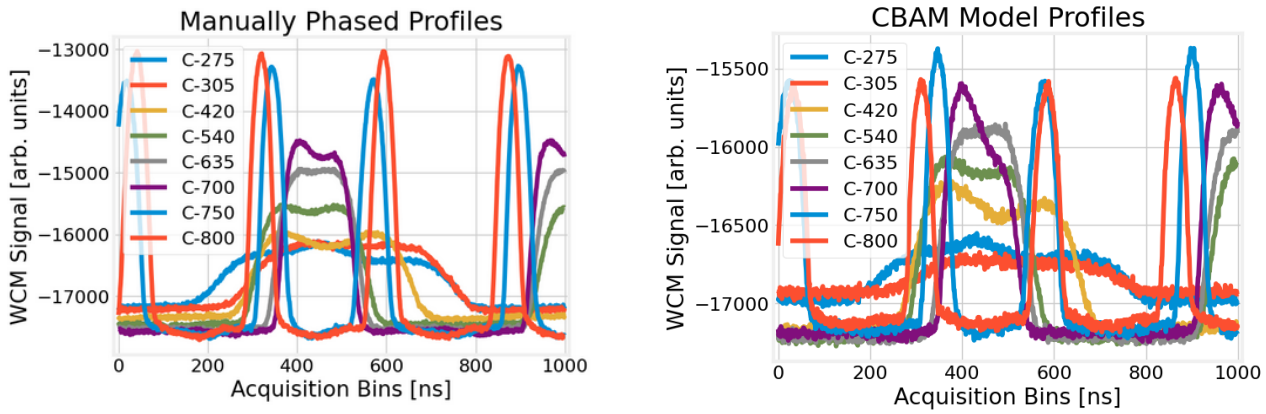
Figure 5.51: CBAM MTE: Φ_2 solutions for consecutive corrections (highlighting the perturbation caused by the faulty acquisition) compared to the manually phased Φ_2 program



(a) Reinterpolated profiles: correction 4

(b) Reinterpolated profiles: correction 5 (bad acquisition)

Figure 5.52: CBAM MTE: equal charge in both buckets at correction 4 becomes unbalanced after correction 5 due to the bad acquisition.



(a) Manual phasing

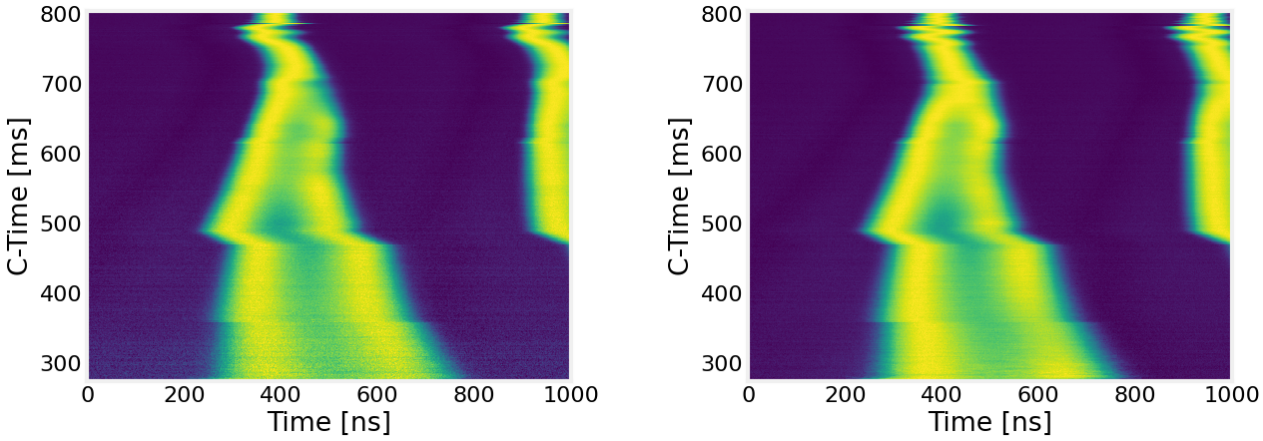
(b) CBAM

Figure 5.53: CBAM MTE: select profile comparison at certain C-Times for the manually phased and CBAM solution

In summary, even with two bad acquisitions out of ten, the model performs adequately for MTE splitting. The dominant limitation is not the phase precision, but the alignment of the computation points to the true onset of the split, together with robustness to faulty acquisitions. The latter can be mitigated with testing and ensuring that the acquisition hardware is performing as expected, while the computation point issues can be mitigated with a more resilient sampling algorithm. Nevertheless, the performance is acceptable given that splitting is the most demanding operation in terms of phase accuracy.

TOF

For TOF, the blow-up voltage is disabled. The manually phased solution is well aligned primarily up to C-500, where space-charge reduction is most critical; beyond this point the manual corrections begin to lose precision. This behaviour is visible in the side-by-side waterfalls of Figure 5.54 for the manually phased Φ_2 in the left panel and in the CBAM solution waterfall in the right panel. The final Φ_2 program used for the CBAM solution is provided in Figure 5.55.



(a) Manually phased waterfall

(b) CBAM solution waterfall

Figure 5.54: CBAM TOF: manually phased solution and CBAM solution measured waterfalls shown side by side

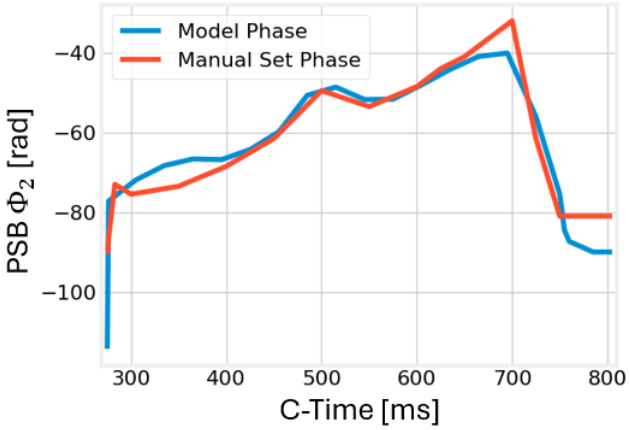
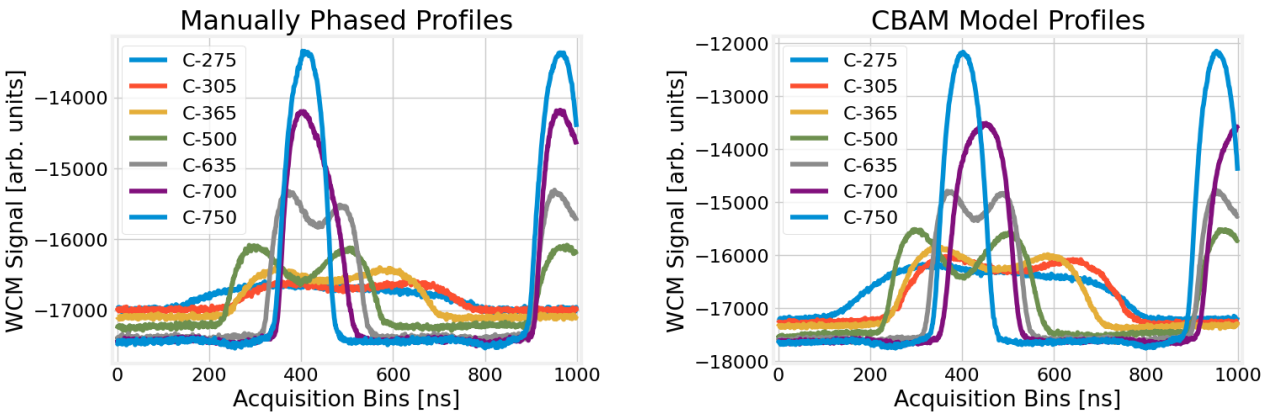


Figure 5.55: CBAM TOF: final Φ_2 program with the CBAM solution compared to the manually phased Φ_2 program

A plausible explanation for the reduced precision of the manual phasing after C-500 is that short, high-intensity bunches induce sufficient beam loading to shift the optimal bunch-lengthening phase towards higher absolute values. This interpretation is specific to TOF and is supported by acquisitions taken at operational intensities (approximately ten times higher than those used here), for which the same profiles appear correctly phased. However, it is important to note this marked intensity dependence was not observed for the other cycles, likely due to the lower intensities being used in those cases operationally and the cycles not being optimised with the utmost precision that is capable with an automated solution.

Furthermore, the CBAM solution deviates slightly from the manual phasing between C-300 and C-390, and at injection it sets Φ_2 roughly 20° lower. As shown by the discrete profile comparisons in Figure 5.56a and 5.56b, the impact of these differences on the profile shape is minimal in that region due to the low acceleration rates, but it still means slightly higher peak differences and not total symmetry.

Later in the cycle, the CBAM-corrected profiles show clear improvements, with sustained bunch lengthening



(a) Manual phasing **(b)** CBAM

Figure 5.56: CBAM TOF: select profile comparison at certain C-Times for the manually phased and CBAM solution

and good symmetry. The localized underperformance from C-300 to C-390 is consistent with the regressive nature of the model, which lacks explicit convergence behaviour and correction consistency. This likely also contributes to the mild overestimation observed in BCMS 2024 where the manually set phases are similar. Overall, TOF still achieves BLM across the cycle with small peak differences early on, and the model provides roughly correct phases in some regions and fully correct phases in others. Further gains are expected from the improved convergence logic of the LSTM-TD3 RL agent.

Concluding Remarks

In conclusion, the CBAM model is able to improve upon the manually set phase in general at computation points later in the cycle, but it still suffers from the fact that it is regressive in nature and does not consider the actions that it is taking and their effect on the observed bunch profiles. The adaptation from a 1 shot correction in the training of the model, to a decaying-corrections scheme, has helped to improve the performance of the model, but it is still not able to converge to the optimal phase in all the cycles. This is mainly due to the disconnect between the training and usage objectives of the model, which make it difficult to optimise the network to reach the desired phase in every circumstance. Additionally, the lack of generality and robustness of the computation point selection strategy causes important points where the bucket shape is shifting considerably to not be sampled, which ultimately causes the model to interpolate in this region which can cause an incorrect phase to be used at a critical point as occurred in all the cycles. This was most critical for the bunch splitting operations in MTE, but even then a satisfactory level of phasing is still achieved when observing the final waterfall plots even in the presence of disturbances caused by the acquisition hardware. This performance is expected to improve for the memory-augmented RL agent, whose training and results will now be discussed.

5.2. LSTM-TD3 RL Agent

This section introduces the recurrent RL agent based on TD3 with LSTM policies to give the controller with short-term memory of the profiles it is correcting and of the actions it has taken. The motivation for this architecture was extensively discussed during Chapter 4, which is to facilitate convergence and decouple the agent from the regressive nature of the CBAM model into a more adaptive data-driven approach that does not require correction decay as was implemented in Section 5.1.7. Recurrent policies can integrate information across successive acquisitions and actions to form a more robust interpretation of the system state and use it to act in a more precise fashion.

We therefore follow the same structure as for the supervised models: we describe the training procedure and hyperparameter optimisation, then present verification in simulation followed by validation in the PSB. The data acquisition and preprocessing is not discussed since it uses the same framework as for the CBAM model, as there was not enough time between tests to implement the more robust computation point selector. Throughout, we compare the recurrent TD3 (LSTM-TD3) against the previously established baselines and emphasize profile bunch-length control, symmetry, and stability of successive corrections. Furthermore, we begin by briefly discussing the TD3 agent without memory augmentation and mention its limitations which are to be solved by its successor.

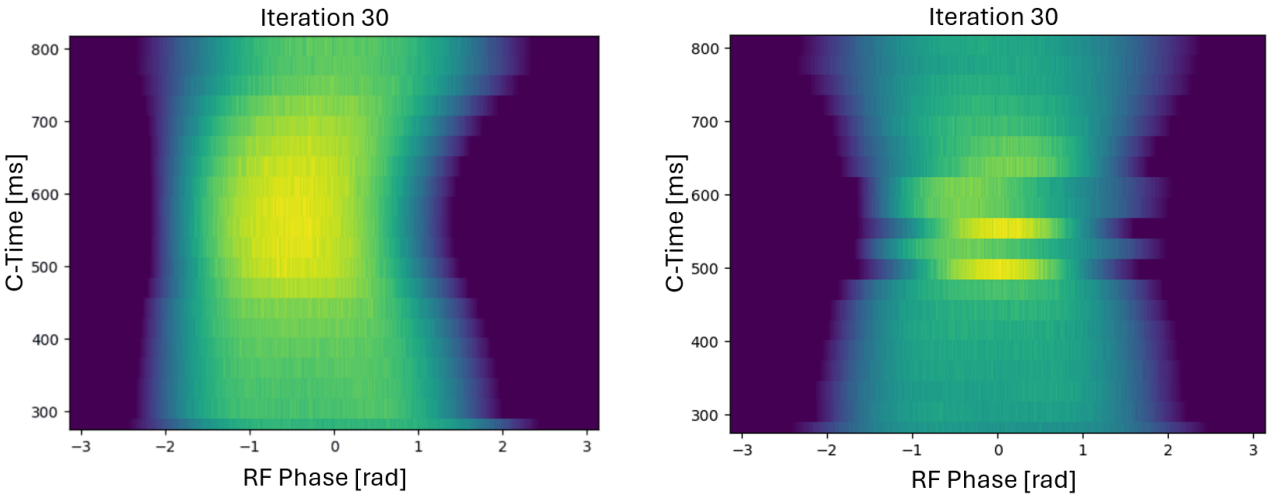
5.2.1. TD3 RL Agent Performance

A non-recurrent TD3 agent (feed-forward actor/critic) was first implemented as a reference. In practice, this agent achieved the same performance as the MLP variant of the supervised network presented previously, but of course with slower convergence to the solution as actions are bounded to a restricted Φ_2 range for the TD3 agent. The policy repeatedly learned to produce corrections of similar magnitude and sign to the MLP solution, differing only by the specific MLP architecture used in the TD3 actor, which was the standard architecture

given by Stable-Baselines3 (the framework in which this was tested) and is also the one recommended by the original TD3 paper: a network with 2 hidden layers of sizes 400 and 300 [52, 22]. Although still providing more expressiveness by having more parameters than that of the optimised CBAM model (MLP only uses FC1 and FC2 widths), it offered no tangible gain in profile quality or convergence. It should be noted that each training episode is comprised of 30 steps, meaning 30 actions being taken before the parameters that define the optimal phase $(V_{total}, r_2, \dot{\mathcal{B}}) : \Phi_2(BLM)$ are reshuffled from those available in the dataset. This is general for all the RL agents that are trained for the Φ_2 optimisation task, but this will change in the triple-harmonic optimisation.

In addition to the lack of performance gain, this feed-forward TD3 was substantially more expensive than supervised learning and required multiple restarts due to RL instabilities. In particular, the critics tended to overestimate Q-values in regions of sparse or noisy reward, leading to policy drift and the need for heavy target smoothing and conservative exploration. This also set the stage for what was likely to happen with the LSTM-TD3 architecture given that these instabilities are more related to the architecture itself rather than the reward function implemented, which is shown in Algorithm 7 (it will be explained in more detail in subsequent subsections). Even with careful tuning, the learned corrections often oscillated around certain C-Times, especially when the impedance was increased. This behaviour mirrors the MLP-with-decay baseline: once the bounded actions saturate, the agent alternates updates rather than converging smoothly.

The lack of memory also prevented the feed-forward agent from generalizing to modified impedance conditions at high energy-gain regions (roughly C-500 to C-650). When evaluated with a doubled Finemet impedance (real and imaginary parts), the solution begins to oscillate and fails to stabilize. The waterfall plots in Figure 5.57 illustrate this: at nominal impedance the profiles remain lengthened and smoothly transition even at high energy gain regions, whereas at 2× impedance asymmetric profiles start to occur, and the agent's actions oscillate leading to a Φ_2 that is above and below the ideal phasing. This is very dangerous considering this is the idealized simulation environment, which makes the agent unfit for safe implementation in the PSB.



(a) Nominal impedance

(b) Impedance ×2

Figure 5.57: Feed-forward TD3 at iteration 30: nominal versus doubled-impedance behaviour for the ISOLDE cycle

Thus, because the TD3 agent matches the MLP baseline in accuracy, requires much longer training, is sensitive to Q-value overestimation and training instability, and does not generalize under impedance shifts or at high energy gains, we do not discuss it further in the remainder of this chapter. The focus is instead on the recurrent LSTM–TD3 agent, which explicitly incorporates memory to mitigate these limitations and leverage temporal correlations across successive acquisitions and actions.

5.2.2. Training

General Considerations

For all the training procedures in the 1D Φ_2 optimisation, the produced when finding the optimal phases $(V_{total}, r_2, \dot{\mathcal{B}}) : \Phi_2(BLM)$ is used. Thanks to the parallelization that was implemented and the individual RNGs for each process, it is possible to shuffle the order of this mapping to randomly select the parameters that define the optimal phase. This allows for domain randomization such that at each step the RL agent is gaining sufficiently varied information across each asynchronous, parallel interaction with environment and from different parameter combinations.

The missing parameter here is the filling factor FF, and the way it is incorporated is by using the same discrete filling factors as in the dataset generation (see Table 4.1). However, each time the parameter combination is

chosen, a random filling factor is selected from a list that keeps track of the remaining filling factors available for this parameter combination. Once the episode is finished, this filling factor is removed from the list for this parameter combination, so that all filling factors can be sampled in a randomized way. Additionally, during one training episode for each parallelized environment, the profiles are enhanced or not with the baseline droop shown in Figure 4.7 randomly. As a result of this, the RL agent is getting and storing transitions in the most varied form possible, such that the sample efficiency is improved and the epoch reward curves are as indicative as possible of true performance across sufficient parameter combinations. This is important for the hyperparameter combination because, as explained in Section 4.8, the mean epoch return and the standard deviation of the epoch return is used as the objective value. For reference, when the epoch return is mentioned, it is referring to the return (meaning the reward of each step summed over the entire episode) averaged out across the n_{env} environments that are parallelized and interacted with at each training epoch.

Aside from these general behaviours of the training procedure, the main component that will influence the behaviour that is learned by the LSTM-RL agent (under ideal learning conditions, as we will see) is the reward function that is used for the agent to understand what is a good or bad action. As a result, it is important to define the convergence criteria that one would want through positive reinforcement when this is achieved. However, during the training iterations, two reward functions were used, an initial one and a simplified one. First, the initial reward function is presented.

Initial Reward Function

The initial reward function that was devised is shown algorithmically in Algorithm 7.

Algorithm 7 Reward for Double Harmonic RL Step ($\Delta\Phi_2^*$ is the amount of degrees to the optimal phase and s is the step number)

function 2HREWARD($\Phi_2^{\text{current}}, \Phi_2^{\text{optimal}}, \text{action}, \text{action}^{\text{prev}}, \Delta\Phi_2^{*\text{prev}}, \Delta\Phi_2^*, \text{failed}, s$)
  $\mathcal{R} \leftarrow 0$
  ▷ Phase 1: Smart adjustment bonus (before phase update)
  if $|\Delta\Phi_2^*| \leq 5°$ **and** $|\text{action} + \Delta\Phi_2^*| \leq |\Delta\Phi_2^*|$ **and** $|\text{action} + \Delta\Phi_2^*| \geq 0.5 \cdot |\Delta\Phi_2^*|$ **then**
    $\mathcal{R} \leftarrow \mathcal{R} + \frac{C_{\text{smart}}}{(1+||\Delta\Phi_2^*|-|\text{action}|)\cdot(s+1)}$ ▷ Decrease the reward with the step number
  end if
  $\Phi_2^{\text{new}} \leftarrow \text{PhaseWrap}(\Phi_2^{\text{current}} + \text{action})$
  $\Delta\Phi_2^{*\text{new}} \leftarrow \text{CyclicDistance}(\Phi_2^{\text{optimal}}, \Phi_2^{\text{new}})$
  ▷ Phase 2: Simulation penalty if matching fails
  $\mathcal{R} \leftarrow \mathcal{R} - 10 \cdot \mathbb{1}[\text{failed}]$
  ▷ Phase 3: Base Penalty to minimize $\Delta\Phi_2^*$
  $\mathcal{R} \leftarrow \mathcal{R} - C_{\text{dist}} \cdot \frac{\Delta\Phi_2^{*\text{new}}}{\pi}$
  ▷ Phase 4: Progress reward
  $\text{progress} \leftarrow \frac{|\Delta\Phi_2^{*\text{prev}}| - |\Delta\Phi_2^{*\text{new}}|}{\pi/4}$
  $\mathcal{R} \leftarrow \mathcal{R} + C_{\text{prog}} \cdot \text{progress}$
  ▷ Phase 5: Convergence bonus
  if $|\Delta\Phi_2^{*\text{new}}| < 0.5°$ **then**
    if steps_in_convergence > 0 **then**
      $\mathcal{R} \leftarrow \mathcal{R} + C_{\text{conv}} \cdot \exp(-|\Delta\Phi_2^{*\text{new}}|)$ ▷ Full bonus for staying
    else
      $\mathcal{R} \leftarrow \mathcal{R} + \frac{C_{\text{conv}}}{3} \cdot \exp(-|\Delta\Phi_2^{*\text{new}}|)$ ▷ Reduced bonus for arriving
    end if
  end if
  ▷ Phase 6: Oscillation penalty if the action is oscillating for no reason
  if $\text{action}^{\text{prev}} \neq \text{None}$ **and** $\text{sign}(\text{action}) \neq \text{sign}(\text{action}^{\text{prev}})$ **and** $\text{sign}(\text{action}) \neq \text{sign}(\Delta\Phi_2^{*\text{prev}})$ **then**
    ▷ Only relevant if far away from the optimum and large actions
    $\mathcal{R} \leftarrow \mathcal{R} - C_{\text{osc}} \cdot |\Delta\Phi_2^{*\text{new}}| \cdot \frac{|\text{action} - \text{action}^{\text{prev}}|}{\pi/4}$
  end if
  return $\mathcal{R}$
end function

The initial reward function $\mathcal{R}$ consists of six distinct phases, each addressing specific aspects of the optimisation problem:

$$\mathcal{R} = \mathcal{R}_{\text{smart}} + \mathcal{R}_{\text{sim}} + \mathcal{R}_{\text{dist}} + \mathcal{R}_{\text{prog}} + \mathcal{R}_{\text{conv}} + \mathcal{R}_{\text{osc}} \tag{5.9}$$

1. **Smart Adjustment Incentive**:

$$\mathcal{R}_{\text{smart}} = \begin{cases} \frac{C_{\text{smart}}}{(1+||\Delta\Phi_2^*|-|\text{action}||\cdot(s+1))} & \text{if conditions met} \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

   Encourages appropriately-sized actions (less than the distance to the optimum but more than 50% of it) when close to convergence ($|\Delta\Phi_2^*| \leq 5$). The step-dependent denominator $(s+1)$ creates diminishing returns, promoting faster convergence and preventing the exploitation of this bonus by staying close to convergence and not converging.

2. **Simulation Robustness**: $\mathcal{R}_{\text{sim}} = -10 \cdot \mathbb{1}[\text{failed}]$ provides strong negative feedback when physics simulations fail, ensuring the agent avoids corrections that lead to not having a bucket area (meaning loss of all particles).

3. **Distance Minimization**: $\mathcal{R}_{\text{dist}} = -C_{\text{dist}} \cdot \frac{|\Delta\Phi_2^*|}{\pi}$ provides continuous guidance toward the optimal phase. Normalization by π keeps the penalty bounded, while allowing coefficient tuning for relative importance in the total step reward.

4. **Progress Reinforcement**: $\mathcal{R}_{\text{prog}} = C_{\text{prog}} \cdot \frac{|\Delta\Phi_2^{*\text{prev}}|-|\Delta\Phi_2^*|}{\pi/4}$ directly rewards improvements in phase accuracy, providing immediate feedback on action quality to distinguish beneficial from detrimental actions even when far from the target.

5. **Adaptive Convergence Bonus**:

$$\mathcal{R}_{\text{conv}} = \begin{cases} C_{\text{conv}} \cdot \exp(-|\Delta\Phi_2^*|) & \text{if staying in convergence} \\ \frac{C_{\text{conv}}}{3} \cdot \exp(-|\Delta\Phi_2^*|) & \text{if newly arriving} \\ 0 & \text{if not converged} \end{cases} \tag{5.11}$$

   Differentiates between reaching and maintaining convergence. The exponential decay ensures that more precise convergence receives proportionally higher rewards, essential for the sub-degree precision required for low emittance beams.

6. **Oscillation Prevention**: $\mathcal{R}_{\text{osc}} = -C_{\text{osc}} \cdot |\Delta\Phi_2^*| \cdot \frac{|\text{action}-\text{action}^{\text{prev}}|}{\pi/4}$ penalizes oscillatory behaviour when the agent changes the direction of action without justification based on phase error. Multiplication by $|\Delta\Phi_2^*|$ ensures oscillations far from the target are penalized more heavily than minor adjustments near convergence.

This reward structure promotes convergence through several mechanisms: gradient consistency from combined distance penalty and progress reward, stability near convergence through smart adjustment and adaptive bonuses, and oscillation suppression targeting the common RL control instability such that faster learning is achieved. The design is also suitable for Φ_2 phasing because it respects physical constraints through simulation penalties, addresses precision requirements via the exponential convergence bonuses, and provides robustness to baseline droop through the random augmentation procedure.

Under adequate exploration and function approximation, the reward function satisfies convergence conditions for policy gradient methods: bounded rewards for stable gradients, consistent directional information, and sufficient discrimination between policies even when near optimum phasing. These properties, combined with oscillation prevention and precision mechanisms, make it particularly effective for second harmonic phase optimisation in particle accelerators.

However, because $\mathcal{R}_{\text{smart}}$ and $\mathcal{R}_{\text{conv}}$ is dependent on previous behaviours and the amount of steps that have been completed, it could lead to difficulties in modelling the Q-values, regardless of the fact that the critic networks also have memory. This, combined with the fact that Q-networks are famous for over-estimating the discounted return, and that this directly affects the loss of the policy, could lead to instabilities in the training process. As a result, a more simplified and equally as bounded reward function was also considered.

Simplified Reward Function

To address the potential training instabilities caused by the history-dependent components in the advanced reward function, a simplified approach was developed that maintains the essential guidance properties while eliminating problematic dependencies on previous states, step counts, and relative importance coefficients. This reward function is shown in Algorithm 8

Algorithm 8 Simple Reward for Double Harmonic RL Step ($\Delta\Phi_2^*$ is the amount of degrees to the optimal phase)

function 2HSimpleReward($\Phi_2^{\text{current}}, \Phi_2^{\text{optimal}}, \text{action}, \Delta\Phi_2^{*\text{prev}}, \text{failed}$)
  $\mathcal{R} \leftarrow 0$
  $\Phi_2^{\text{new}} \leftarrow \text{PhaseWrap}(\Phi_2^{\text{current}} + \text{action})$
  $\Delta\Phi_2^{*\text{new}} \leftarrow \text{CyclicDistance}(\Phi_2^{\text{optimal}}, \Phi_2^{\text{new}})$
  ▷ Phase 1: Simulation penalty if matching fails
  $\mathcal{R} \leftarrow \mathcal{R} - 10 \cdot \mathbb{1}[\text{failed}]$
  ▷ Phase 2: Base penalty to minimize $\Delta\Phi_2^*$
  $\mathcal{R} \leftarrow \mathcal{R} - \frac{|\Delta\Phi_2^{*\text{new}}|}{\pi}$ ▷ Direct distance penalty [-1, 0]
  ▷ Phase 3: Progress reward
  $\text{progress} \leftarrow \frac{|\Delta\Phi_2^{*\text{prev}}| - |\Delta\Phi_2^{*\text{new}}|}{\pi/4}$
  $\mathcal{R} \leftarrow \mathcal{R} + 2.0 \cdot \text{progress}$ ▷ Fixed coefficient progress reward
  ▷ Phase 4: Near-convergence bonus
  if $|\Delta\Phi_2^{*\text{new}}| < 0.5°$ **then**
    $\mathcal{R} \leftarrow \mathcal{R} + 3.0$ ▷ Fixed convergence bonus
  end if
  ▷ Phase 5: Action cost to prevent oscillation
  $\mathcal{R} \leftarrow \mathcal{R} - 0.1 \cdot \frac{|\text{action}|}{\pi/4}$ ▷ Simple action penalty
  return $\mathcal{R}$ ▷ Total reward [-13.1, 5]
end function

The simplified reward function consists of five core components:

$$\mathcal{R}_{\text{simple}} = \mathcal{R}_{\text{sim}} + \mathcal{R}_{\text{dist}} + \mathcal{R}_{\text{prog}} + \mathcal{R}_{\text{conv}} + \mathcal{R}_{\text{action}} \tag{5.12}$$

1. **Simulation Robustness**: $\mathcal{R}_{\text{sim}} = -10 \cdot \mathbb{1}[\text{failed}]$ remains unchanged, providing essential feedback to avoid complete particle loss.

2. **Direct Distance Penalty**: $\mathcal{R}_{\text{dist}} = -\frac{|\Delta\Phi_2^*|}{\pi}$ now uses a fixed coefficient of 1.0, eliminating the need for hyperparameter tuning while maintaining bounded penalties in the range $[-1, 0]$.

3. **Progress Reward**: $\mathcal{R}_{\text{prog}} = 2.0 \cdot \frac{|\Delta\Phi_2^{*\text{prev}}| - |\Delta\Phi_2^*|}{\pi/4}$ provides immediate feedback on action quality without scaling, using only the current and previous phase errors which are part of the observable state.

4. **Binary Convergence Bonus**:
$$\mathcal{R}_{\text{conv}} = \begin{cases} 3.0 & \text{if } |\Delta\Phi_2^*| < 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{5.13}$$
Eliminates the exponential decay and step-counting mechanism, providing a simple binary signal to inform the Q-networks when a step is taken that leads to near-optimal performance.

5. **Action Regularization**: $\mathcal{R}_{\text{action}} = -0.1 \cdot \frac{|\text{action}|}{\pi/4}$ applies a gentle penalty proportional to action magnitude, discouraging unnecessary large adjustments without the complex oscillation detection logic.

Advantages for Q-Learning Stability:

The simplified reward function addresses several key issues that can destabilize actor-critic training:

Markovian Property: By eliminating dependencies on step counts and convergence history, the reward function becomes truly Markovian, depending only on the current state, action, and immediate next state. This ensures that Q-value estimates are based solely on observable information, reducing the complexity of the value function approximation task, and also allowing the Q-networks to focus more on the evolution of the rewards as a consequence of the actions taken, rather than using the history to calculate the current reward. This last element is fundamental, as will be shown later.

Reduced Overestimation Bias: The removal of exponentially scaled bonuses and step-dependent rewards reduces the magnitude of Q-value targets, mitigating the well-known overestimation bias in Q-learning algo-

rithms. The bounded reward range $[-13.1, 5.0]$ provides predictable target values for the critic networks.

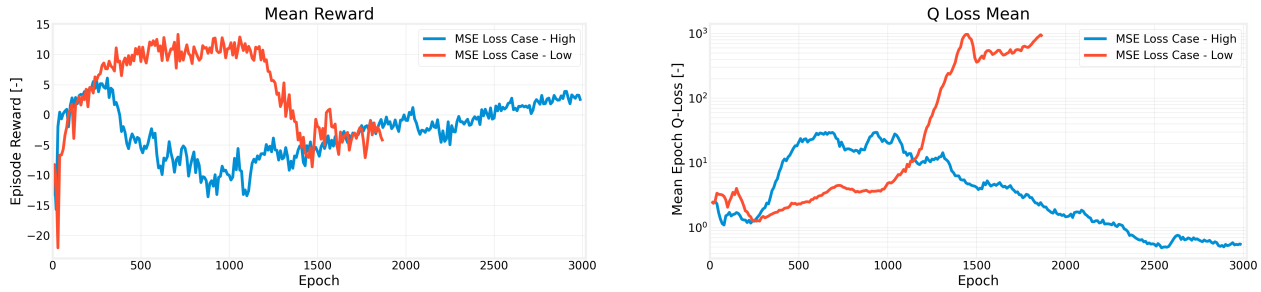
Consistent Gradient Signals: Fixed coefficients eliminate the need for tuning the reward function, and they are defined based on the desired behaviour: focus on converging quick convergence, and high rewards for staying there.

Simplified Credit Assignment: The binary convergence bonus and direct action penalties create unambiguous signals about desirable behaviour, simplifying the temporal credit assignment problem that can cause ambiguities in the assignment of Q-values.

Despite its simplicity, this reward function maintains the essential properties required for Φ_2 phase optimisation: it provides continuous guidance toward the optimal phase, rewards progress, penalizes simulation failures, and includes mechanisms to prevent excessive actions. The trade-off between sophisticated behavioural shaping and training stability makes this simplified approach particularly suitable for applications where reliable convergence is prioritized over optimal sample efficiency.

Furthermore, a key aspect of the current training using Prioritized Experience Replay (PER) buffers, is that while it allows for faster training, it is incredibly susceptible to overestimations in the Q-values with feedback mechanisms that deteriorate performance. This is because the priority assignment to each transition is based on the temporal difference (TD) error,which can explode when the Q-network assigns large values that differ substantially from the rewards that are stored in the PER buffer. As a result, the transitions that cause the large overestimation in Q-values, are now given a high priority and are sampled more often in the updates, when it could be simply due to an instability in the Q-value assignment and not because that transition is particularly informative. Then the TD error is squared on top of this, so the loss of the Q-network spikes leading to a huge gradient update in a direction that perhaps was not the optimal one.

This behaviour was observed in early trainings with the initial reward function of Algorithm 7, as shown in Figure 5.58 for a high ($r_l = 10^{-4}$) and low ($r_l = 10^{-5}$) learning rate (using $n_{envs} = 46$) and it led to complete training degeneracy after having already settled on a policy that either performed well (for the low learning rate) or was learning to do so (for the high learning rate). It also makes sense that a higher learning rate would lead to an earlier onset of learning degeneracy, since the Q-networks are applying larger gradient updates, which makes the training inherently more unstable. However, $\mathcal{L}_Q$ is much larger for the low learning rate case than for the high one. This is because it was already settling on a well performing policy, and the instability completely shifted the perception that the Q-network had on good Φ_2 corrections, which is also why the performance decreased in a dramatic fashion (over a lower amount of epochs) in comparison to the high learning rate case. Also note that the training was terminated early for the low learning rate case as the performance plummeted.



(a) Mean Epoch Reward

(b) Mean Q-loss $\mathcal{L}_Q$

Figure 5.58: Initial training statistics before applying fixes to Q-value over-estimation for high ($r_l = 10^{-4}$) and low ($r_l = 10^{-5}$) learning rates using the initial reward function in Algorithm 7 using 46 parallel environments in training

Thus, a strategy was devised to mitigate this which is not used in any of the available LSTM-TD3 implementation: replace the MSE loss for the Huber loss, which linearizes the error and makes it more similar to the Mean Absolute Error (MAE), after a certain deviation from 0. This is exemplified in Figure 5.59, where using this loss function will lead to less sensitivity to outliers while still giving smooth loss gradients near a TD error of 0, which combines the best of both worlds. This loss was shown to improve performance and stability in regular TD3, so applying it to LSTM-TD3 seems natural [14]. Additionally, an auxiliary loss is added to the complete loss function to regularize the Q-values coming from both networks. This makes the final loss function now be (modified from Equation 4.21):

$$\mathcal{L}_{Q_i} = \frac{1}{B}\sum_{j}^{B}[\text{Huber}(Q_{\phi_i}(s_j, a_j) - y_j) + Q_{\phi_i}(s_j, a_j)^2 \times 0.001] \tag{5.14}$$

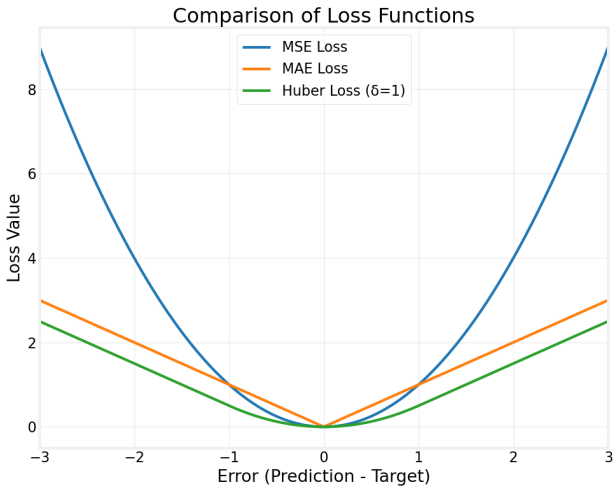


Figure 5.59: Huber loss compared to Mean Squared Error (MSE) and Mean Absolute Error (MAE) loss

Finally, as a last resort, to avoid the feedback loop between the PER buffer and any possible overestimation, priority values are clipped at a value of 100. This value was chosen empirically based on the current training rewards and the observed Q-loss during multiple trial runs when learning was still stable, and it is done such that some values cannot achieve higher priority and be over-sampled in updates due to any instabilities in the Q-network updates. This allows us to move on to hyperparameter optimisation using the Optuna framework.

5.2.3. Hyperparameter Optimisation & Analysis

Initial Considerations

Contrary to the case for the CBAM model, each trial is now much more expensive as training the model is done by interacting with the environment and it takes considerably longer than training using a pre-made dataset. Thus, only one coarse hyperparameter optimisation is performed with a low amount of trials due to the limitation of computing resources at CERN's High Performance Computing (HPC) cluster (and also only being able to run code for a max runtime of a week) and more analysis is required using knowledge on the TD3 algorithm to determine which parameters are the most relevant. This is because, as mentioned in the previous hyperparameter optimisation, the accuracy of the Gaussian Process in modelling the objective value as a function of the hyperparameters, and hence the confidence of the TPE sampler in selecting the most promising hyperparameters, relies on the amount of times that the objective function is sampled. Since we are crucially limited in this case with the number of trials we can perform, the reliability of the model will be lower, and it will need to be confirmed with the aforementioned analysis.

Regardless of these limitations, it is crucial to consider how long we will be training for since it is useless to keep training after the agent has already converged to a good solution as it limits even further the amount of trials we can perform. Using the plots shown in Figure 5.58, other similar initial trainings performed (not shown), considering the fact that the PER buffer used had a capacity of 1,000,000 transitions, and that $n_{envs} = 46$ in the plots shown, it was possible to determine a fixed amount of epochs to use for the hyperparameter optimisation. The loss would be expected to converge (under good training conditions without learning degeneracy) after about 700 epochs, which corresponds to 1,000,000 steps being performed. This, coincidentally, is the size of the replay buffer, which could only be this large because of the large amount of memory available in nodes used in the HPC cluster, in practice it need not be this large for good training since we are weeding out uninformative transitions anyway.

Thus, because we do not care how long we take to train in the optimised model that will come from this optimisation, we want to make sure that we are getting indicative performance of the model when we prove the final mean epoch reward and standard deviation. As a result, we use $n_{env} = 190$ (the maximum amount of threads available for a node in the entire HPC cluster) to get the most out of the max runtime available, and limit the number of epochs to the amount of mappings that there are in the $(V_{total}, r_2, \dot{\mathcal{B}}) : \Phi_2(BLM)$ dataset, which is 365. Like this, we are also certain that there will be enough epochs to have trained on all of the parameter combinations and all filling factors as a result of the 190 environments, each with their random order of parameters. This corresponds to roughly 2,000,000 steps being performed, allowing for a full refresh of the PER buffer.

The optimisation is performed using the initial reward function detailed in Algorithm 7 with a high base penalty (meaning a large C_{dist}) in order to foment the algorithm to not have a high steady-state error, as the primary concern of the optimisation is the accuracy that can be achieved. As a result, the mean epoch returns are expected to be low, and perhaps negative. Finally, it is important to define the architectures

that are evaluated. Anticipating that the network architectures will not be the most significant aspect of the training given the literature (besides the amount of parameters that will need to be optimised), a simplified approach is taken [53]. To not have to optimise each individual modules as was done for the CBAM model, three architectures are defined with varying levels of complexity, with a large, medium or small amount of parameters. These parameters are summarized in Table 5.5 including a description of what each component or module is performing in the actor or critic networks. This is a more detailed view of what was presented in Figure 4.10.

Table 5.5: LSTM-TD3 network architecture details for Optuna optimisation (see Figure 4.10 for depiction). If two values are placed for a given module, it means there are 2 hidden layers

Component	**Small**	**Medium**	**Large**	**Purpose & Information Processing**
		Critic Network (Q-Value Estimation)		
Pre-LSTM MLP	64	128	256 → 128	Preprocesses history (obs+act): extracts temporal patterns from beam profile sequences
LSTM Core	128	256	512	Sequential memory: captures long-term dependencies in phase adjustment effects
Post-LSTM MLP	64	128	256	Memory encoding: distills LSTM output into control-relevant temporal features
Current Feature	128 → 64	256 → 128	512 → 256 → 128	Instantaneous processing: current observation + action value assessment
Post-Combination	64	128	256 → 128	Feature fusion: combines temporal memory with current state for Q-value
Output Layer	1	1	1	Final Q-value: scalar reward prediction for state-action pairs
		Actor Network (Policy)		
Pre-LSTM MLP	64	128	256 → 128	History preprocessing: beam profile sequence feature extraction
LSTM Core	128	256	512	Temporal memory: learns phase adjustment strategies from history
Post-LSTM MLP	64	128	256	Memory distillation: extracts actionable insights from temporal patterns
Current Feature	128 → 64	256 → 128	512 → 256 → 128	Current state processing: beam profile analysis for immediate action
Post-Combination	64	128	256 → 128	Policy integration: fuses memory with current state for action selection
Output Layer	1 (Tanh)	1 (Tanh)	1 (Tanh)	Phase adjustment: $\Delta\Phi_2$ action bounded to $[-\pi/4, \pi/4]$
		Input Specifications		
Observation Dim	1001	1001	1001	Beam profile (1000 slices) + synchronous phase ϕ_s

Continued on next page

Table 5.5 – continued from previous page

Component	Small	Medium	Large	Purpose & Information Processing
Action Dim	1	1	1	Second harmonic phase adjustment $\Delta\Phi_2$
History Length	Variable	Variable	Variable	Past observations/actions for temporal context (episode boundaries)
Architecture Characteristics				
Total Parameters	∼180K	∼520K	∼1.8M	Parameter count scales with network complexity
Training Speed	Fast	Medium	Slow	Smaller networks train faster, and are less prone to overfitting

Optimisation Results

As was the case for the CBAM optimisation, we average the importances over 1000 runs, which is especially important in this case since there are **only 18 trials** performed. The results for the parameter importances are shown in Table 5.6, while the optimal parameters are listed and described in Table 5.7, which corresponds to 2 and a half weeks of runtime to get these results (since every trial basically took an entire day to perform). Meanwhile, contour plots are shown in Figure 5.60 showing the evaluation points for the top 3 hyperparameters and the respective objective values that are obtained.

Table 5.6: Parameter importance (% variance) for the LSTM-TD3 Optuna study (averaged over 1000 runs, renormalised excluding history-action flags)

Parameter	fANOVA (%)	MDI (%)
Critic learning rate	26.90	32.35
Exploration noise (std)	21.14	21.27
Actor learning rate	11.40	9.76
Batch size	10.18	9.32
LSTM history length	8.08	6.58
Target network Polyak factor	7.64	8.00
Replay ratio	4.65	3.95
Target smoothing noise clip	2.99	2.74
Target smoothing noise (std)	2.66	2.08
Discount factor γ	2.44	2.63
Policy update delay	1.00	0.77
Network complexity scaling	0.88	0.55

The most important parameters were the learning rates, the exploration noise, and the batch size. This is consistent with the TD3 architecture knowledge. For the learning rate, given that they are defining how quickly we are learning and making use of the estimates of the critic networks, if they are too high (especially for the actor) then the performance deteriorates since we are updating our actor based on an estimate, but if they are too low (especially for the critic) then the network learns too slowly, making the agent even more sample inefficient. Consequently, as evidenced in the lower middle plot of Figure 5.60, a critic learning rate one order of magnitude higher than the actor learning rate seems to be the most ideal combination, and it can be appreciated that because the axes are logarithmic, any exponential relationship now becomes linear in this plot, and there appears to be an optimal line where the highest returns are found (up to a certain extent when the actor learning rate becomes too high). This indicates that if this line is followed further to the left (meaning lower actor learning rates) then there is potential for even further increased rewards as the actor will now be updating in a more precise manner when close to the optimum. Since in the complete training, there will not be the limitation of the reduced epochs that we had to use here in order to have enough trials, then this can be done to basically make the actor learn slower, but converge to a better performing model. Thus, the learning rates in Table 5.7 will be reduced in the final training to 0.7×10^{-5} and 7×10^{-5} for the actor and critic, respectively. For the rest of the parameters, everything is maintained the same.

Meanwhile, the action noise can also be seen to have a large effect in Figure 5.60, where there seems to be a sweet spot for the exploration noise added to every action given by the actor. This makes perfect sense as too low noise and then there could be reduced exploration causing the actor to never be able to learn the best possible transitions. This is most important when close to convergence and wanting to stay there, as the profiles can sometimes not be very informative of what action to take, especially at high filling factors FF and low voltage ratios r_2 where one could change the phasing 5° and the profile would look practically the same. So being able to learn how to reach very precise convergence in these cases requires small actions to be performed in an explorative fashion such that the critic can start to learn to look for the smallest of features in these cases that lead to the correct actions which could be added to the buffer as a consequence of the small exploration noise, but sufficiently high. Luckily, when this happens, we reward the agent handsomely, so then the TD error can be high once the critic encounters these transitions.

As for the batch size, it is not important in the sense that the gradient updates are more smoothed out due to more transitions being included inside of the loss estimate, but it has more to do with how many gradient updates we are performing with the constant amount of parallelized environments ($n_{env} = 190$). Remember, the amount of gradient updates performed in a given step was given in Equation 4.37, in which a lower batch size B indicates more gradient steps to be performed. Thus, given that the optimal number for the batch size was 128, and this was the lowest value available in the allowed values (128,256,512 and 1024), it indicates that more updates are more beneficial for this current training task. This probably improves the sample efficiency, while the usage of small batch sizes implies more high priority transitions are being used in the updates (due to the sampling probability of the PER buffer), which clearly benefits faster learning as this was the entire reason the PER buffer was invented and implemented in many RL trainings [33, 32].

Finally, the suspicions on the network architecture were confirmed, showing that it did not account for considerable variance in the objective value. However, the best trial used a small network architecture, and using a larger amount of parameters does correlate with decreased final returns, which makes sense given that the training will take considerably longer for 1.8 million parameters than it does for 180 thousand. This decision is also supported by the findings of studies that indicate that simpler RL actor architectures with a high compression of input information, perform better than overcomplicated and "too" expressive architectures. This could also be due to similar reasons as for why the research that was performed on the compatibility the convolutional layers with the dense layers in MLP: dormant neurons beginning to appear that do not contribute to the learning and are unnecessarily bloating the actor and critic networks [57].

Table 5.7: Best hyperparameters for LSTM-TD3 (single study), with descriptions

Parameter	**Description**	**Value**
Actor learning rate	Step size for the actor optimiser.	1.29×10^{-5}
Critic learning rate	Step size for the critic optimiser.	9.93×10^{-5}
Discount factor γ	One-step return discount.	0.971
Target Polyak factor	Update coefficient for target networks.	0.994
Exploration noise (σ)	Standard deviation of action noise during data collection.	0.0266 rad or 1.524°
Target policy smoothing noise (σ)	Standard deviation of noise added to target policy actions.	0.0257 rad or 1.473°
Batch size	Samples per gradient update.	128
Policy update delay	Q-function updates per policy update.	3
Replay ratio	Amount of times a transition should be used in the update on average.	37.77
LSTM history length	Number of past steps provided to the LSTM.	5
Network complexity preset	Preset for network complexity illustrated in Table 5.5.	Small

In conclusion, while the results produced by the hyperparameter optimisation are done with a very limited number of evaluations, the results are consistent with the TD3 theory, and more importantly, they generated an improvement of halving the negative returns that were produced by the initial, unoptimised model. As a result, thanks to the analysis, the improvements produced, and the empirical confirmation of the understanding of the importance of TD3 hyperparameters, there can be confidence in the results of the hyperparameter optimisation. Of course, it would still be beneficial to repeat this optimisation for a reduced amount of parameters and smaller

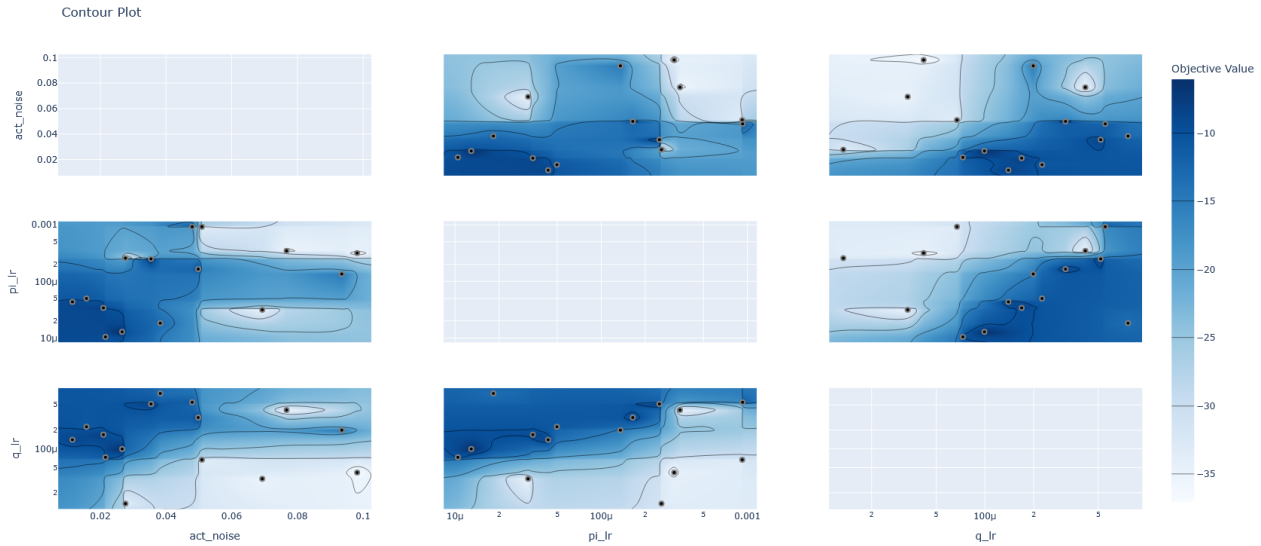


Figure 5.60: Contour plots for the 3 most influential hyperparameters: the learning rate r_l of the critic and actor, and the action noise used to explore the domain.

parameter bounds as was done with the CBAM model (including lower learning rates than what was allowed by the parameter ranges), and also using the simple reward function for more consistent rewards and returns.

5.2.4. Simulation Results

Before starting to discuss the verification analysis for the LSTM-TD3 model, it is important to first consider one change to the actor and critic networks that is catering directly to the current task at hand. This change was considered after having tested the RL training using the optimised hyperparameters, and using the CAE latent space as input as it was seen to have meaningful phase correction information encoded within it. However, after 2 million steps in training, there were absolutely no improvements, with the rewards being completely stagnant with a small oscillation due to the added action noise of the TD3 architecture. This was also confirmed by the diagnostic animations which showed that no action was being taken (or very small random ones of 0.5°), as the RL agent could not understand at all what the encoding meant. It is unclear whether this is because the CAE was not trained together with the RL agent, or because of the issues with the CAE training itself.

Considering the fact that one of the main strengths of the CBAM model was that through the spatial attention, it was able to bypass completely the baseline droop through proper bin weighting where the important features lied, but this is not something that could be implemented in the RL actor network due to the convolutional layer incompatibility. As a result, an efficient solution was devised that only requires one additional parameter to be trained, and improved the performance considerably: an adaptive threshold that masks the input in a differentiable manner before entering the MLP actor structure

Thresholding: Solution to Baseline Droop

The `SoftThreshold` module implements a learnable gating mechanism for bunch profile "de-noising" in the LSTM-TD3 actor network. The module can be described mathematically in the following manner:

$$\tau_{eff} = \tau \cdot \bar{p} \quad \text{where } \bar{p} = \frac{1}{N}\sum_{i=1}^{N} p_i \tag{5.15}$$

$$M_i = \sigma(s \cdot (p_i - \tau_{eff})) \tag{5.16}$$

$$p_{filtered,i} = p_i \cdot M_i \tag{5.17}$$

where $p \in \mathbb{R}^{1000}$ is the bunch profile, τ is the learnable threshold parameter (initially it is 0.1), $s = 50$ is the sigmoid scaling (high values mean a more pronounced gradient in the mask), $\sigma(\cdot)$ is the sigmoid function, and M_i is the learned mask. By using the mean value of the profile, we are making it adaptive to the profile, and ideally making it adapt to different emittances in the PSB. Smaller emittances, and therefore shorter bunches, would have lower mean values (less bins that are to 1) and cause larger induced voltages that induce more baseline droop, and since we want to maximize the profile information for the shorter bunches (to capture statistics like overall width), it is good to have a lower effective threshold in these cases.

This masking mechanism is applied at two stages in the actor:

- **History buffer profiles**: $p_{hist} \leftarrow \text{SoftThreshold}(p_{hist}, \bar{p}_{hist})$
- **Current profile**: $p_{cur} \leftarrow \text{SoftThreshold}(p_{cur}, \bar{p}_{cur})$

The synchronous phase ϕ_s is preserved unchanged and is not gated, which also gives a much clearer signal to the network that this parameter is different from the 0's (or very small numbers) that are next to it in the majority of cases: $[p_{filtered}, \phi_s]$.

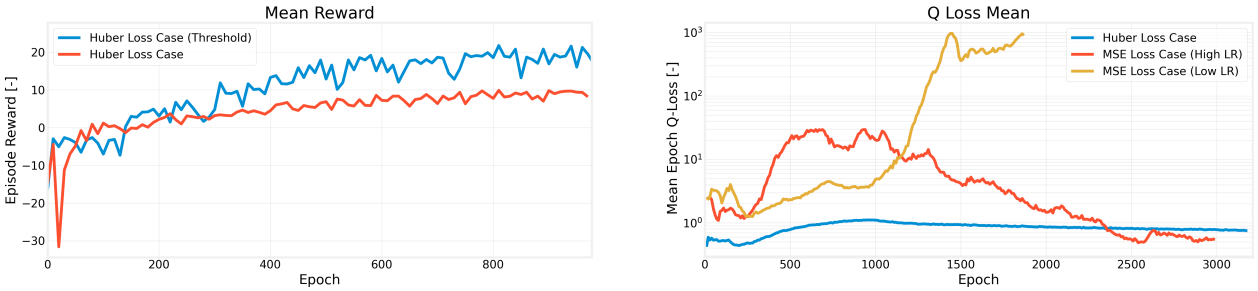
Furthermore, the key properties of this method are that it is adaptive by scaling with the mean of the profile, it is differentiable by using the sigmoid instead of jagged binary masks, easily learnable (as only one additional parameter is introduced), and it suppresses the low-signal artifacts induced by baseline droop and any noise in the signal. This enables robust Φ_2 phase control by filtering measurement artifacts that can appear randomly as shown in Section 5.1.7 and focusing the agent on relevant bunch profile features.

Comparison to Baseline Model

Now we compare the two models based on the epoch returns they attain and the phase errors that are achieved throughout training using the simplified reward function as depicted in Algorithm 8. These are shown in Figure 5.61a and Figure 5.62. respectively. Additionally, the stabilization capabilities of the simplified reward function combined with the Huber loss and Q network regularization are demonstrated through the mean Q-loss shown in Figure 5.61b, which (if TD errors are contained within the parabolic region of the Huber loss) should be slightly larger under ideal learning conditions due to the added regularization loss terms.

However, the important comparison to be made is where the training reward starts to plateau with each model and how the learning progresses. The models were trained with a maximum action of $\pm\frac{\pi}{4}$ radians or $\pm 45°$ to bound the behaviour of the model and speed up the learning process[51]. In Figure 5.61a, the learning kicks off from a considerably higher value in the case of the thresholded model. This is primarily because of the fact that the threshold allows the agent to start discerning the relevant from the non-relevant regions of the profile early on, and acting only based on the important parts. This is especially significant when the profiles are augmented with droop, as the non-thresholded model has to learn this through many episodes. Additionally, the non-thresholded model plateaus at a lower episode reward, which might be because of the lack of specialization that the modules have towards identifying relevant phase correction features, as they also have to be robust to the added baseline droop and the high frequency noise that is inherent to the low number of macroparticles used in the profile generation ($N_m = 10^6$). It should also be noted that the training of the model without thresholding was interrupted once it was seen that it was performing worse in terms of the final phase error and the rewards (aside from the diagnostic animations that are produced).

All of these hypotheses are supported by the performance of the model when considering the final (smoothed) episode phase error shown in Figure 5.62. This error shows that the phase error stagnates at a higher value for the non-thresholded model which is in line with the stagnated episode reward in Figure 5.61a. As a result, moving forward, only the thresholded model is considered, except in validation for reasons that will be given later on in Section 5.2.5. Also note that the learnt threshold value τ was roughly 0.3, meaning that when multiplied with the mean of each profile, that can be assumed to range from 0.3 to 0.6 for the normalised profiles, the effective threshold value ranges roughly from 10% to 20% of the profile maximum, which means that the baseline droop (which has a maximum value of 4% as described in Section 4.1) is completely bypassed by the model, as desired, and enabling more meaningful learned parameters for the rest of the LSTM and MLP modules.



(a) Mean Epoch Reward (for only the Huber loss models)

(b) Mean Q-loss $\mathcal{L}_Q$

Figure 5.61: Training statistic after applying stabilization fixes for Q-value over-estimation for MSE error cases with high ($r_l = 10^{-4}$) and low ($r_l = 10^{-5}$) learning rates simplified reward function in Algorithm 8 using 46 parallel environments in training

Figure 5.62: Smoothed phase error (in degrees) comparison between thresholded and non-thresholded model

Effect of Initial Phasing & Varying Impedance

Now the same analysis will be performed as for the CBAM model to assess its robustness to the increased impedance and varying initial phase estimates. The same procedure is followed with analysing two initial conditions: one at $-90°$ for all C-Times and another with $90°$, with the $-90°$ initialization being closer to the optimal value, and the $90°$ case being seeded in BSM. The final results after 10 corrections (with no need for decay like the CBAM model) are shown in Figure 5.63 and 5.64 for $90°$ and $-90°$, respectively, where the ideal phasing is achieved for both cases. The LSTM-TD3 agent is far more robust to initialization in BSM, as it can escape it when slightly perturbed from the most symmetric BSM phasing, unlike what happens to the CBAM model. Additionally, the bounded nature of the action can clearly be seen in Figure 5.63, where several corrections are required to reach the optimum, with the fourth correction essentially achieving the correct phasing.

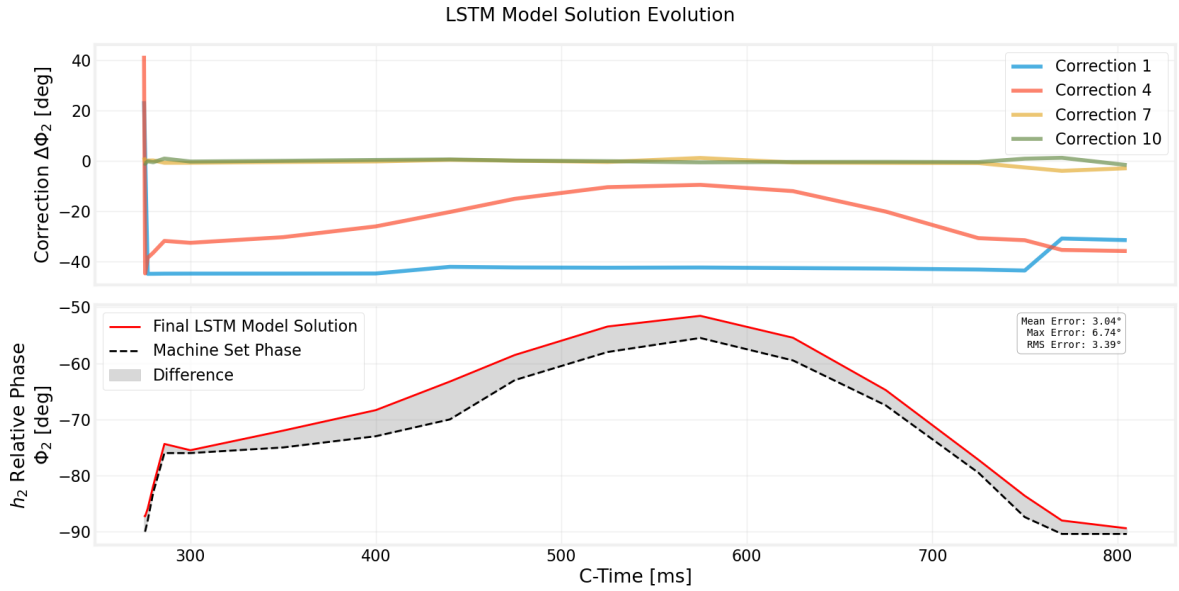


Figure 5.63: CBAM solution for ISOLDE cycle under twice the PSB impedance: correction evolution and final Φ_2 when initialized at $90°$ (CBAM solution in red as before)

5.2.5. Verification Analysis

Global Error Summary

Here the global errors are presented and compared between the optimised model using the soft thresholding method, and an initial unoptimised model, but both trained on the same simplified reward function. These two models are compared because in the validation practices, there was not sufficient time to wait for the training of the optimised model to be completed before the last possible access to the CERN Control Center (CCC) where the optimisation scripts are run. As a result, only the unoptimised, non-thresholded model could be tested in practice, which is why the comparison in simulation is fundamental as it gives an indication as to the possible performance of the model if implemented in the PSB. Therefore, the results are presented in Figure 5.66a and Figure 5.66b for the mean and maximum errors acquired in each cycle.

As can be expected, the unoptimised model performs considerably worse than the optimised model in the mean error, with it being from 100 to 900% higher than for the optimised model. However, for the maximum error, this is not the case for the cycles where splittings are performed (AD Ring 3 and MTE), but this is mainly due

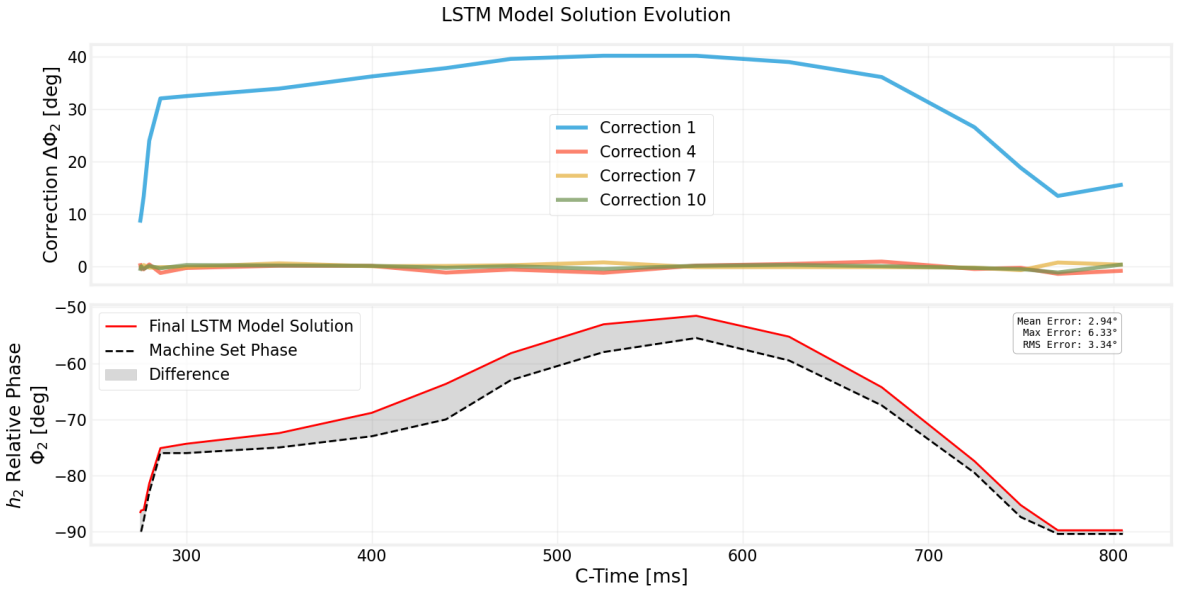


Figure 5.64: LSTM-TD3 agent's solution for ISOLDE cycle under twice the PSB impedance: correction evolution and final Φ_2 when initialized at $-90°$ (CBAM solution in red as before)

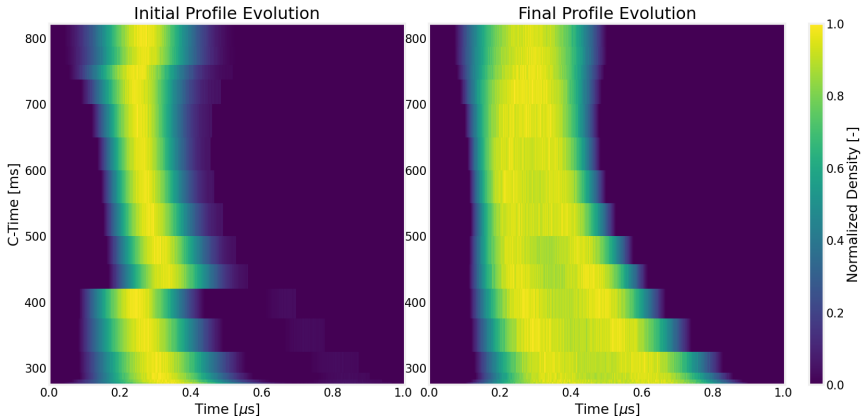
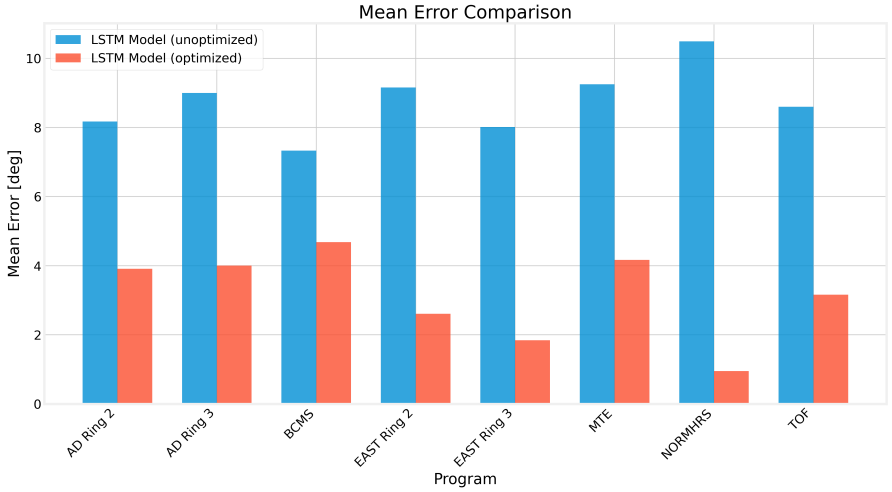
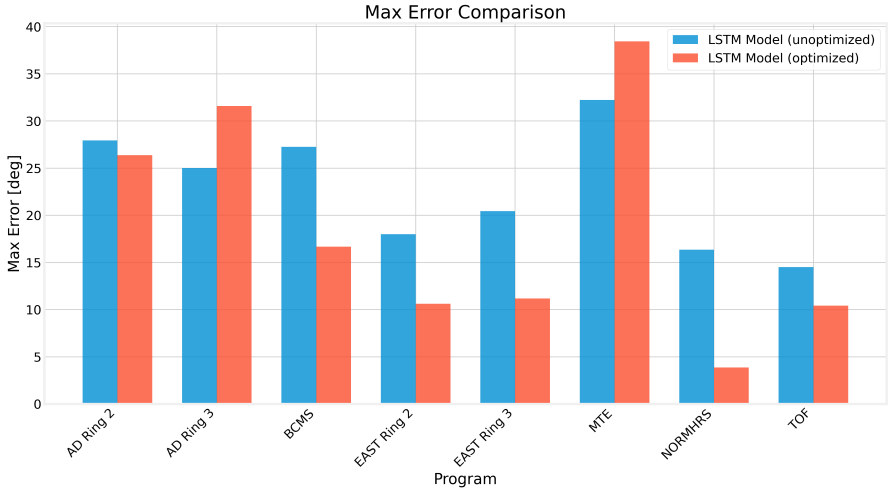


Figure 5.65: LSTM-TD3 agent's initial and final waterfalls for ISOLDE cycle under twice the PSB impedance when initialized at $\pm 90°$ (both have the same solution)



(a) Mean error across programs



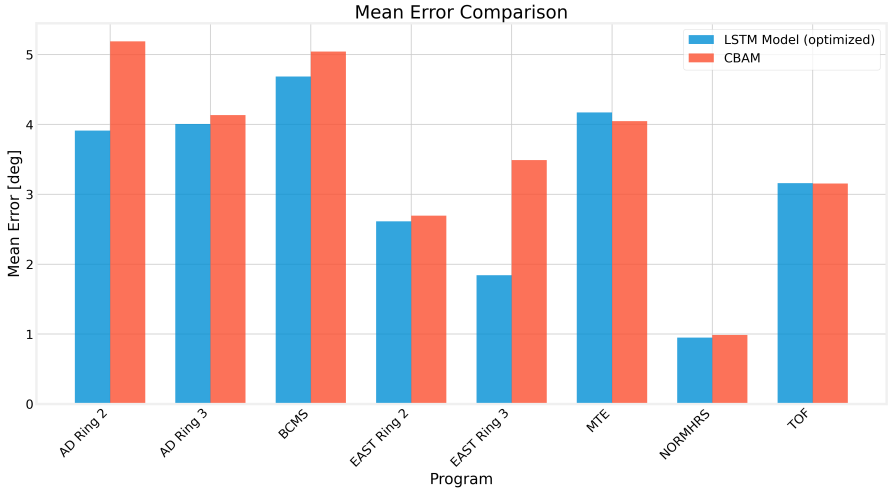
(b) Maximum error across programs

Figure 5.66: Summary of phase-error metrics for optimised and unoptimised LSTM-TD3 agents across the all operational PSB cycles

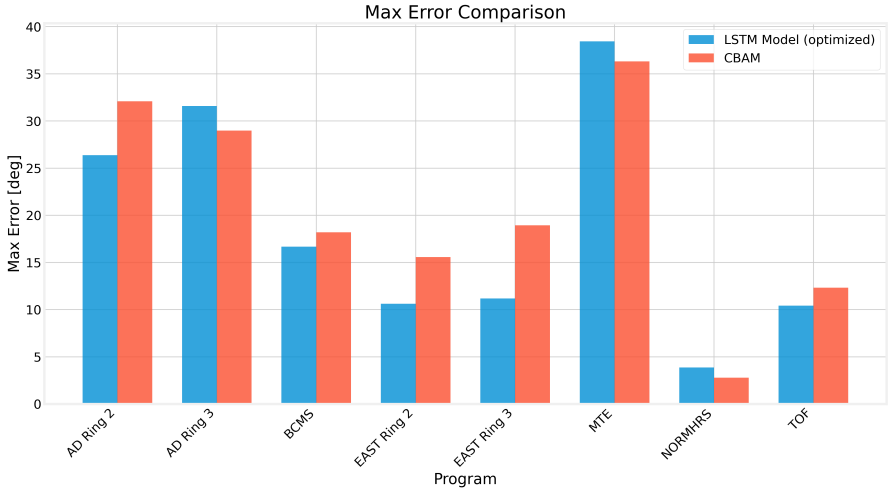
to the reasons presented in the previous verification analysis of the CBAM model. As the matching simulation is not considering the evolution of the particle distribution as it is being split, this results in apparent large errors at the end of the cycle, which are required in order to have matched peaks. Furthermore, the reason why the unoptimised LSTM-TD3 agent is having lower errors is because it is not matching the peaks correctly as will be shown in the verification plots. This comparison will be performed for only the BCMS and MTE beams as it gives a sufficiently detailed view of where the unoptimised model is lacking and it partially explains the results obtained in the validation procedures. If the unoptimised LSTM-TD3 agent's solutions to the ISOLDE and TOF cycles want to be seen, then one can find them in Section B.2.

Comparison with CBAM Model

In order to validate the reasoning behind the development of this model, it is also imperative to compare the errors that are incurred by the CBAM model and LSTM-TD3 agent during verification. As a result, Figure 5.67 shows this in a comparative bar chart plot for the optimised LSTM-TD3 agent and the optimised CBAM model. The mean errors for the LSTM-TD3 agent are reduced for almost all cycles, with the exception of MTE and practically the same behaviour for TOF and AD Ring 3. This coincides with the cycles in which the behaviour of the simulation deviates the most from the behaviour in the machine due to the static matching simulation which is performed at each C-Time without tracking the particle distribution splitting, as explained previously. Meanwhile, for TOF, it seems like the machine set solution is catering to PSB-specific effects which are not captured in simulation. As a result, for these cycles, the comparison should only be considered tentatively, and what mainly should be compared are the profiles that are produced.



(a) Mean error across programs.



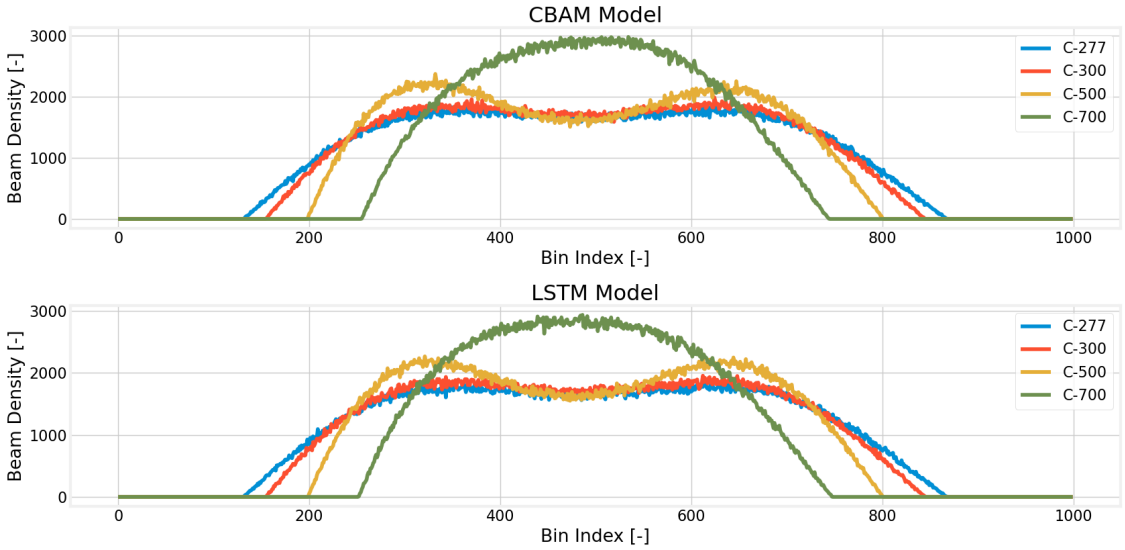
(b) Maximum error across programs

Figure 5.67: Summary of phase-error metrics for optimised CBAM model and LSTM-TD3 agent across the all operational PSB cycles

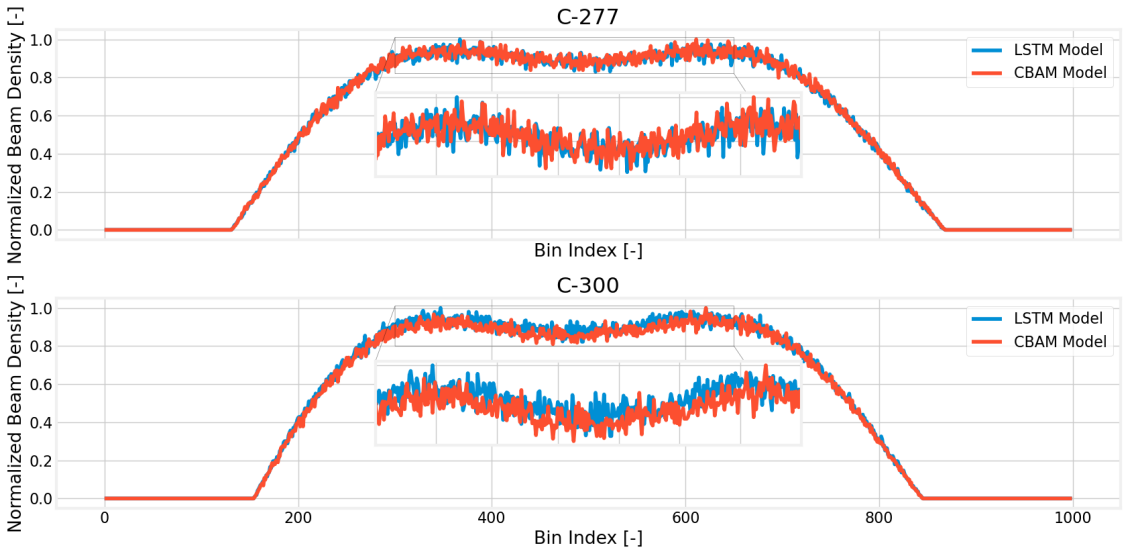
To probe where the differences between the solutions are most significant, and neglecting the splitting cycles and TOF, we will compare select profiles for the EAST Ring 3 cycle. The profiles selected are those where the differences in the solution are the greatest (see Figure B.7 and Figure B.15) which were at C-Times C-277,

C-300, C-500 and C-700. The results are shown for all the profiles in Figure 5.68a and a detailed view for the profiles at injection in Figure 5.68b. The performance is most distinguishable at C-500 and C-700 where the profiles for the CBAM model are slightly asymmetric with more charge on the left lobe at C-500 and slightly curved to the right at C-700, while the LSTM-TD3 agent achieved perfect symmetry in both cases.

Additionally, at injection the difference can barely be discerned, which is why the zoomed in insets were included. Here, the $\dot{\mathcal{B}}$ is very close to 0 at either of these points, so while the solutions deviate by a couple of degrees, there is almost no visible difference even in the normalized density plots, but it can slightly be seen that the peaks are better matched in the LSTM-TD3 case as here both lobes are at the same height with no tilt in the profile, while there exists a slight tilt towards higher charge in the right lobe for the CBAM model as shown in the zoomed inset plots. This performance increase is entirely due to the memory component that is integrated into the agent which allows it to achieve much better convergence behaviour than for a regressive model based on only one profile. Nevertheless, it is still impressive that the CBAM model can keep up with the LSTM-TD3 agent when the agent has more information to work with: 6 times as many profiles to decide on an action and actions which the CBAM did not have.



(a) Comparison of profiles at selected C-Times



(b) Comparison of profiles at injection

Figure 5.68: Comparison of LSTM-TD3 and CBAM produced profiles at the optimal solution for the EAST (Ring 3) cycle after 10 corrections for the points where the solutions deviated the most

Now that the LSTM model's capabilities have been demonstrated with respect to the CBAM model, since the optimised model has the same performance as the CBAM model, but slightly lower max and mean errors on average implying a better performance, the plots will not be discussed in as much detail as for the CBAM

model. More emphasis will be made in the differences between the CBAM and LSTM-TD3 solutions and, where pertinent, profiles will be shown to indicate which model is coming out on top at given computation points.

On a final note, it is interesting that the LSTM-TD3 model achieves lower error with respect to the machine set solution, as it indicates that the machine set solution is valid in simulation (except perhaps in splitting operations) it is fine-tuning to a degree in which it is more capable than that of the CBAM model. This is consistent with the training results presented in Figure 5.61a, as the LSTM-TD3 agent had its rewards maximized at a deviation of 0.5°, while a deviation of 0.5° from the optimum meant a loss of 3.8×10^{-5}, which is achieved only for very few parameter combinations in the CBAM model, while it is being achieved consistently for the LSTM-TD3 agent. This performance comparison also confirms why the CBAM model does not achieve exactly the right phasing at different points in the cycle in the validation study in Section 5.1.7, as the model does get close to the correct solution, but does not converge to the optimal profile.

BCMS

As mentioned previously, for BCMS both the optimised and unoptimised LSTM-TD3 agents will be analyzed. The 10 iterative and bounded corrections and final solutions for the optimised and unoptimised model are shown in Figure 5.69 and 5.70, respectively, while their corresponding waterfall plots are shown in Figure 5.71 and 5.72. The solutions differ, but since the initial estimate is −90° for all C-Times, optimal phase is reached in one correction for the optimised model, while the unoptimised is appearing to overestimate the phase at injection. This also shows in the waterfall of the unoptimised model where the first profile appears to have higher density in the left lobe than in the right one. So there is a certain drift in the corrections towards asymmetry for the unoptimised model, which should be expected when implemented in the PSB. Additionally, this drift is not only present at injection, but also at the highest acceleration rate between C-450 and C-650 and a large asymmetry to the left at C-750 corresponding again with the overestimation of Φ_2. It seems to be particularly worse when the double-lobe feature is not available, but it does still provide decent estimates in the idealized simulation environment. However, in simulation the results should ideally be optimal, if not very close to being so.

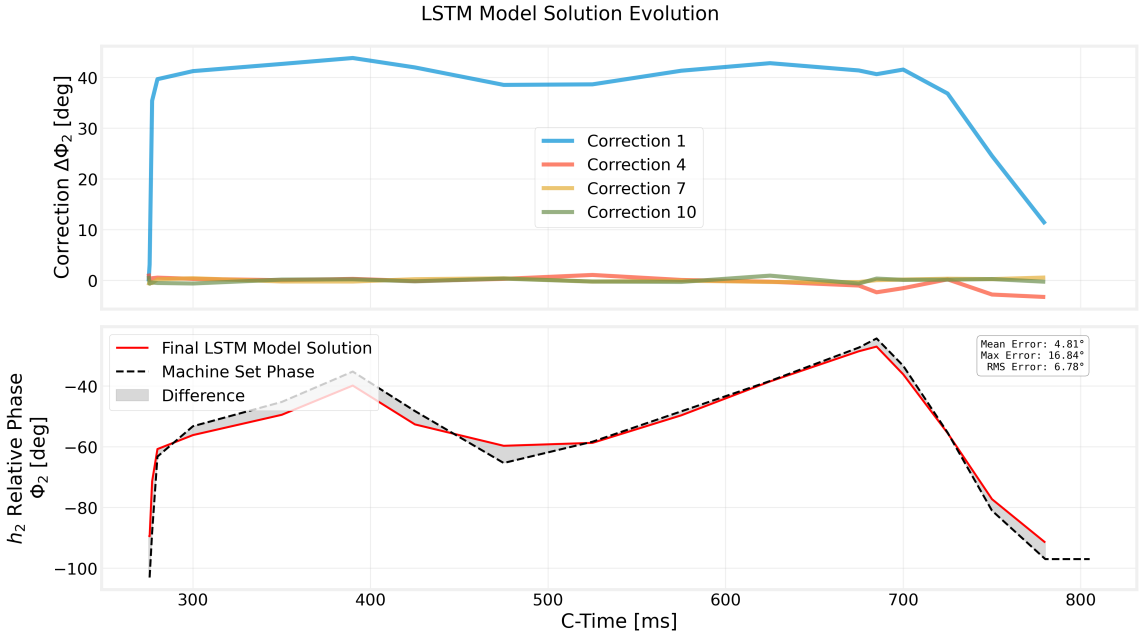


Figure 5.69: BCMS (2025) phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

Nevertheless, for the optimised case, the desired profile shape is maintained and there are no immediately worrying asymmetries in any of the profiles in the waterfall. Particularly promising is that the solution avoids kinks at injection and extraction which the CBAM model had in its verification and follows the manually phased solution closer (compare with Figure 5.12), especially at the kinks in the solution and notably at C-680 where CBAM underestimates Φ_2 causing a higher charge density in the right lobe.

ISOLDE

For ISOLDE, only the optimised model is analyzed, with its 10 corrections and solutions shown in Figure 5.73, while the resulting waterfall depicting the initial and final profiles throughout the cycle are shown in Figure 5.74. In contrast to the CBAM model in Figure 5.14, the LSTM-TD3 agent follows the manually phased solution almost exactly after C-575, but has a larger deviation around C-430 by +2°, but lower deviation practically

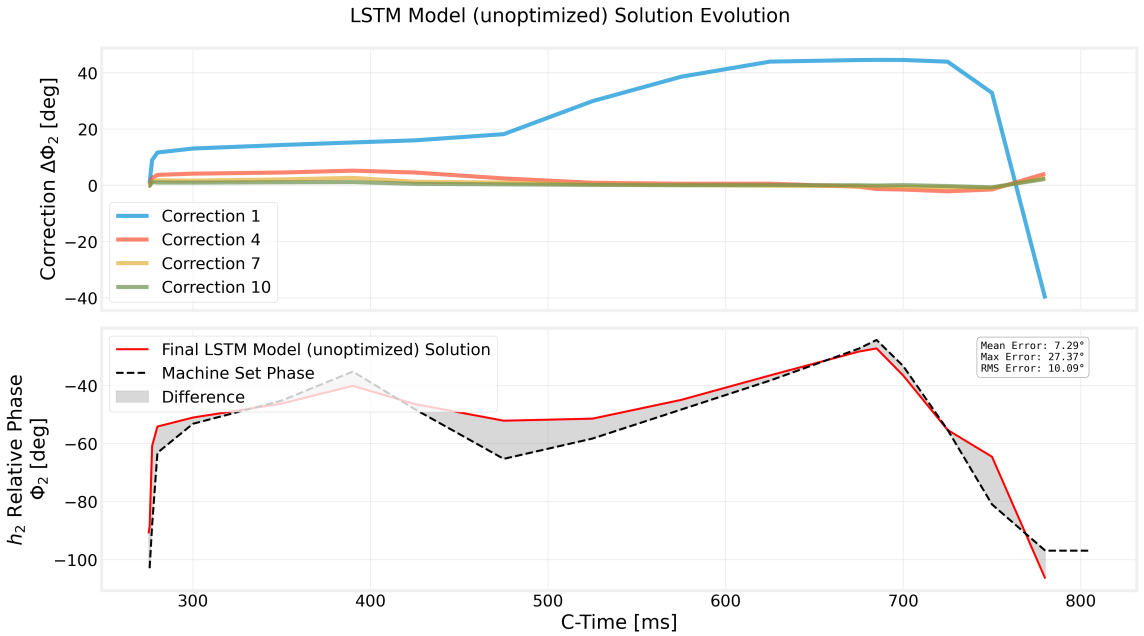


Figure 5.70: BCMS (2025) phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom) for unoptimised model tested in the PSB.

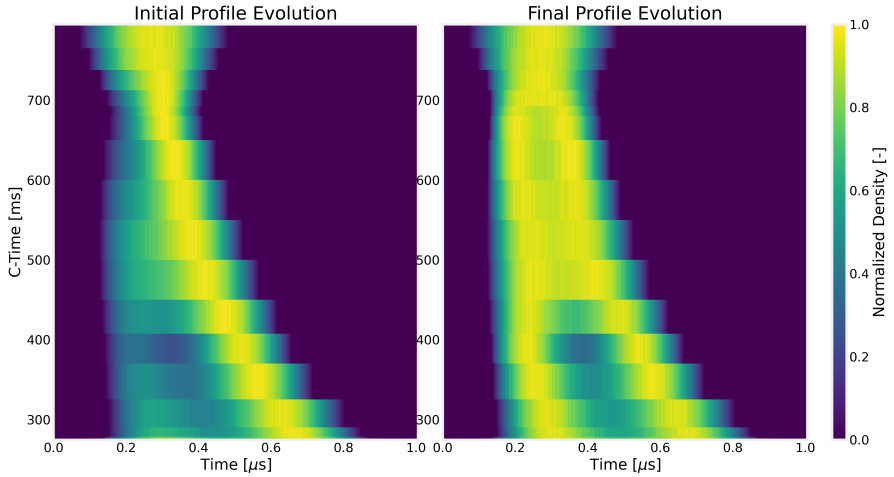


Figure 5.71: BCMS (2025) normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

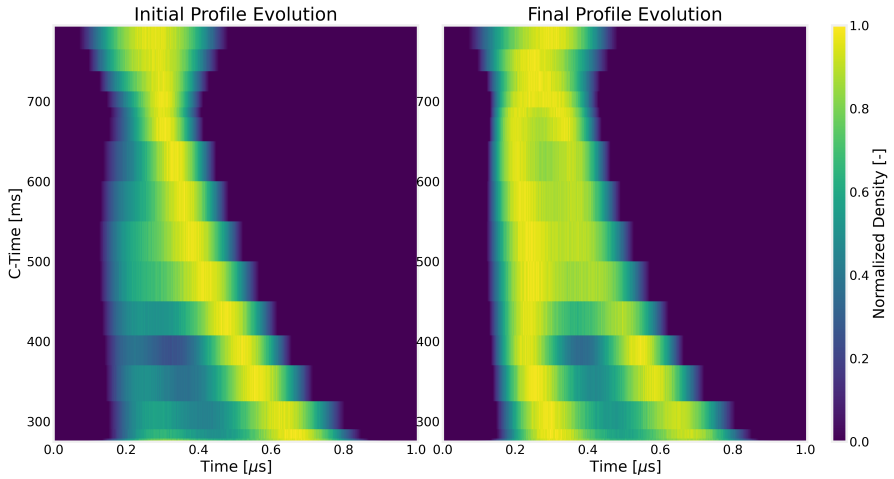


Figure 5.72: BCMS (2025) normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right) for unoptimised model tested in the PSB.

everywhere else. When this deviation occurs, as evidenced in the comparison profiles in Figure 5.75, this is working to correct a slight higher density in the right lobe at C-440, but the difference is almost negligible for all profiles as they all show good lengthening with the peaks practically matched for all C-Times.

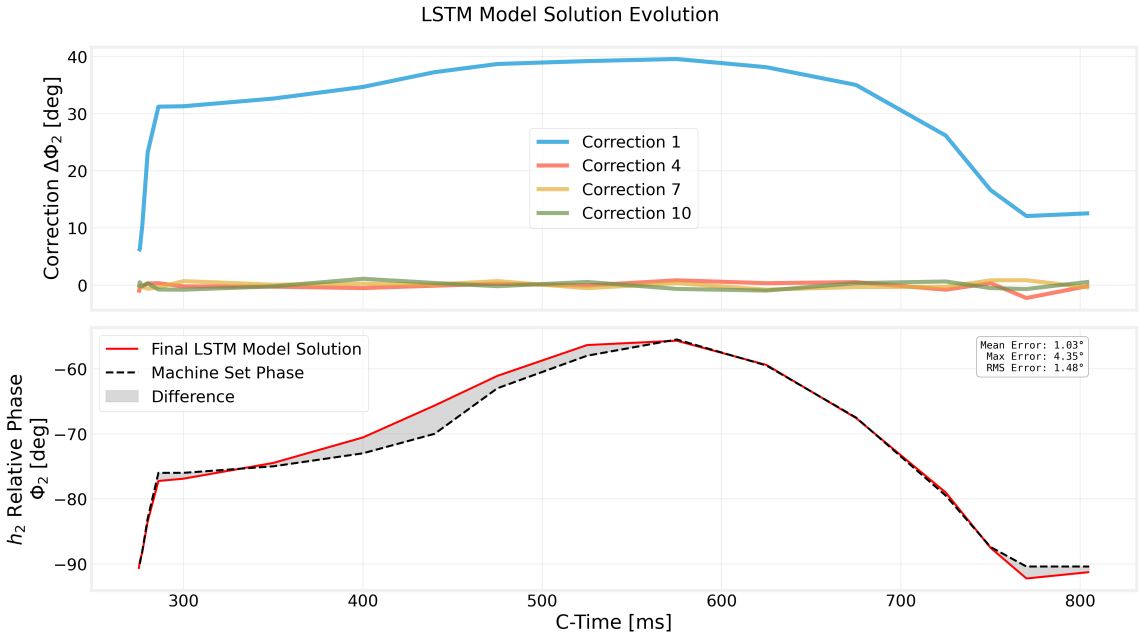


Figure 5.73: ISOLDE phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

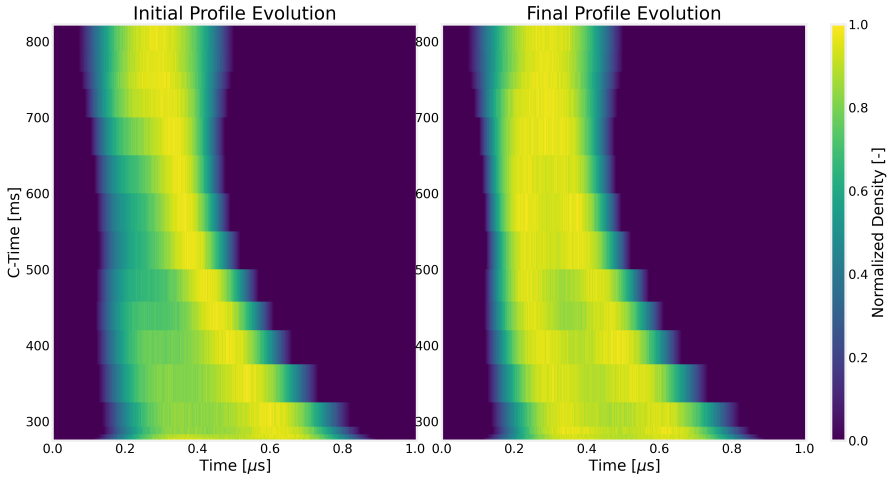


Figure 5.74: ISOLDE normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

MTE

For MTE the comparison between the optimised and unoptimised LSTM-TD3 agents returns, with the corrections and solutions shown in Figure 5.76 and 5.77, for each respective model. These solutions correspond to the waterfall plots shown in Figure 5.78 for the optimised model and Figure 5.79 for the unoptimised model. In this case, the overestimation of Φ_2 at injection that was present at injection for BCMS, is now exaggerated to a difference of 20° and maintained throughout the cycle with proper phasing only from C-680 to C-710. This is also evident in the final waterfall plot where the charge density is evidently higher on the left lobe than on the right one.

Furthermore, with regards to the optimized model, in the final waterfall plot in Figure 5.78 Φ_2 is always well phased. The only point where the solution differs by roughly 2° from the CBAM model (compare with Figure 5.16) is at C-450 and and C-760, with the latter leading to the higher maximum error in Figure 5.67b. As a result, the final profiles at these points are compared in Figure 5.80, and almost no difference can be discerned at first glance, but in the zoomed insets, the CBAM solution has a tilt towards the right in the C-450 case, and practically no differences in the C-760 case. As mentioned before, the Φ_2 value at the end of the cycle has little to no effect other than shifting the bucket. Thus, the optimised LSTM-TD3 agent's solution is verified as well for MTE.

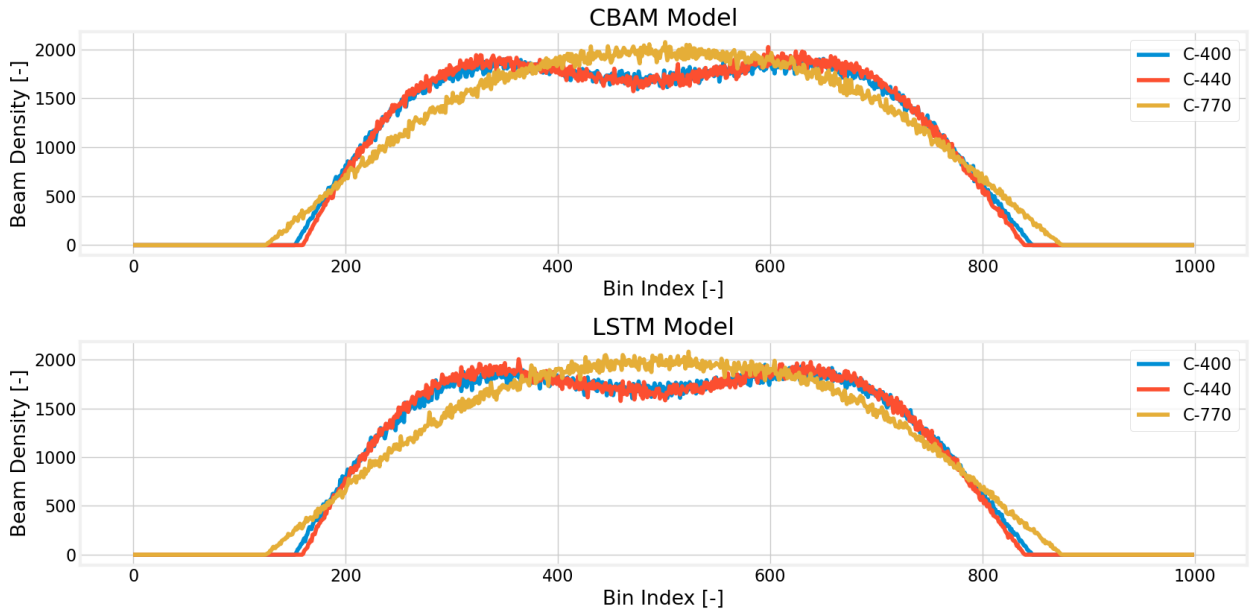


Figure 5.75: ISOLDE CBAM vs LSTM-TD3 model solution comparison at selected C-Times (C-400,C-440 and C-770)

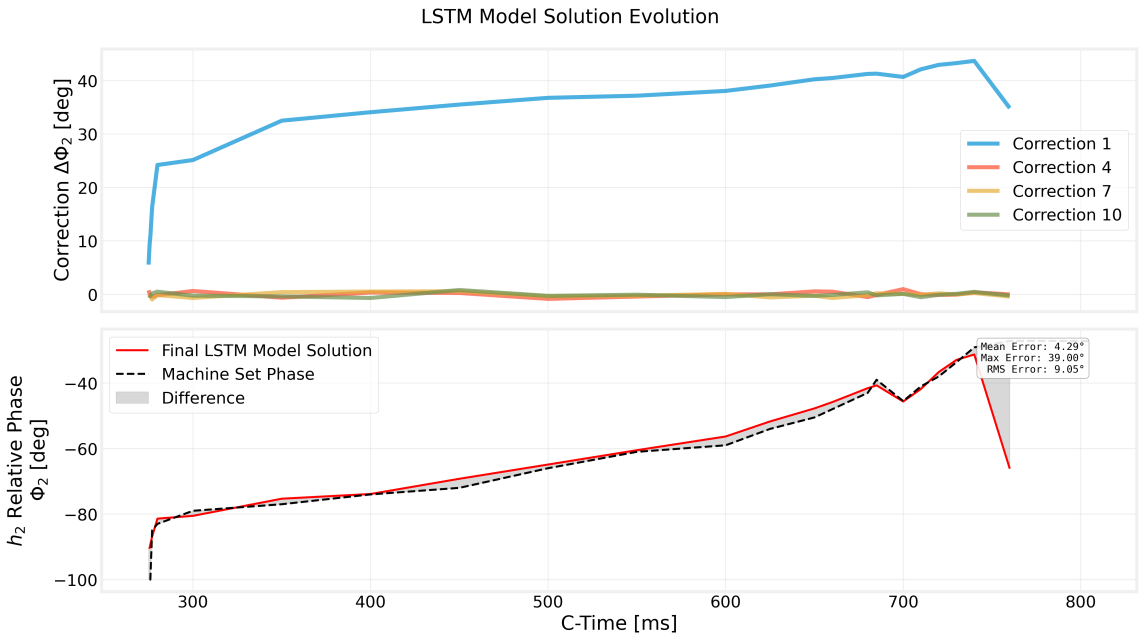


Figure 5.76: MTE phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

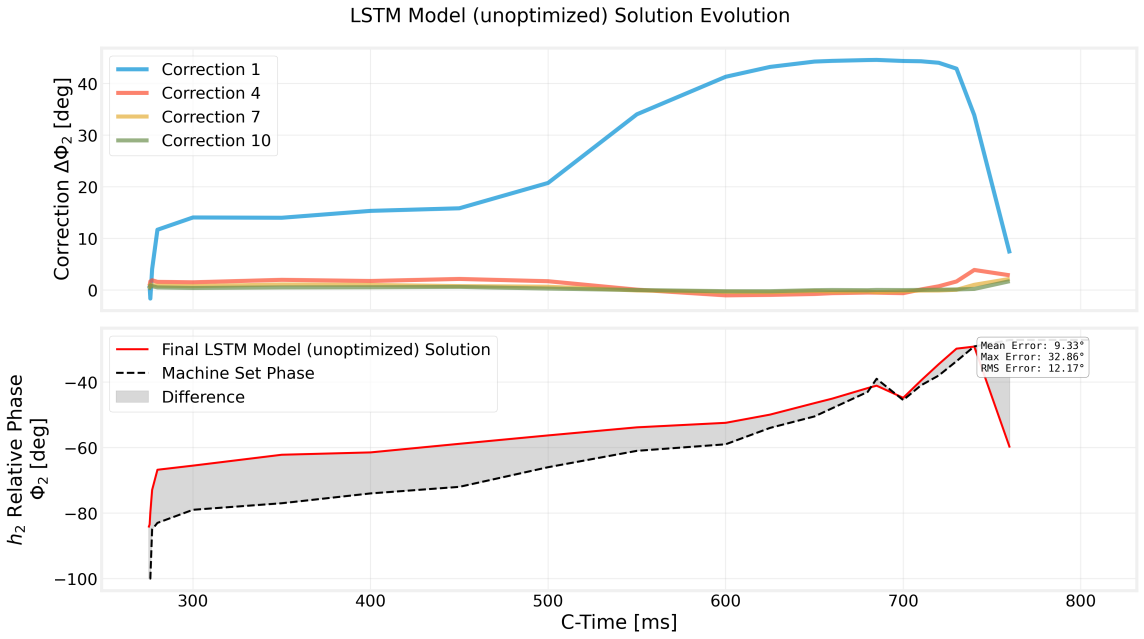


Figure 5.77: MTE phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom) for unoptimized model tested in the PSB.

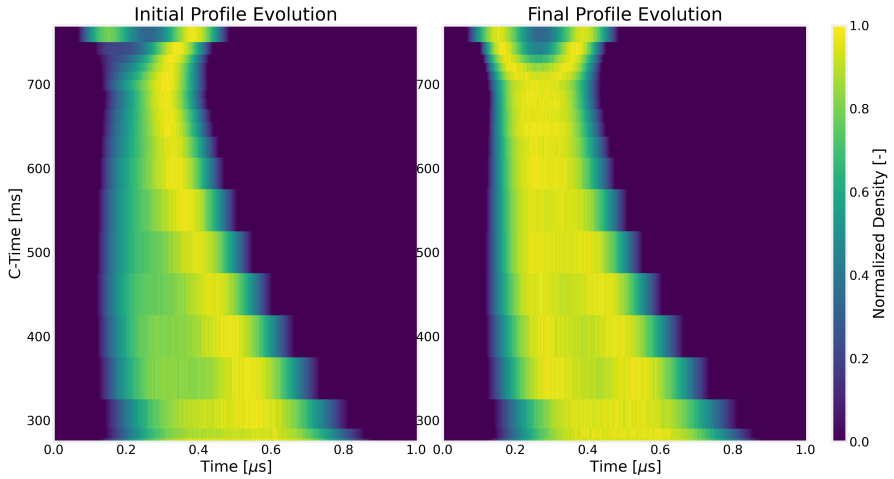


Figure 5.78: MTE normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

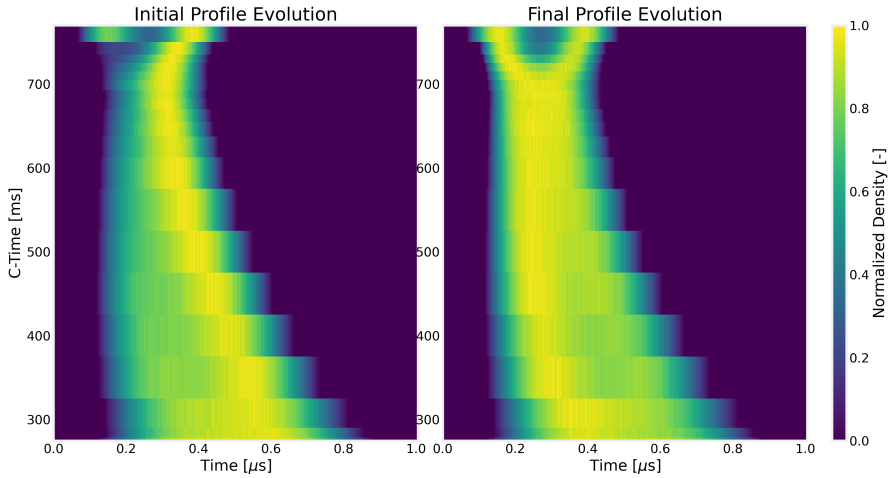


Figure 5.79: MTE normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right) for unoptimised model tested in the PSB.

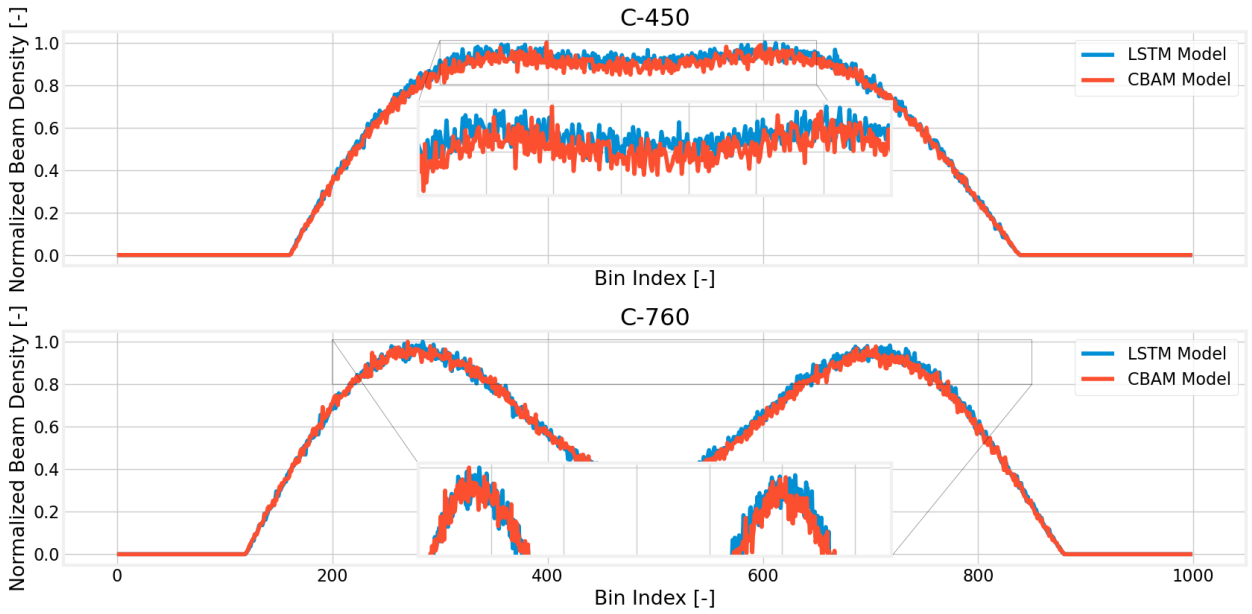


Figure 5.80: MTE CBAM vs LSTM-TD3 model solution comparison at selected C-Times with zoomed insets (C-450 and C-760)

TOF

Finally, for the TOF cycle the corrections and solutions are shown in Figure 5.81, while the corresponding waterfall plot is shown in Figure 5.82. Here there is practically no difference from the CBAM solution, except that there is a slightly higher (1° or less) phase in the from C-300 to C-400, and that the solution at C-700 is 2° higher and closer to the manually phased solution. Aside from this, all the profiles are symmetric and lengthened, and the difference in the profiles at these points is so small that it is not worth mentioning. Meanwhile, the unoptimised LSTM-TD3 agent heavily overestimates the Φ_2 value again, meaning that the profiles are skewed to the left, consistently drifting with positive corrections up to correction number 7 (see Figure B.19), which is becoming the predictable behaviour of the unoptimised agent. This tells us that the same behaviour should be expected in the PSB, which is the next and final discussion of this chapter.

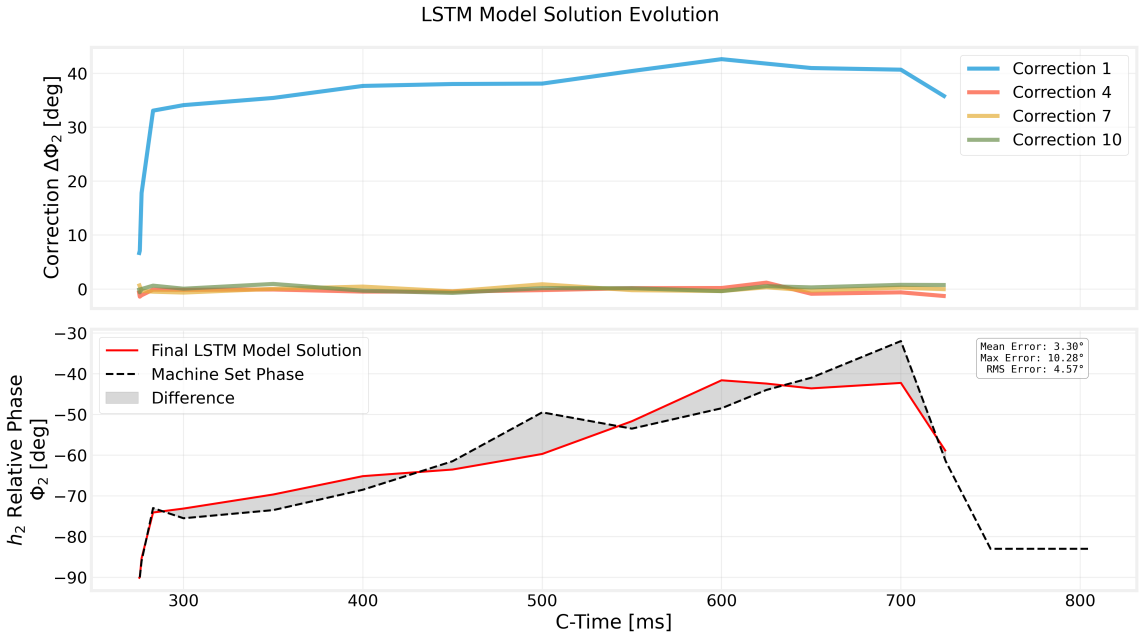


Figure 5.81: TOF phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

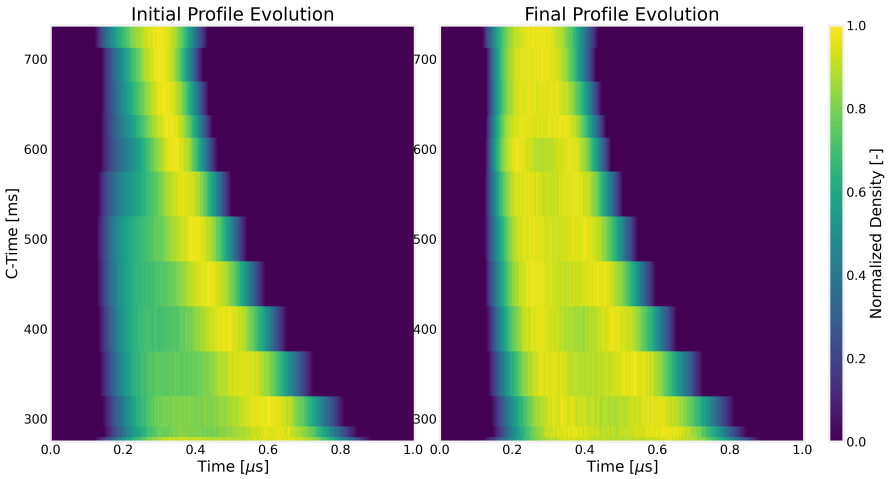


Figure 5.82: TOF normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

5.2.6. Validation Procedure: PSB Results

As mentioned previously, given the time constraints related to access to the CCC, there was no time to test the optimised network which is trained with baseline droop augmented profiles. As a result, an unoptimised model, which was not trained with augmented profiles, as this caused even worse performance, is tested in the PSB. Consequently, the results presented here are merely indicative and the analysis is catered towards how much the model's behaviour (which is already assumed to be sub-optimal) is replicated by the verification analysis. This is because if the same trends that occur in simulation are repeated for the validation results, then that gives us more confidence that the optimised LSTM-TD3 agent will be able to perform well in the PSB when implemented or tested in the near future. Additionally, the effects of not choosing the correct computation point are not discussed as they were analysed extensively for the CBAM model which had better performance than the unoptimised agent. However, the same effects still persist and cause slices in the waterfall plot where the profiles are considerably skewed. Finally, the model also achieved the sub-cycle latency target for the correction, taking roughly 20 seconds to acquire, preprocess, infer and correct the profile, as mentioned before, this can be improved with a better smoothing and reinterpolation mechanism. Again, by far the largest bottleneck is the reinterpolation and smoothing process, which takes roughly 1 second per computation point, while the inference takes less than a second to perform and send the phase setting to the PSB.

Thus, we will first analyse the results in the same order as in Section 5.1.7, starting with the 2025 BCMS variant.

BCMS 2025

For the 2025 BCMS variant, the performance was only acceptable for 1 of the 3 attempts, so it is the only attempt that will be shown here. This is consistent with lack of robustness with regards to initial conditions and erroneous corrections, as will be shown. The final waterfall plot produced under the Φ_2 program indicated in Figure 5.84b is shown in Figure 5.83. It can be appreciated in the final waterfall plot that asymmetries are present throughout the cycle, which is expected given the behaviour it had in simulation without any type of perturbations.

At injection, the unoptimised LSTM-TD3 agent constantly corrects positively and continues to do so until the end of the 10 corrections, thinking that the optimum lies at higher Φ_2 values. This is shown in the corrections made in Figure 5.84a. This is likely aggravated by the fact that the unoptimised model was not trained with profiles augmented with baseline droop. So, with the appearance of this droop in the PSB profiles, which was much more apparent for the profiles later in the cycle (at C-700), likely caused correction issues. In the first correction, the model goes the complete opposite direction, which is similar to what it does in the first correction in simulation (without any baseline droop) in Figure 5.70 at C-780. This effect does not occur for the optimised model, so this behaviour is not expected. After 9 corrections, the agent luckily manages to enter a region in Φ_2 in which BLM features are visible and that allows slight convergence, but this is not consistent behaviour as demonstrated by the fact that only 1 out of the 3 optimisations converged (with one of them having used 15 corrections instead of 10).

Additionally, this unoptimised agent was also expecting profiles that were not smoothed, so it could have also played an effect on this cycle and others, decreasing performance even more. This could be because it might have depended on the slight noisy features to be able to perform corrections. In an attempt to prove this effect, the model is tested for the BCMS cycle in simulation with profiles with droop and that are smoothed to see how the agent interacts with these corrections. The simulated profiles are shown in Figure 5.86 while the real reinterpolated profiles are shown in Figure 5.85, and it can be seen that they look similar.

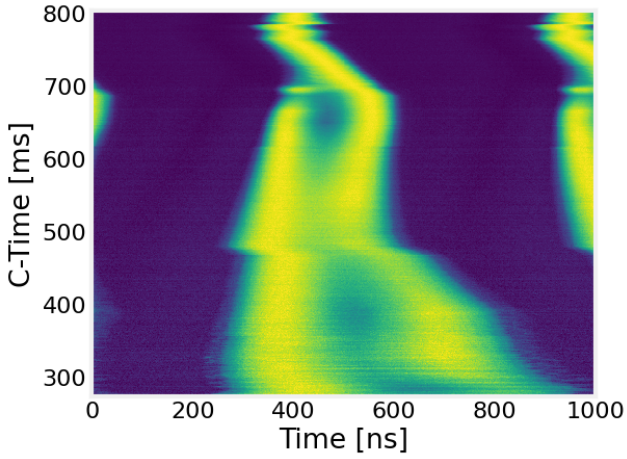
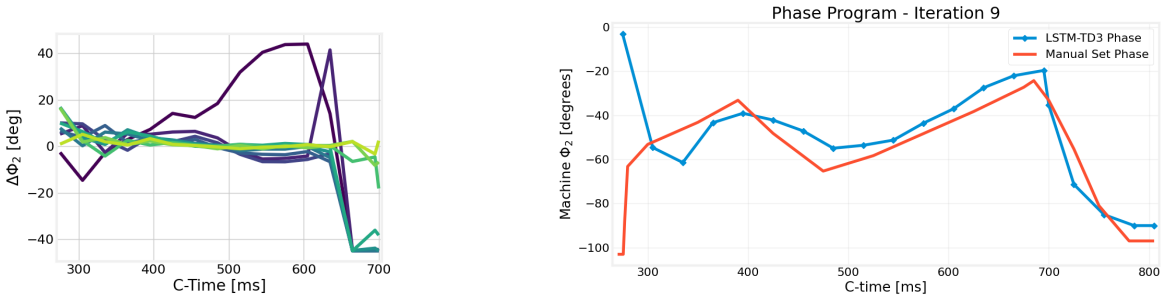


Figure 5.83: LSTM-TD3 BCMS 2025: final measured waterfall after applying the LSTM-TD3 solution



(a) Correction amplitude $\Delta\Phi_2$ per iteration across C-Time (yellow indicates corrections later in the cycle)

(b) Final Φ_2 program with the solution compared to the manually phased Φ_2 program

Figure 5.84: LSTM-TD3 BCMS 2025: agent corrections (left) and final solution (right)

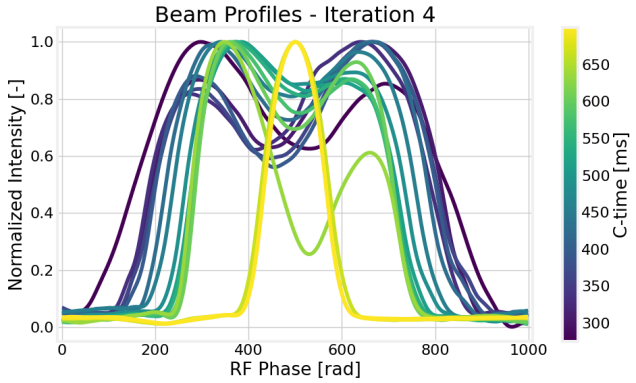


Figure 5.85: LSTM-TD3 BCMS 2025: Reinterpolated profiles at 4th correction

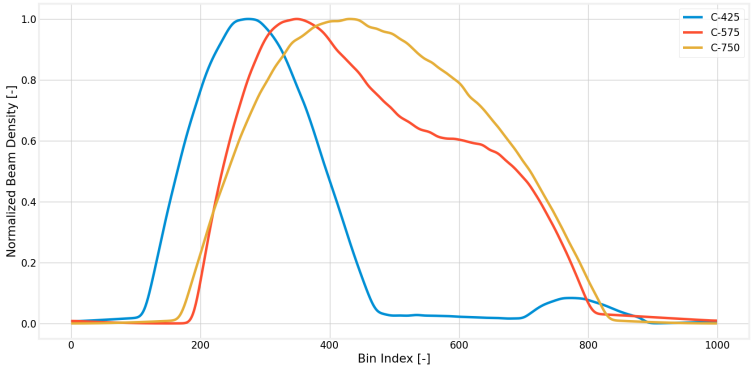
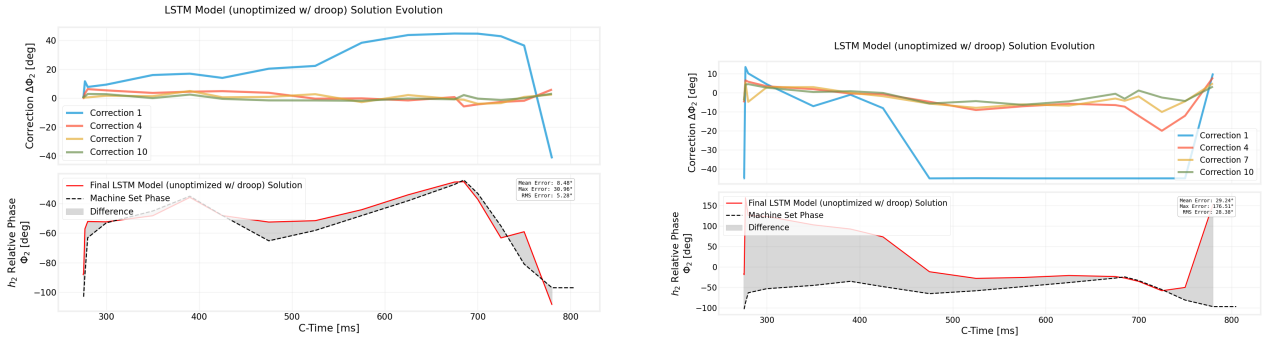


Figure 5.86: LSTM-TD3 BCMS 2025: Simulated droop for BCMS cycle

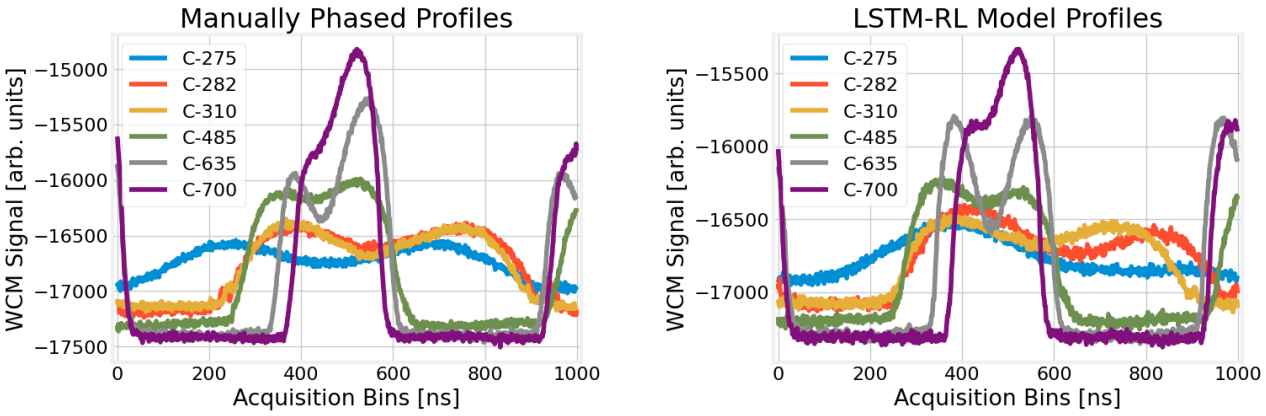
Using this simulated droop, it is possible to evaluate the unoptimised LSTM-TD3 agent on the simulated BCMS cycle. The results are shown in Figure 5.87 for two different starting phases $\Phi_2 = \pm 90°$ for all C-Times. As can be appreciated in the comparison of the solutions, while the solution converges some values that are closer to the optimal solution (like $\Phi_2 = -90°$), it does not for those that start further away. This effect is also evident when starting from $\Phi_2 = 0°$ and behaves as it does in the PSB: lack of convergence at injection and at C-700. The important takeaway here is that because it lacks robustness to converge while being independent of the starting point, it is the main driving reason behind the bad performance. Note that this effect of lack of convergence is not only due to the appearance of droop, it occurs in general for this unoptimised agent and it does not have the characteristics shown by the optimised agent in Figure 5.63 and 5.64, but it is aggravated by the appearance of droop in simulation.



(a) Initial $\Phi_2 = -90°$ **(b)** Initial $\Phi_2 = 90°$

Figure 5.87: LSTM-TD3 BCMS 2025: agent final solution in simulation augmented with baseline droop for starting at $\Phi_2 = -90°$ (left) and $\Phi_2 = 90°$ (right) for all C-Times

Nevertheless, roughly correct phasing was achieved for the profiles from C-500 to C-650 in 2 out of the 3 attempts. This again underlines the lack of convergence of the agent. However, suboptimal profiles were obtained from injection up to C-500 for both attempts which can be appreciated in the comparison of the profiles between the manually phased solution and the LSTM-TD3 solution Figure 5.88. All these profiles were also skewed to the left, much like the profiles in verification, which gives promise to the optimised model for which this does not happen in simulation. Thus, it can be said that the unoptimised model has an inherent bias towards positive corrections, which is only made worse and less stable by the appearance of baseline droop.



(a) Manual phasing **(b)** LSTM-TD3

Figure 5.88: LSTM-TD3 BCMS 2025: select profile comparison at certain C-Times for the manually phased and LSTM-TD3 solution

Furthermore, in the third attempt the phase was changing so drastically near C-700 with big changes in phase that caused particle losses of up to 20%. This was enough cause to terminate the optimisation immediately and make a switch towards the BCMS 2024 cycle. In addition, given the analysis presented and the simulation results for the optimised LSTM-TD3 agent and its performance relative to that of the CBAM model, it can be expected that it will perform better than the CBAM model in this phasing task.

BCMS 2024

As for the 2024 BCMS cycle, the results were considerably more stable. Nevertheless, in order to maintain safety and knowing that large oscillations were likely in the solution for the higher extraction energy cycles, the agent was only allowed to act from injection (C-275) to C-625, while at the rest of the points, the approximate

synchronous phase was used. Therefore, the final solution achieved and its corresponding measured waterfall plot are shown in Figure 5.89 and Figure 5.90, respectively.

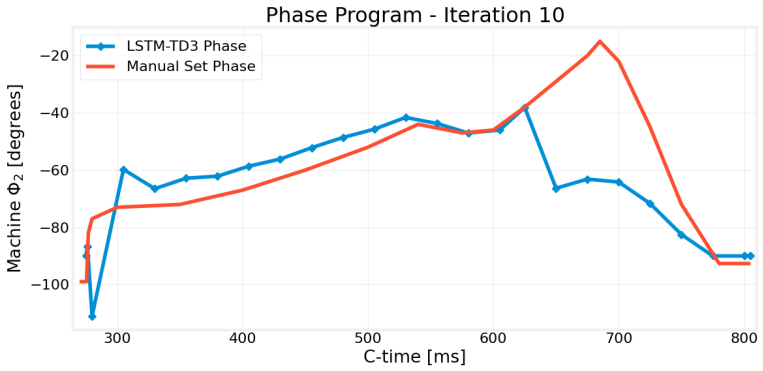


Figure 5.89: LSTM-TD3 BCMS 2024: final Φ_2 program with the solution compared to the manually phased Φ_2 program

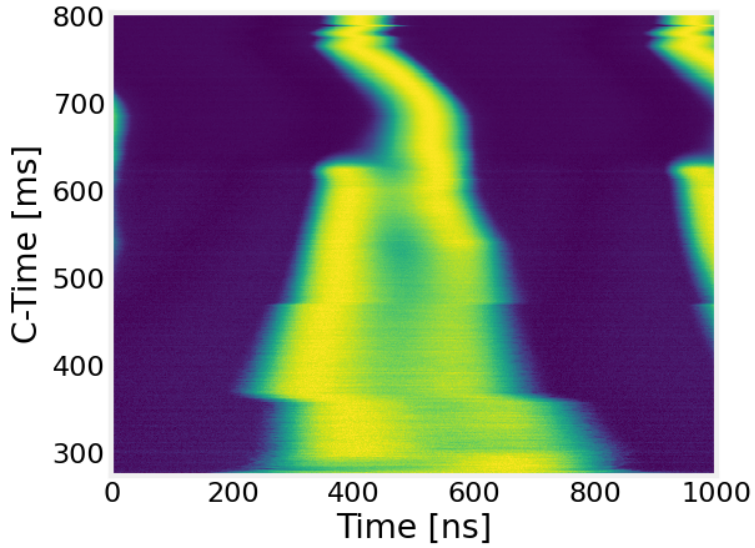


Figure 5.90: LSTM-TD3 BCMS 2024: final measured waterfall after applying the LSTM-TD3 solution

The same tendency to over-correct is persistent in this case, with overestimations throughout the cycle while also going the opposite direction at injection. Again, in Figure 5.91 iterative corrections were underestimated at injection and overestimated in the middle of the cycle. This is evident in the waterfall plot in Figure 5.90 where there is consistently a higher charge density on the left lobe, except at injection where the under-estimation of Φ_2 causes higher densities on the right. Furthermore, the effect of its rapid change from $-110°$ to $-60°$ can also be appreciated as in the waterfall plot, the bulk of the charge quickly shifts to the left. This can be clearly discerned when comparing the manually phased profiles to those of the agent in Figure 5.92, where now the profiles have only been shown in the range in which the optimisation has been performed. Although the agent gets closer, it still does not reach the optimum, as is expected. Furthermore, the same behaviour as in the simulated environment persists with over, and under corrections, and the incapability to converge with consistency. For example, in another attempt that was done for this cycle, the actionable range was extended to C-700, where now the model was incapable of converging at C-650, when it was able to do so previously for the BCMS 2025 case, which has the same acceleration rate and voltages at this C-Time. This again gives us more confidence in the correct functionality of the optimised LSTM-TD3 agent given that it does not display these behaviours in simulation, and is much more robust to these changes.

ISOLDE

Moving on to the ISOLDE results, these represent the case in which the performance was the best for the unoptimised agent, and the optimisation could be run up to C-750 like it was done for the CBAM model. Two different attempts were made in order to study the effect of smoothing as here the agent had some consistency in its results which allowed this comparison to be valid. As a result, in one of the attempts the smoothing is removed completely, while in the other it is maintained. An example of the profiles that the model is taking as input is shown in Figure 5.93 where the left panel shows the case where the reinterpolated profiles are smoothed and the right panel shows this without smoothing at the second iteration (so after one correction is applied).

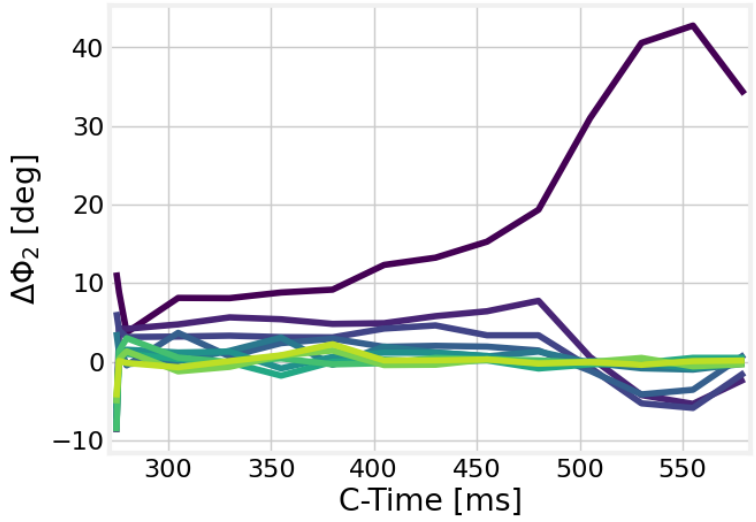
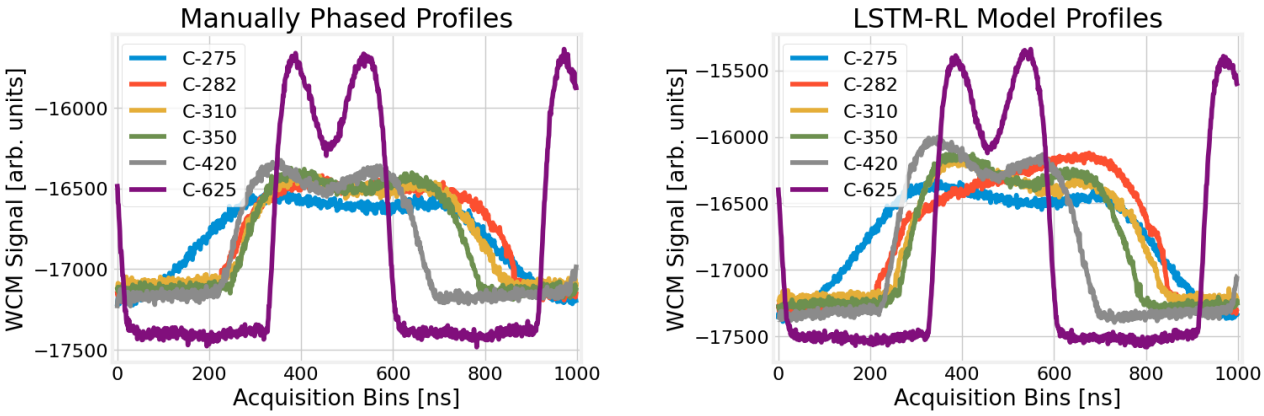


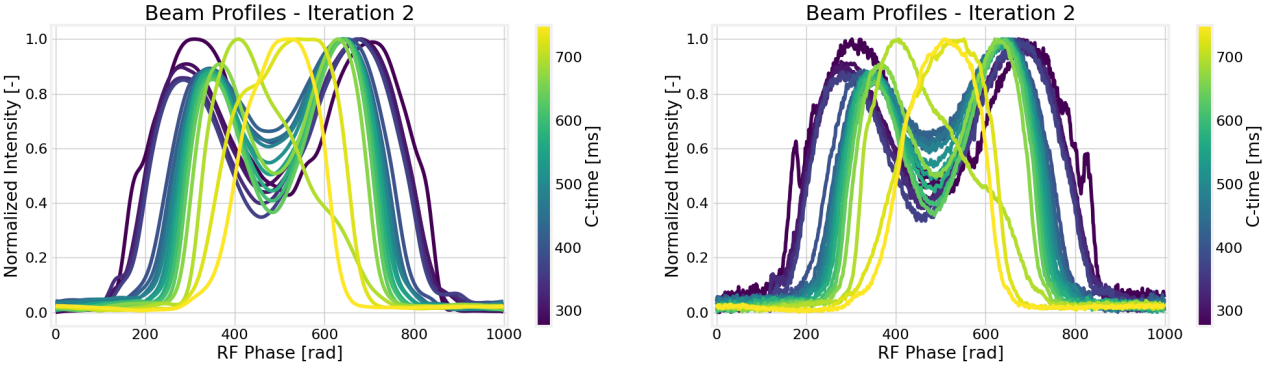
Figure 5.91: LSTM-TD3 BCMS 2024: correction amplitude $\Delta\Phi_2$ per iteration across C-Time (yellow indicates corrections later in the cycle)



(a) Manual phasing

(b) LSTM-TD3

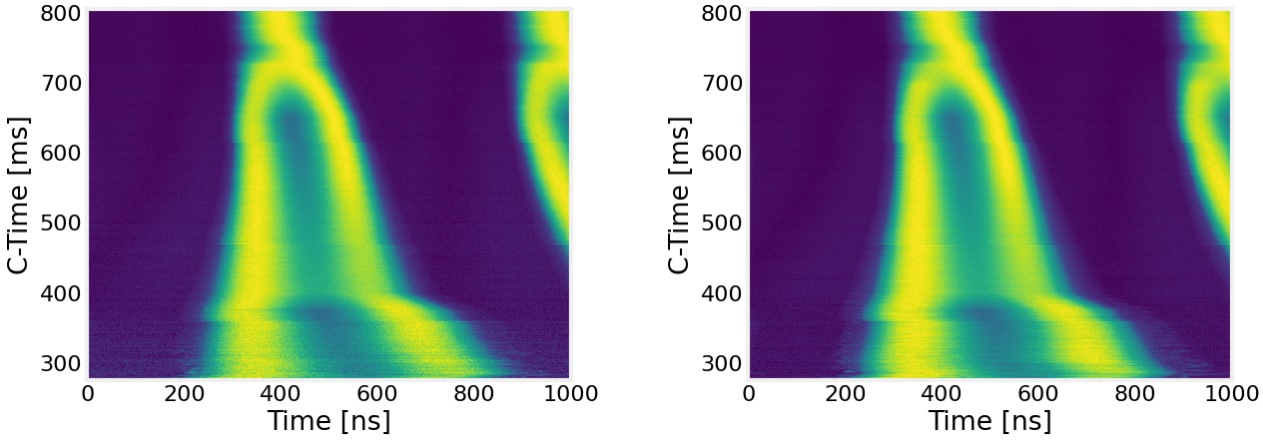
Figure 5.92: LSTM-TD3 BCMS 2024: select profile comparison at certain C-Times for the manually phased and LSTM-TD3 solution



(a) Smoothing

(b) No Smoothing

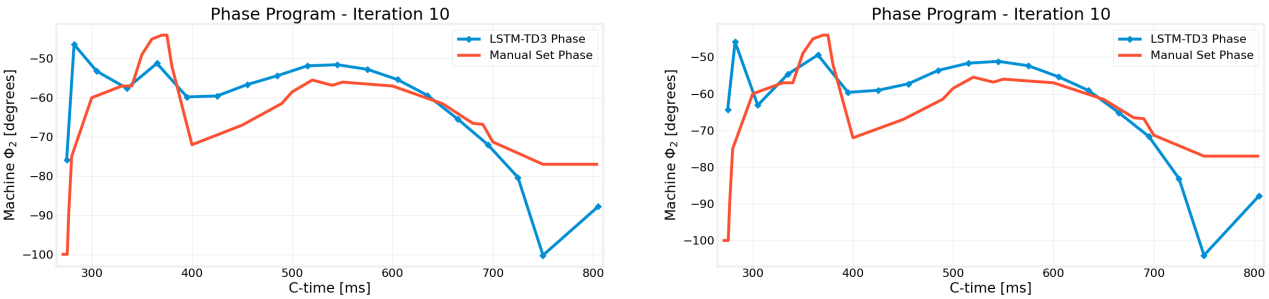
Figure 5.93: LSTM-TD3 ISOLDE: reinterpolated profiles at the second iteration with smoothing (left) and without smoothing (right)



(a) Smoothing **(b)** No Smoothing

Figure 5.94: LSTM-TD3 ISOLDE: final waterfall plots with smoothing (left) and without smoothing (right)

Meanwhile, the final waterfall plots for both cases are shown in Figure 5.94 in the same way. Note that it was possible to do this with an acceptable signal-to-noise ratio (SNR) because of the relatively high intensity used (80×10^{10} ppb). The Φ_2 programs that produced the waterfall plots are shown in Figure 5.95 where it can be seen that they are practically the same, but that the injection transients that are not smoothed out are causing a greater over-estimation of Φ_2 at injection and a larger under-estimation of Φ_2 at C-750. Nevertheless, an important point to make is that in the non-smoothed solution, Φ_2 is about 3° closer to the optimal solution in the kink formed at C-370. However, the difference is almost negligible, with the only one that truly matters being the larger overestimation at injection which clearly affects the injection process as demonstrated in Figure 5.94b where there seems to be a shift at the bottom of the plot.



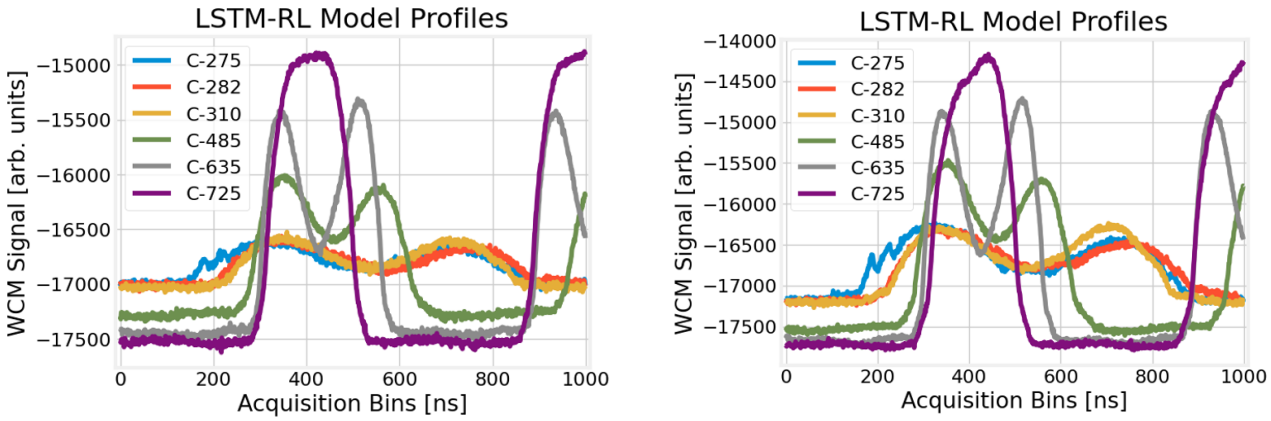
(a) Smoothing **(b)** No Smoothing

Figure 5.95: LSTM-TD3 ISOLDE: final Φ_2 program with the solution compared to the manually phased Φ_2 program for smoothing (left) and no smoothing (case)

Furthermore, when comparing the profiles that are produced with either method shown in Figure 5.96 (note the spurious reduction in SNR for the smoothing case at injection), with the manually phased profiles shown in Figure 5.97, it can be seen that both methods produce profiles of better phasing at the end of the cycle. However, at injection, the manually phased profiles are more symmetric. Furthermore, at extraction the smoothed method produces more lengthened and symmetric bunches, but this could simply be due to luck as both methods diverged in similar ways after C-700, and perhaps if more iterations were performed we could see the methods converge to a more reasonable value. What is also interesting to note is that the same type of pattern is achieved here as in the verification plot for the unoptimised model in Figure B.17. There is an overestimation of Φ_2 over the majority of the cycle, and an underestimation at extraction, which further validates our comparison between the two models.

MTE

For the splitting case with the MTE beam, the optimisation is run until C-750 as this is minimum required in order to be able to split the beam, while still avoiding the large oscillations which are known to occur at points where Φ_2 has minimal or no effect. Furthermore, given the precision that is needed for this particular beam type, 15 corrections are performed instead of 10. The final resulting Φ_2 program is shown in Figure 5.98 while the corresponding waterfall plot is in Figure 5.99. Note that it was not possible to normalize the profile due to noisy acquisitions that completely hinder visualization, but the final reinterpolated profiles are shown in Figure 5.100 to aid visualization.



(a) Smoothing **(b)** No Smoothing

Figure 5.96: LSTM-TD3 ISOLDE: select profile comparison at certain C-Times for the LSTM-TD3 solution with (left) and without smoothing (right)

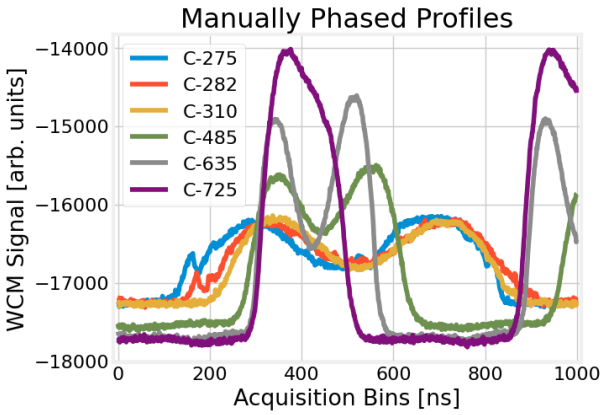


Figure 5.97: LSTM-TD3 ISOLDE: select profile comparison at certain C-Times for the manually phased solution

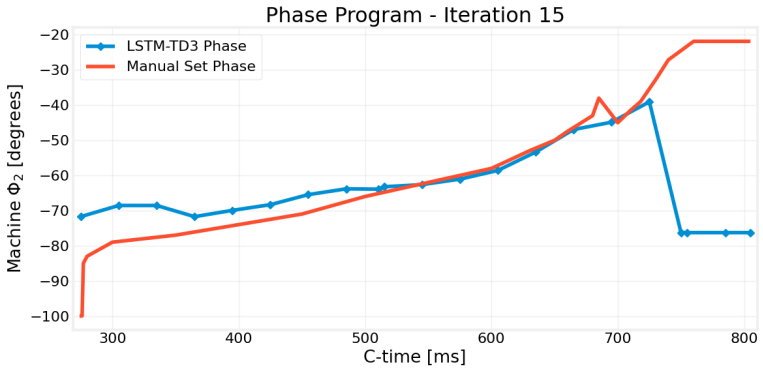


Figure 5.98: LSTM-TD3 MTE: final Φ_2 program with the solution compared to the manually phased Φ_2 program

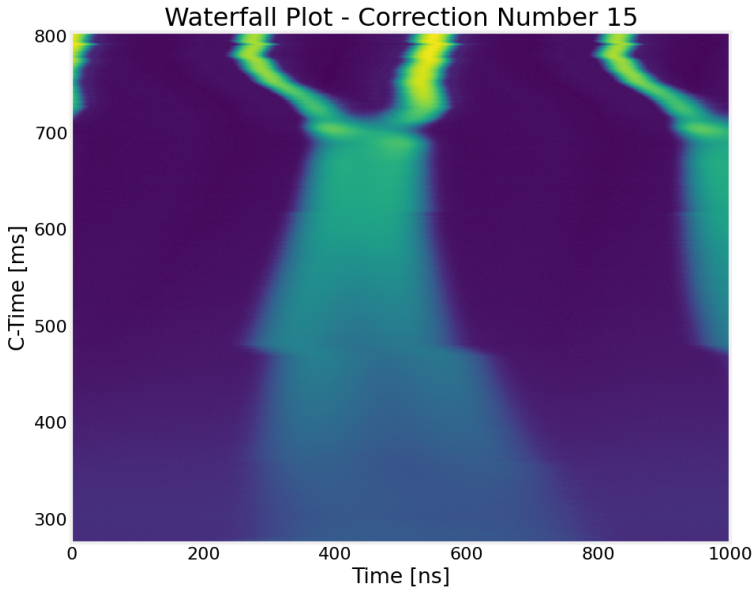


Figure 5.99: LSTM-TD3 MTE: final measured waterfall after applying the LSTM-TD3 solution

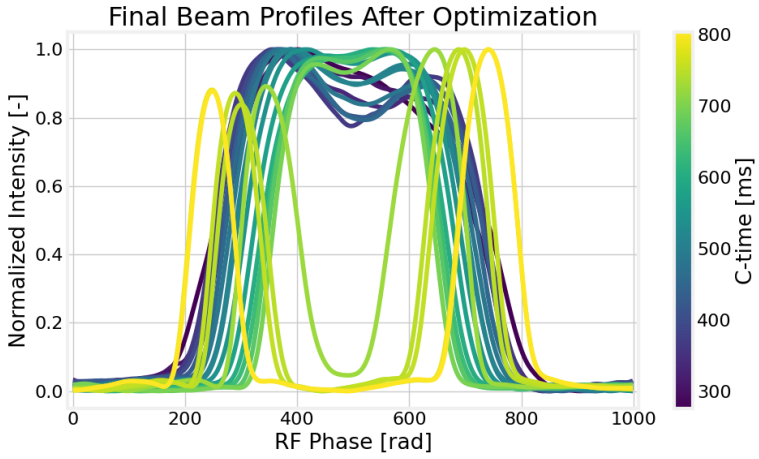


Figure 5.100: LSTM-TD3 MTE: final measured reinterpolated profiles after applying the LSTM-TD3 solution

Interestingly, the model performs almost exactly like in simulation, albeit with the issue that now the profiles are not automatically re-centred as they are in simulation at every individual C-Time. So the Φ_2 value at extraction has considerable repercussions on the bucket location since at this point $V_2 >> V_1$, making it dominate where the bucket will lie as now we also don't have any acceleration after about C-790. This effect can be visualized in the reinterpolated profiles shown for the fourth correction in Figure 5.101. Regardless, it can be seen through Figure 5.100 that the profiles, look similar to the profiles that are produced in simulation with the same overestimation of Φ_2 which causes the profiles to be skewed towards the left (i.e. a higher density of particles on the left). Surprisingly, looking at the simulated solution in Figure 5.77, the same overestimation occurs, but only from C-275 to C-500, and not until C-680 like in simulation. This means that the model is actually performing better with the machine data than it does with simulated data, for this particular beam type. This again outlines the lack of robustness of the unoptimised agent, since it is sensitive to small variations in the profiles and initial conditions. A plausible reason as to why it is able to get better profiles when tested in the PSB is solution seeding with a good initial guess provided by the approximate synchronous phase $\phi_{s_{approx}}$, and the fact that this unoptimised agent is notoriously sensitive to initial guesses. However, it could simply be due to the baseline droop shifting the corrections towards the negative direction and slightly correcting this, but the lack of consistency in the agent's corrections make discerning the reason difficult.

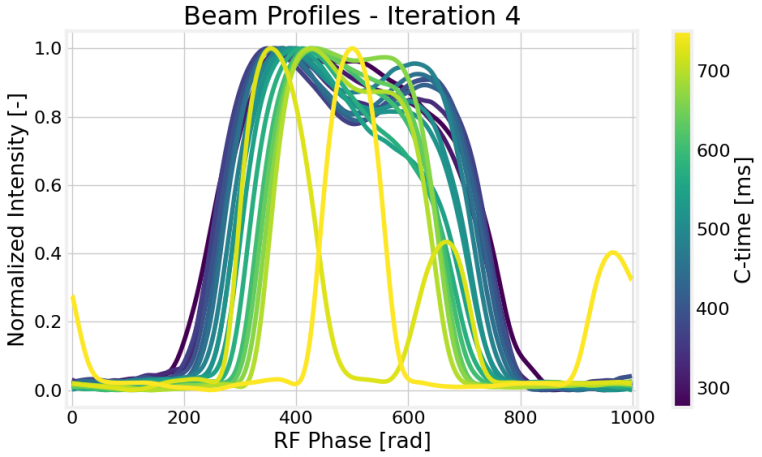


Figure 5.101: LSTM-TD3 MTE: reinterpolated profiles showing the bucket shift issue when large changes are made to Φ_2 when $V_2 >> V_1$

In summary, the agent was able to approach the correct phasing for the splitting procedure, but was not able to optimise it entirely. This is mainly determined by the particle density in either bunch, and as can be seen in Figure 5.100, the heights are different, which would be detrimental to the fixed target experiments down the line. Nevertheless, the predictable erroneous behaviour of the model on the basis of the simulation results, indicates that the optimised agent would perform the task much better, likely rivalling the performance of the CBAM model.

TOF

Finally, given that the TOF cycle is similar to that of the 2024 BCMS variant, it was already expected that it would not perform well up to C-750 or even C-700. Indeed, this was the case, with large change in Φ_2 at C-700 as shown in the final solution in Figure 5.102, which caused beam oscillations as seen in the corresponding measured waterfall plot in Figure 5.103. Although it has an overall negative result, this was the perfect scenario to test the safety measures introduced to mitigate the potential beam loss that this type of Φ_2 program can induce.

The safety measure detected a change in phase that was larger than 70° over a Δt interval, which meant that it was very likely to induce oscillations and instabilities. This procedure alerts the user, showing them the points where large phase changes are induced, and then sanitizes the solution by reverting to the initial estimate of $\phi_{s_{approx}}$. The entire procedure is shown in Figure 5.104, where the large phase change $\Delta\Phi_2$ is detected in Figure 5.104a and the sanitized solution is given in Figure 5.104b. Note that the sanitized Φ_2 program contains the point $(0, 0)$ as this point is required when setting the phase in the machine.

Consequently, the optimiser was run again, but now bounding the optimisation interval from C-275 to C-650, which is slightly larger optimisation interval than for the 2024 BCMS case. The fact that it could converge for this interval for the TOF cycle and not for 2024 BCMS one, which is almost the same beam but with a different intensity, showcases the lack of robustness of the unoptimised agent. The final solution is shown in Figure 5.105 and the corresponding waterfall is given in Figure 5.106. The solution follows the same behaviour as in the previous cases with an overestimation from injection up to C-500, which coincides with what happens in simulation (see Figure B.19). This again coincides with a larger charge density on the left side of the profiles, which is visible in the measured and simulated waterfall plot (see Figure B.20). This is also seen

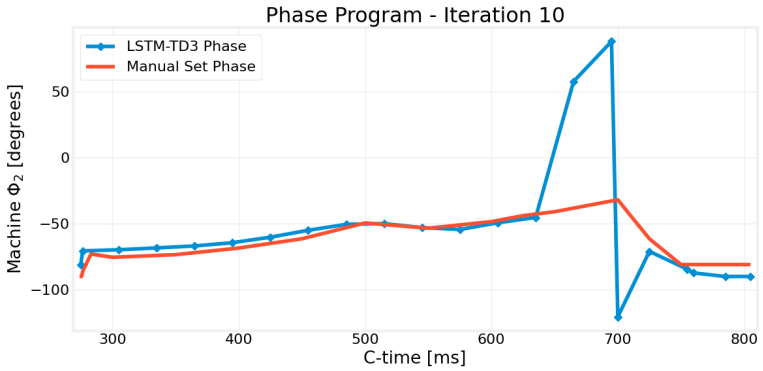


Figure 5.102: LSTM-TD3 TOF: final (incorrect) Φ_2 program with the solution compared to the manually phased Φ_2 program

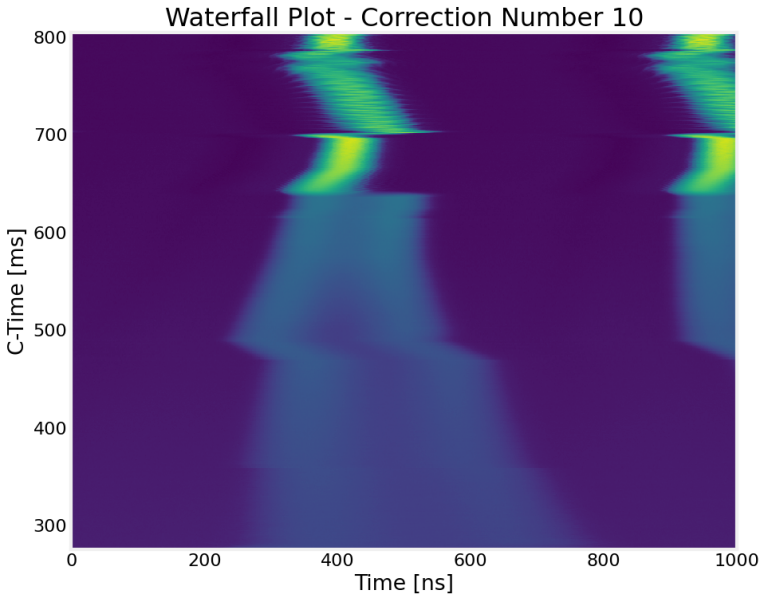
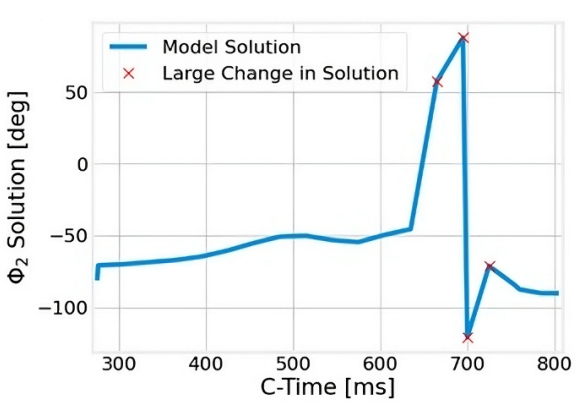
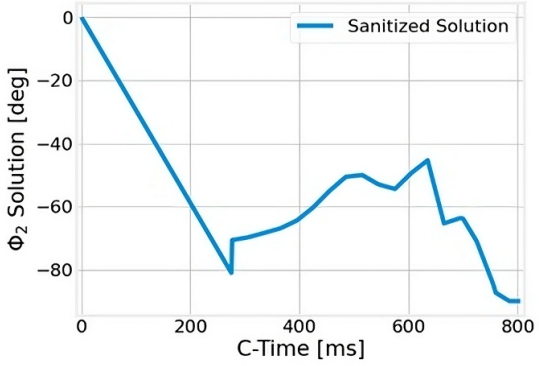


Figure 5.103: LSTM-TD3 TOF: final measured waterfall after applying the LSTM-TD3 (incorrect) solution with profile longitudinal beam oscillations visible after C-700



(a) Identification of large $\Delta\Phi_2$



(b) Sanitized solution: the point at $(0, 0)$ is required when setting the phase in the machine

Figure 5.104: LSTM-TD3 TOF: automatic solution sanitization when phase changes considerably over one time-step Δt

in the individual profiles at selected C-Times shown in Figure 5.107, where it can be seen that while the profile is well phased for the later points in the cycle (C-635), it is clearly asymmetric from injection up to C-500. However, this was expected from the simulated results, so it is good to validate those results, which again provides more confidence in the optimised LSTM-TD3 model in being able to correctly phase profiles throughout the cycle.

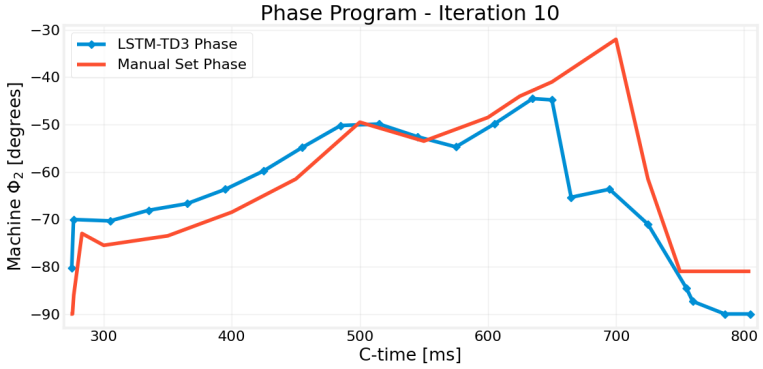


Figure 5.105: LSTM-TD3 TOF: final Φ_2 program with the solution compared to the manually phased Φ_2 program

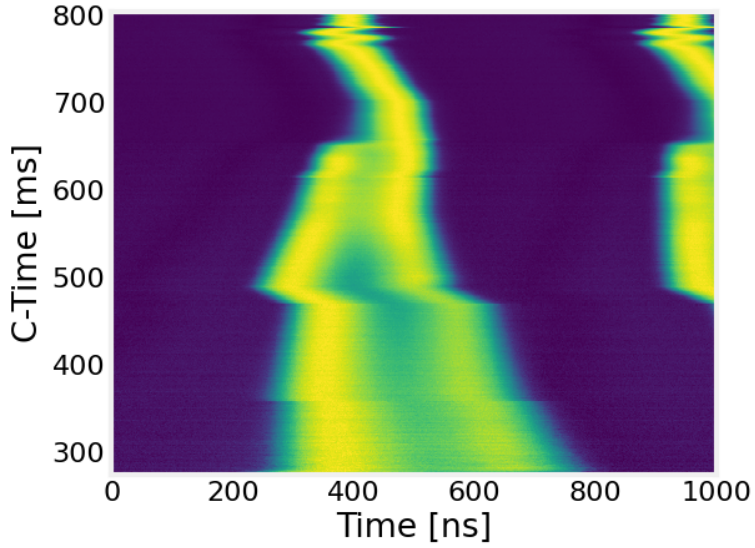


Figure 5.106: LSTM-TD3 TOF: final measured waterfall after applying the LSTM-TD3 solution

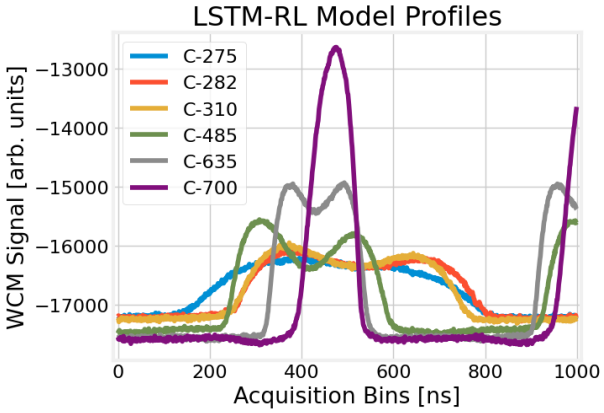


Figure 5.107: LSTM-TD3 TOF: select profile comparison at certain C-Times for the LSTM-TD3 solution

Concluding Remarks

In conclusion, the analysis presented showcases the limited performance of the unoptimised LSTM-TD3 agent without data augmentation when implemented in the PSB. Its behaviour in simulation and its behaviour

in experiments were analysed and the main reason behind this unsatisfactory performance is likely the lack of robustness of the agent to different initialization conditions (that exposes lack of correction consistency), different impedances not matching what is has been trained on, and its own corrections producing shortened profiles and longitudinal oscillations which it seems it cannot cope with. Additionally, this is aggravated by the fact that it has no thresholding capabilities to discern the important features in the profile, and is therefore susceptible to changes in the baseline droop of the profiles with it likely altering the correction produced.

Given these limitations, and the good correspondence of the simulation results to the behaviours witnessed in the PSB implementation, the optimised LSTM-TD3 agent, trained on augmented data with baseline droop and with thresholding capabilities, is expected to perform significantly better. Also, given the performance comparison presented in Section 5.2.5 and the enhanced robustness and convergence capabilities that the optimised LSTM-TD3 agent has over the CBAM model, it is expected that it will perform even better when tested in the PSB and other accelerators.

6

Triple-Harmonic Operation: 5D Optimisation

As explained in Chapter 2, in triple-harmonic operation the beam profile is shaped by five coupled control variables: the total RF voltage V_{total}, its second and third harmonic ratios r_2 and r_3, together with the relative phases of the second and third harmonics Φ_2 and Φ_3. This yields a 5D optimisation problem whose objective is to obtain a flat-topped, lengthened bunch while respecting longitudinal stability constraints. Building on the insights and tooling established in the 1D study (see Section 5.2), we extend the same RL-based approach to this higher-dimensional setting, reusing the agent design and many of the diagnostics while avoiding re-introducing already covered background.

This chapter presents: (i) the training setup used to learn a robust control policy for triple-harmonic optimisation; (ii) the shaped reward functions that enforce stability first and profile quality second; (iii) simulation results and verification; and (iv) a roadmap for deployment in the PSB. Where concepts have already been introduced, we point back to earlier sections and only state what is specific to the 5D case.

6.1. LSTM-TD3 RL Agent

6.1.1. Training

Training Setup & Considerations

We now revisit training of the LSTM-TD3 agent in light of the updated 5D optimisation problem. The recurrent off-policy agent from Section 5.2 is retained, but its role is gentler than the one imposed by the 1D case with all the variants: it learns to make small, informed adjustments in a space that is more complex but is still guided by clear physics objectives. Crucially, the Bayesian-optimiser study in Section 2.4 showed that the parameter mismatch can be summarised effectively by the KL divergence which is computed using the deviation of the selected, central portion of the profile where $\sim$94% of the charge is concentrated and the mean of the selected profile itself. This value is then divided by the length of the region of the profile where $\sim$94% of the charge is concentrated so as to encourage wider profiles. This lets us use L_{KL} as the primary learning signal and removes the need for the supervised optimal $(V_{total}, r_2, \dot{\mathcal{B}}) : \Phi_2(\mathrm{BLM})$ mapping that was necessary in the 1D case due to the amount of emittances that had to be considered. In this case, emittance is usually high when one wants to mitigate space charge, and this high emittance is practically assured by the injection mechanism of the PSB: longitudinal phase painting to fill the RF bucket as evenly as possible. As a result, we remove the filling factor from the equation and consider only filling factors (70 to 100% of the RF bucket). The agent therefore can directly optimise for flatter, more uniform profiles while exploration remains confined to simulation instead of having to explore the parameter space in the PSB.

This approach also makes more sense since at low emittances it is impossible to reach the desired flatness to reduce space charge, so it is something that would not be done in practice. Furthermore, this approach is also valid for attempting new types of beam production methods that make use of the triple harmonic operation like a triple splitting. This would be possible to do with the bunch lengthening approach, and then adiabatically ramping down the 1st and 2nd harmonic voltages to 0 such that the particles are driven to the centre of each triple harmonic bucket. This is of course contingent on if such a beam type would be beneficial or even desirable for the experiments or the LHC, but one application that comes to mind is the use of three bunches for the SFTPRO beam as the PS wants to have as constant as possible bunch intensity and a stable proton beam for continuous production of by-products. Injecting with three bunches into the PS would allow capture to be done with h_{12} instead of the h_8 which is currently used for the SFTPRO beam. Nevertheless, the beam is split again in the PS with the current setup, so it would have to be seen if it is even beneficial to do so.

Returning to the training setup, the observations used as input to the policy concatenate three elements: (i) the normalised longitudinal profile (it could be augmented with baseline droop or not); (ii) the current triple-harmonic parameters $(V_{total}, r_2, r_3, \Phi_2, \Phi_3)$ expressed as normalised parameters; and (iii) a normalised magnetic field ramp rate $\dot{\mathcal{B}}$, which gives information on the acceleration rate for the particles. We include only normalised parameters so that the learned policy can, in principle, transfer across accelerators or survive parameter-range changes (e.g., PSB voltage upgrades) without architectural changes. As a result, parameter and action ranges become a generalised feature and not a fixed feature of the policy, so as not to limit to a specific accelerator.

Actions are bounded, incremental updates to the five parameters with the parameter ranges and action ranges defined in Table 6.1. On the learning side, we keep prioritised replay with a single unified priority structure over parallel environments to speed up the training, together with a gentle annealing of importance weights along the training process for the PER buffers. Given how well it performed in the 1D case as shown in Section 5.2, a learnable soft-threshold gate attenuates low-importance regions of the bunch profile in both actor and critic before any neural network operations, stabilising value estimation under low signal-to-noise and baseline droop. Additionally, as for the optimised 1D LSTM-TD3 agent, the simulator augments profiles with controlled distortions (e.g., droop) and computes the RF bucket area at every step using the Solfege module, which is then woven into the reward function described next [29].

Table 6.1: Parameter domains and per-step action bounds for triple-harmonic optimisation

Parameter	Symbol	Range	Action per step
Total voltage	V_{total}	$[6,\ 20]\,\mathrm{kV}$	$\Delta V \in [-2,\ 2]\,\mathrm{kV}$
2nd-harmonic ratio	r_2	$[0,\ 5]$	$\Delta r_2 \in [-0.4,\ 0.4]$
3rd-harmonic ratio	r_3	$[0,\ 5]$	$\Delta r_3 \in [-0.4,\ 0.4]$
2nd-harmonic phase	Φ_2	$[0,\ 2\pi]\,\mathrm{rad}$	$\Delta\Phi_2 \in [-\frac{\pi}{4},\ \frac{\pi}{4}]\,\mathrm{rad}$
3rd-harmonic phase	Φ_3	$[0,\ 2\pi]\,\mathrm{rad}$	$\Delta\Phi_3 \in [-\frac{\pi}{4},\ \frac{\pi}{4}]\,\mathrm{rad}$

Also, phases are applied additively as in the 1D case and wrapped modulo 2π while voltage-like parameters (meaning the total voltage and the voltage ratios) are clipped to their domains after each action. However, the phases Φ_2 and Φ_3 in simulation are defined in an absolute manner, meaning that $\Phi_{ref} = 0$ as in Equation 2.34, not as how they are defined in the PSB with $\Phi_{ref} = \Phi_1$. Nevertheless, the following transformations can be applied to the phases in the simulation to make them relative to Φ_1 as in the PSB, where there is also a sign inversion due to the definition of the phases in the PSB, and it is shown in Equation 6.1 and Equation 6.2:

$$\Phi_2(PSB) = ((90^\circ - \Phi_2(simulation)) + 180^\circ) \mod 360^\circ - 180^\circ \tag{6.1}$$

$$\Phi_3(PSB) = ((-\Phi_3(simulation)) + 180^\circ) \mod 360^\circ - 180^\circ \tag{6.2}$$

These transformations were determined empirically and Equation 6.1 was used throughout validation in Section 5.1.7 and Section 5.2.6 to convert the model correction phases to the PSB phases, where the correction $\Delta\Phi_2$ was added to the Φ_2(simulation) value and then converted. Furthermore, the magnetic field ramp rate is normalised to be between 0 and 1, where 0 is no ramping and 1 is the maximum ramping rate encountered in the PSB operational magnetic cycles: $3.7\,\mathrm{T/s}$. This could have possibly been increased to the highest value in the recently developed magnetic cycles for space-charge mitigation ($\sim 4.37\,\mathrm{T/s}$) but it was kept at the current value to be more conservative with the operational cycles and ideally the model would be able to extrapolate to the higher ramping rates through the normalisation technique, or with proper training augmentation at these higher ramping rates (simply retraining the model with the higher ramping rates or using the current model and adding the higher ramping rates to the environment). Also, the number of different magnetic-field ramp rates that are considered is increased to 20 (in comparison to the 7 that were considered in the 1D case), to give the agent more sampling points on the magnetic-field ramp (see Figure 4.1) and be able to interpolate better between different acceleration rates. Furthermore, since the acceleration rate and the boolean on whether we augment the profile with droop or not are the only parameters that change every episode, it is also natural to increase the number of sampling points.

Reward Functions

Reward shaping follows a two-phase approach to target different optimisation objectives: first enforce longitudinal stability via an RF bucket-area constraint that is inversely proportional to the acceleration rate (since accelerating reduces the bucket area), then optimise the bunch shape using a KL divergence as the learning signal. We use two closely related variants during development: a simplified form that proved robust early on, and a modified form that adds an explicit peak-equalisation term to encourage wider, flatter tops. Both start with feasibility checks that discourage degenerate solutions (failed simulations and setting r_2 or r_3 to 0) before entering the other phases. The implementation is defined in Algorithm 9 for the simplified form and Algorithm 10 for the modified form.

The simplified reward focuses on robustness and learning speed:

$$R_{\text{simple}} = R_{\text{fail}} + R_V + R_A + R_{\text{KL}} + R_{\text{prog}} + R_{\text{act}} + R_{\text{peaks}}. \tag{6.3}$$

1. **Simulation robustness**: $R_{\text{fail}} = -10\mathbb{1}[\text{failed}]$ penalises when the simulation fails because matching the particle distribution fails, usually if there is no potential well for the particles to be in.

2. **Voltage sanity**: $R_V = -10\mathbb{1}[r_2 < 10^{-3} \ \vee \ r_3 < 10^{-3}]$ prevents collapsing higher-harmonic content, as otherwise we either revert to single or double harmonic operation, and usually the way that the RL agent learns to have sufficient bucket area is to increase V_1 as the other voltages reduce it due to them being in counter-phase (see Figure 2.23).

3. **Sufficient bucket area**: when we have less than the minimum required bucket area ($A < A_{\min}$), we give a negative reward proportional to the normalised deficit ($\delta_A = \min\left(1, \frac{A_{\min} - A}{A_{\min}}\right)$) and also give a reward for increasing the bucket area if we do not have sufficient bucket area: $R_A = -5\,\delta_A + \max\left(0, \min\left(1, 10\,\frac{A - A^{\text{prev}}}{A_{\min}}\right)\right)$.

4. **Profile quality (KL)**: when we have sufficient bucket area ($A \geq A_{\min}$), we use the KL divergence as the learning signal (we use 8 because $L_{\text{KL}} = 10^{-8}$ is basically impossible to achieve and to keep the reward in a range that can be used to understand how the agent is doing): $R_{\text{KL}} = -\frac{1}{2}(\log_{10} L_{\text{KL}} + 8)$.

5. **Progress bonus (bounded)**: we also reward progressing to lower KL divergence values in comparison to the previous step: $\Delta = \log_{10} L_{\text{KL}}^{\text{prev}} - \log_{10} L_{\text{KL}}, \quad R_{\text{prog}} = 2\,\tanh(\Delta)$. The hyperbolic tangent is used to prevent gradient explosions from rare large improvements.

6. **Action regularisation**: $R_{\text{act}} = -0.1\,\|a\|_1$ discourages large, oscillatory changes. This is a simple action penalty to prevent the agent from oscillating for no reason.

Meanwhile, the modified reward augments the shape phase by trying to equalise the peaks of the profile. In addition to R_{fail}, R_V, and R_A as above, its shaping includes a peak equalisation bonus:

$$R_{\text{peaks}} = \exp(-100\,\hat{\sigma}) \ \text{ if } \ |\Omega| > 0, \ \text{ else } \ 0 \tag{6.4}$$

where $\Omega = \{x : \mathbf{p_{norm}}(x) \geq 0.8\}$ is the domain of the profile that is above 80% of the maximum value of the profile, and $\hat{\sigma} = \text{std}(p_{norm}(\Omega))/(|\Omega|/1000)$ is the standard deviation of the "flat" part of the profile divided by the normalised length of the flat part. This is a peak equalisation bonus to encourage wide, equalised plateaus. As a result, using these reward functions it is possible to start training the agent to reach the flattest possible profile.

Algorithm 9 Simplified Reward for Triple Harmonic RL step

function REWARD($L_{\text{KL}}, A, A^{\text{prev}}, A_{\min}, \text{failed}, s, h$)
  $R \leftarrow 0$
  $R \leftarrow R - C_{\text{fail}} \cdot \mathbb{1}[\text{failed}]$ ▷ Phase 0: simulation didn't fail
  $R \leftarrow R - C_V \cdot \mathbb{1}[r_2 = 0 \vee r_3 = 0]$ ▷ Phase 0: didn't set either voltage ratio to 0
  if $A < A_{\min}$ **then** ▷ Phase 1: satisfy bucket area
    $R \leftarrow R - C_A \, \frac{A_{\min} - A}{A_{\min}}$
    if A^{prev} exists **then**
      $R \leftarrow R + C_{A_{prog}} \max\left(0, \frac{A - A^{\text{prev}}}{A_{\min}}\right)$ ▷ Phase 1: reward going in the right direction
    end if
  else ▷ Phase 2: shape profile via L_{KL}
    $R \leftarrow R - C_\ell \, (\log_{10}(L_{\text{KL}}) + 8)$ ▷ Phase 2: direct KL loss
    $\Delta \leftarrow \log_{10}(L_{\text{KL}}^{\text{prev}}) - \log_{10}(L_{\text{KL}})$
    $R \leftarrow R + C_{\text{prog}} \tanh(\Delta)$ ▷ Phase 2: reward progress
  end if
  return R
end function

Algorithm 10 Modified Reward for Triple Harmonic RL step

function REWARD($\mathbf{p_{norm}}, L_{\mathrm{KL}}, A, A^{\mathrm{prev}}, A_{\min}, \mathrm{failed}, s, h$)
  $R \leftarrow 0$
  $R \leftarrow R - C_{\mathrm{fail}} \cdot \mathbb{1}[\mathrm{failed}]$ ▷ Phase 0: simulation didn't fail
  $R \leftarrow R - C_V \cdot \mathbb{1}[r_2 = 0 \vee r_3 = 0]$ ▷ Phase 0: didn't set either voltage ratio to 0
  if $A < A_{\min}$ **then** ▷ Phase 1: satisfy bucket area
    $R \leftarrow R - C_A \frac{A_{\min} - A}{A_{\min}}$
    if A^{prev} exists **then**
      $R \leftarrow R + C_{A_{prog}} \max\left(0, \frac{A - A^{\mathrm{prev}}}{A_{\min}}\right)$ ▷ Phase 1: reward going in the right direction
    end if
  else ▷ Phase 2: shape profile via L_{KL}
    $R \leftarrow R - C_\ell \left(\log_{10}(L_{\mathrm{KL}}) + 8\right)$ ▷ Phase 2: direct KL loss
    $\Delta \leftarrow \log_{10}(L_{\mathrm{KL}}^{\mathrm{prev}}) - \log_{10}(L_{\mathrm{KL}})$
    $R \leftarrow R + C_{\mathrm{prog}} \tanh(\Delta)$ ▷ Phase 2: reward progress
    if $\Omega \neq \emptyset$ where $\Omega = \{x : \mathbf{p_{norm}}(x) \geq 0.8\}$ **then** ▷ Phase 3: peak equalisation
      $R \leftarrow R + C_{\mathrm{peaks}} \exp\left(-\mathrm{std}(\mathbf{p_{norm}}(\Omega)) \cdot 100/(|\Omega|/1000)\right)$ ▷ Phase 3: direct deviation of flat part
    end if
  end if
  return R
end function

Modified Training Parameters

We retain the same architecture from the 1D study (Small network complexity preset) and extrapolate the training hyperparameters to the 5D problem. The replay ratio r_{replay} is increased to 66 to allow more updates per transition given the fact that the transitions are more complicated due to the 5D action space and to provide more data usage. Exploration and target-smoothing noises are doubled with respect to the 1D optimum to accommodate the larger action space. This means that because the exploration noise was 1.524°, then $1.524 \times 2/360 \approx 0.75\%$, while the target smoothing noise was increased slightly to 1% of the total parameter range to be be more conservative in the target policy. Additionally, the LSTM history length is increased to 10 to allow the agent to see more of the past actions and states to make more informed decisions given the more complex transitions and interdependencies between the parameters. Finally, the amount of steps per episode is increased to 100 to give the agent more time to learn and explore the action space, which is still quite high for how fast we would like the agent to be in correcting the beam profile. However, this high number of steps per episode is also consistent with the fact that we would like the agent to be able to act in an online fashion, meaning that it should worry about long term stability in the profile, which can only be trained and assured if the agent acts for a certain acceleration rate for a long time. Nevertheless, given that these parameters are extrapolated from the 1D study, they should be subject to hyperparameter optimisation in the future to improve performance, as surely these are not the optimal parameters for the 5D problem. All the relevant training parameters are reported in Table 6.2.

Table 6.2: Extrapolated training parameters for 5D LSTM-TD3

Parameter	Value
Actor learning rate	0.7×10^{-5}
Critic learning rate	7×10^{-5}
Discount factor γ	0.971
Target Polyak factor	0.994
Exploration noise (σ)	0.75% of the action space
Target policy smoothing noise (σ)	1.0% of the action space
Batch size	128
Policy update delay	3
Replay ratio	65
LSTM history length	10
Network complexity preset	Small

Training Results

We trained two agents in the 5D setting, one with the simplified reward and one with the modified reward that adds a peak-equalisation term. After quickly learning to satisfy the bucket-area constraint, the modified-reward

agent degraded: it frequently collapsed to $r_2 \to 0$ or $r_3 \to 0$ that defeat the purpose of using triple-harmonic operation and accrued large negative returns per episode due to this. Thus, the reason for its quick learning to satisfy the bucket-area constraint was apparent: it was defaulting to single-harmonic operation with a large total voltage. Furthermore, warm-starting the modified-reward agent from the simplified-reward policy when already plateaued in its rewards, meaning starting from an already functional model (but not perfect), did not help. The peak-equalisation term increased the rewards for early updates (since the reward is only positive) and steered learning away from reducing L_{KL} to try to improve this reward component instead.

To be concise and considering the lacking performance of the complete training for the modified reward function, it is not shown (it is simply a stagnant reward), but these effects for the warm-started modified-reward agent are visible in Figure 6.1, where the simple-reward agent shows a consistent improvement trend, while the modified-reward curve rises initially (due to the extra peak bonus) and then decays. Note the x-axis is relative wall time (hours), not steps, to account for two external training interruptions which also explains the reset in the modified-reward curve (starts from a higher value due to the warm-start). The sensitivity of the training to the peak-equalisation reward shows that it can have competing effects with the KL divergence reward and bias exploration toward profiles that are not in the flattest triple-harmonic configuration (as was indicated by the Bayesian Optimisation studies performed shown in Section 2.4).

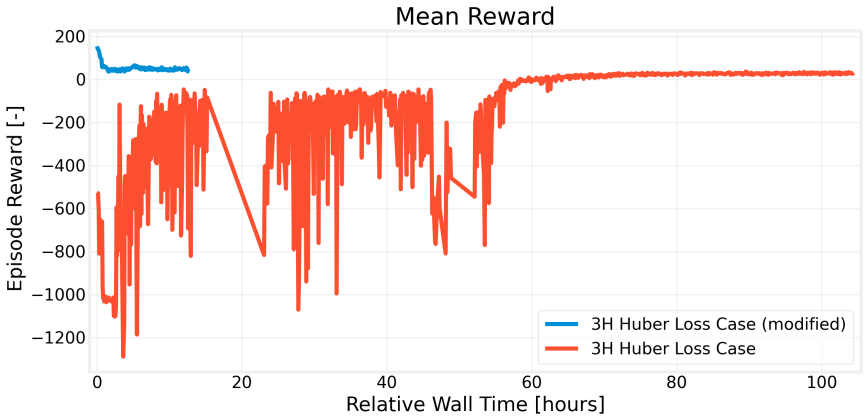


Figure 6.1: Reward curves for the 5D LSTM-TD3 agent with the simple and (warm-started) modified reward functions

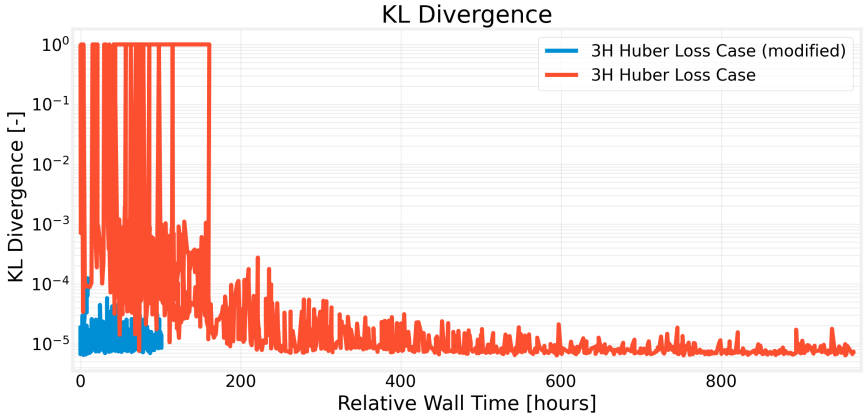


Figure 6.2: KL divergence curves for the 5D LSTM-TD3 agent with the simple and (warm-started) modified reward functions

The trend in Figure 6.2 reinforces this interpretation: the KL divergence for the warm-started, modified reward agent increases in standard deviation within a few updates to the network relative to the simplified baseline which already plateaued, indicating worse profiles. A plausible mechanism is reward leaking into the episode reward from the peak-equalisation term $\mathcal{R}_{\mathrm{peaks}}$ in regimes where only a wrongly set BLM operation in triple harmonic yields a flatter portion of the profile while the other portion is significantly less flat. Given the amount of versatility of triple-harmonic operation, this can occur with relative ease and the agent can collect peak rewards without improving the global flatness.

Beyond scalar training curves, the exploration strategies of each agent differ. The modified-reward agent basically lacks effective exploration capabilities due to how much the peak-equalisation term deteriorates the learning process. Under the simplified reward, early exploration proceeds with coordinated, oscillatory probing

of multiple parameters within a bounded envelope as can be seen in the parameter history in Figure 6.3. This figure is a frame in the animation that is used to verify the training progress of the agent (aside from the curves logged into a TensorBoard file), and it shows the profile, the parameter history for V_1, V_2, V_3, Φ_2, and Φ_3 after each correction, the KL divergence and its history and the current and cumulative rewards. Additionally, it also indicates the acceleration rate through the "B_dot" value in the title and the current epoch when the diagnostic animation was rendered. Back to the analysis, the pattern showed in the parameter history suggests that the agent is learning the coupled sensitivities between V_{total}, r_2, r_3, Φ_2, and Φ_3 by varying them simultaneously but at different rates (different frequencies in the oscillations) as can be seen in the figure.

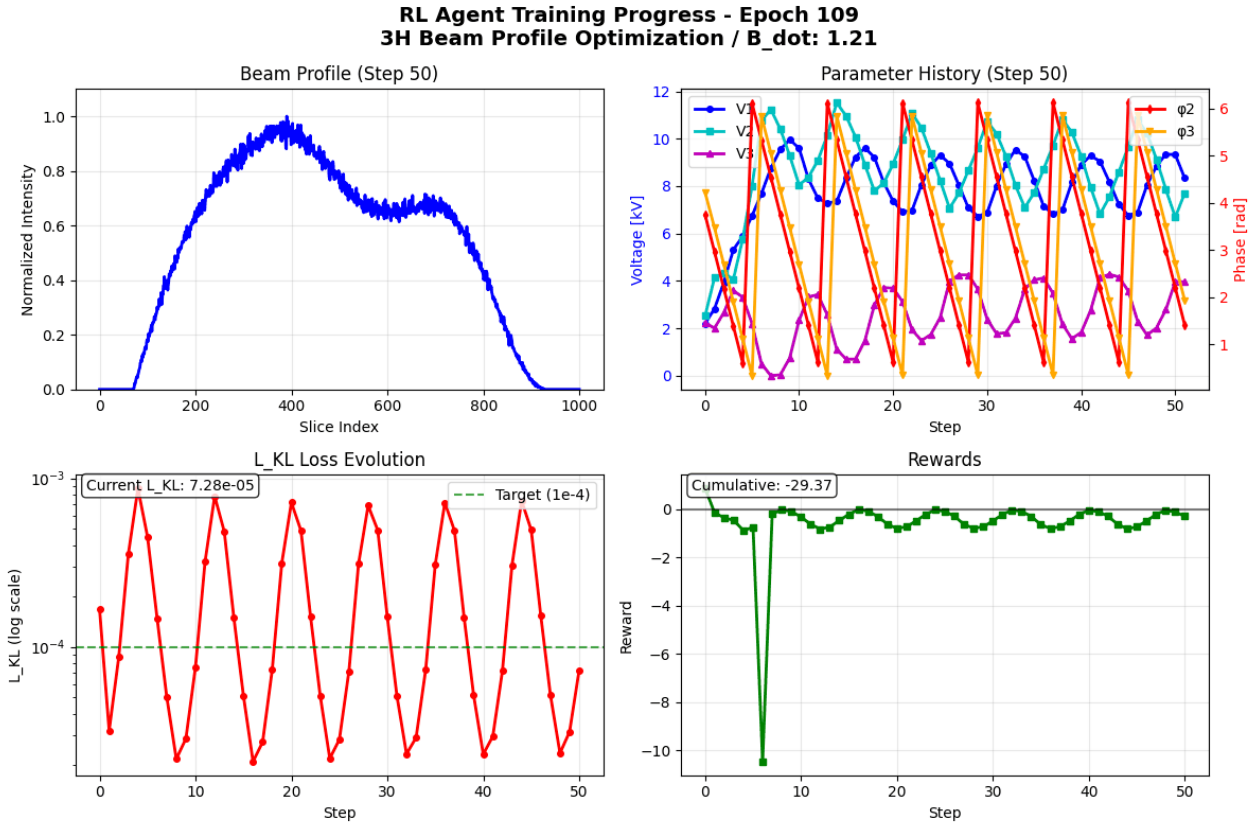


Figure 6.3: Early training of the 5D LSTM-TD3 agent with the simplified reward function (shown for 50/100 steps only)

As learning progresses (around epoch 250), the agent begins to stabilize near a useful region but still oscillates as it calibrates step sizes as is shown in the more jagged oscillations in Figure 6.4. By roughly epoch 500, adjustments become fine-grained and consistent, and after quickly reaching the optimum (within about 10 to 20 steps) and then stabilizing, parameters remain near their operational optima for the remainder of training as is shown in Figure 6.5. It can be seen that while the KL divergence measure does not equalize the peaks exactly as one desires in BLM, it gets very close to doing so, with only slight variations required to reach the optimum (mainly to the phasing Φ_2 and Φ_3).

Taken together, the results indicate that several aspects of the training that need to be considered:

1. KL-centric reward shaping is sufficient to get almost all the way to the optimum and much more robust when bucket-area feasibility is enforced first, confirming what was indicated by the Bayesian Optimisation studies performed shown in Section 2.4. However, it could be argued that the current methodology of computing the KL divergence is not the most optimal way to get the peaks to match. Instead of using the mean of the selected, central profile, one could use the maximum of the selected profile as the point to measure the deviations from. This would simultaneously ensure that the valleys in the profile are minimized and symmetric (since the outliers in the profile would have larger deviations), and also that the peaks are matched to ensure that they do not deviate too much from the target height.

2. Using plateau metrics and reward shaping should be introduced cautiously and, ideally, conditionally (e.g., apply peak equalisation only when the profile exhibits three peaks and r_2, r_3 are non-negligible). Of course this would imply more computational effort and needing robust peak finding algorithms implemented in the training procedure, which is why the modification of the KL divergence to use the maximum of the selected profile is a good compromise.

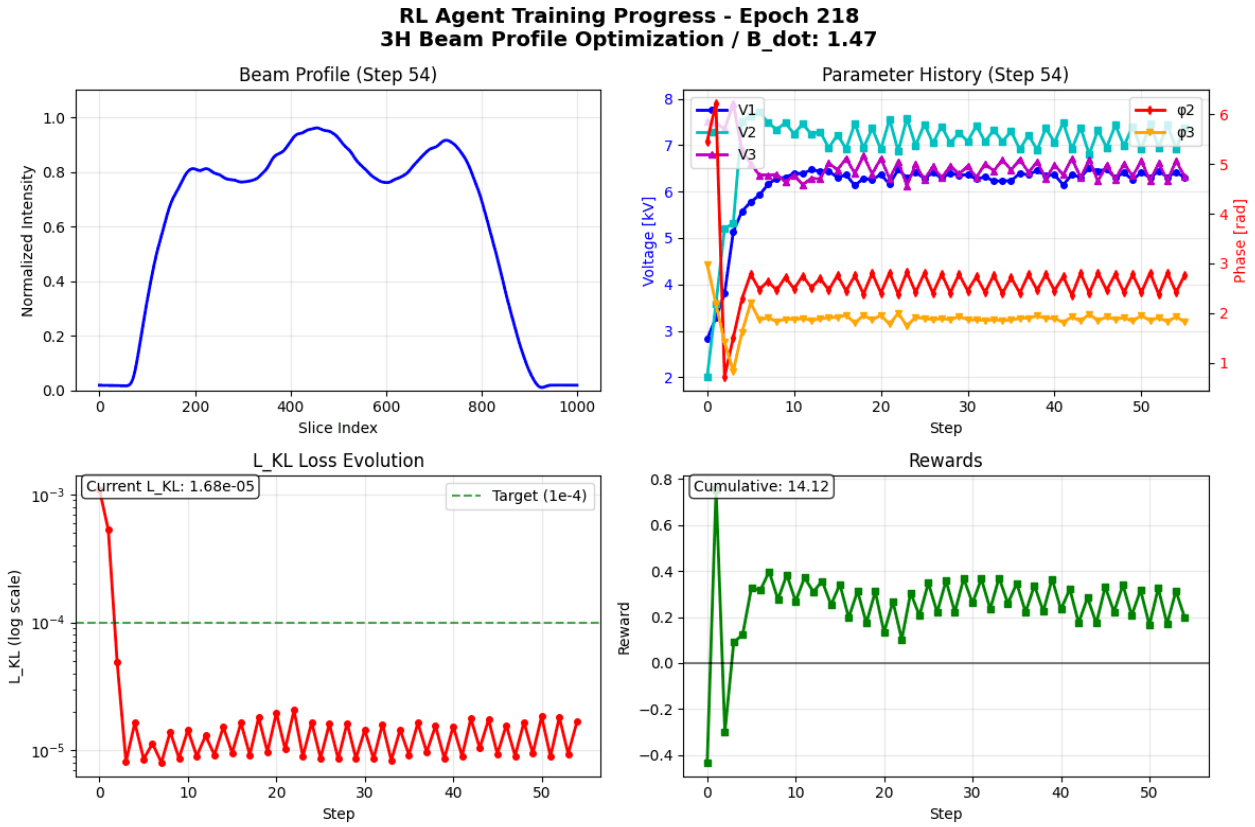


Figure 6.4: Improvement stages of the 5D LSTM-TD3 agent with the simplified reward function (shown for 54/100 steps only)

3. Curriculum elements, such as gradually tightening the bucket-area threshold when needed or introducing the peak term only after a KL target is reached, may further stabilize learning.

4. Adding the KL divergence to the state of the agent could help to stabilize learning and improve the performance of the agent as it has direct access to what it is trying to achieve. It would also be beneficial in an operational point of view as the agent would be able to see how the profiles changes in PSB and adjust its parameters accordingly.

With training concluded, we proceed to evaluate the agent on operational magnetic-field programmes for 1.4 GeV (ISOLDE) and 2.0 GeV beams (BCMS, TOF, SFTPRO, etc.), and on recently proposed exotic cycles for space-charge mitigation. These cases are characterised by their $\mathcal{B}(t)$ and $\dot{\mathcal{B}}(t)$ profiles, which the policy will attempt to exploit via the normalised $\dot{\mathcal{B}}$ feature.

6.1.2. Verification Analysis

Due to the relatively short time that was available to train the agent, it was not possible to test it in the PSB, but the foundations presented here in the verification analysis are laying the groundwork for future testing. For all the cases that will be shown here, the parameters V_{total}, r_2, r_3, Φ_2, and Φ_3 are initialized to random values within their acceptable ranges (see Table 6.1). This is why in some cases there is no existing bucket area initially to match the profile, as the initial parameters are random. Additionally, the triple-harmonic operation is maintained for the entire cycle (from injection to extraction) for all the cases, even though this is not necessary in practice, but it is done to give an indication of the capability of the model to handle the entire cycle.

The plots that are analysed for each case are:

- The initial and final waterfall plots, similar to those presented in Section 5.1.5 and Section 5.2.5, to show how the profile evolves throughout the cycle.

- The evolution of the voltages V_1, V_2 and V_3 throughout the cycle to show how the agent is able to control the voltages to achieve sufficient bucket area and the desired profile shape.

- The evolution of the phases Φ_2 and Φ_3 throughout the cycle to show how the agent is able to control the phases to achieve the BLM operation.

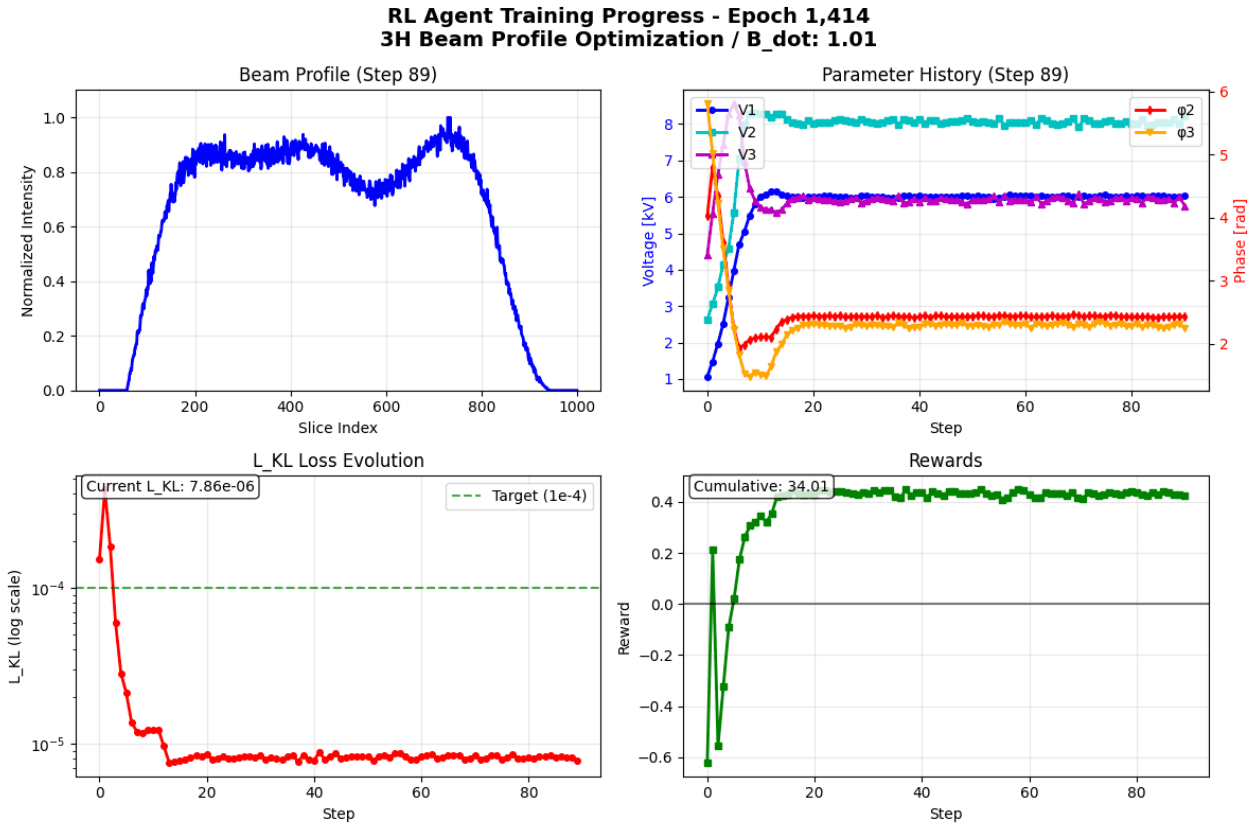


Figure 6.5: Late training stages of the 5D LSTM-TD3 agent with the simplified reward function (shown for 89/100 steps only)

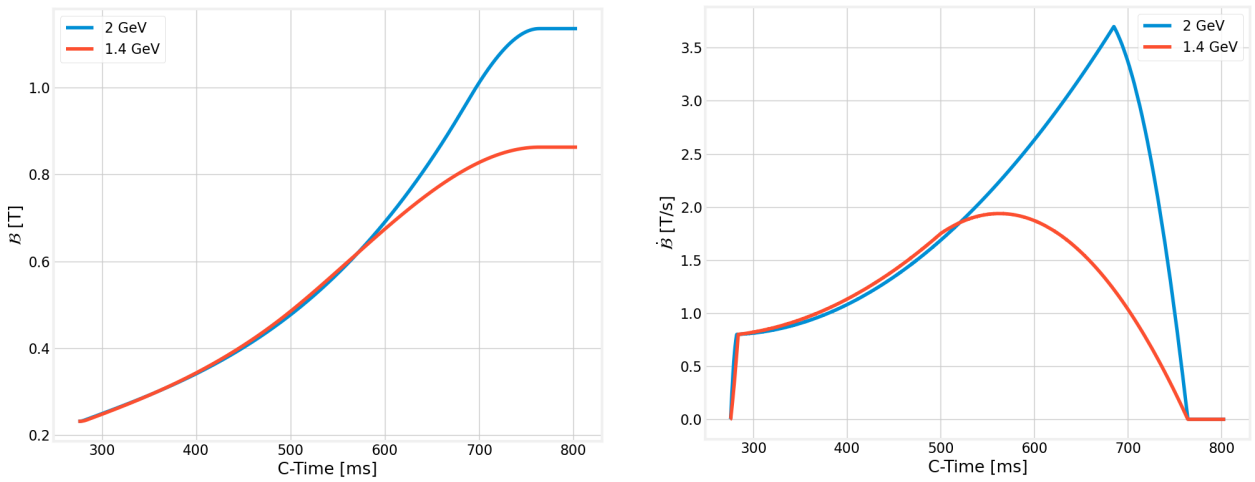
- The parameters ranges that are explored by the agent throughout the 40 corrections performed and the initial and final parameter values to show how the agent is able to reach the optimal parameters by approaching them from randomized initial conditions. This allows us to analyse the dynamic behaviour of the agent in learning the interdependencies between the parameters and if it is able to progressively move in the right direction towards the optimal parameters and how.

Operational Cycles

We start by showing the results for a 1.4 GeV and 2 GeV extraction energy beam, as these are the ones that use triple-harmonic operationally, and seeing how it performs with these magnetic field programmes is the most pressing concern for the applicability of the model in the PSB. Thus, the magnetic fields are shown in Figure 6.6a, while the ramp rates are shown in Figure 6.6b. It can be seen that the magnetic fields are monotonically increasing, while the ramp rates have a bit more structure to them, with the 2 GeV cycle having a kink at C-720 which leads to changes in the acceleration rate of the bunch and demands more careful control of the triple-harmonic parameters.

For the 1.4 GeV beam the profile evolution displayed through the waterfall plots is shown in Figure 6.7. It can be seen that the agent is able to achieve the desired profile shape and the bunch lengthening is achieved throughout the cycle. The phases and voltages are shown in Figure 6.8, where it can be seen that they progress smoothly throughout the cycle, indicating that they are consistent with the change in the magnetic field without any abrupt jumps. The parameters ranges explored by the agent are shown in Figure 6.9. It took only 10 iterations to converge to the optimal parameters for the 1.4 GeV beam, which is a testament to the effectiveness of the agent in learning the interdependencies between the parameters and being able to progressively move in the right direction towards the optimal parameters, which would mean roughly 5 minutes of optimisation when using only 1 slot in the super-cycle.

It can be seen that the agent is able to achieve the desired profiles without diverging considerably from the final optimal results in its corrections and progressively move in the right direction towards the optimal parameters. This is demonstrated by the fact that the initial parameters are usually on the bounds of the explored parameter ranges, and the parameter bounds are considerably closer to the optimal values on the bounds opposite to the initial parameters as observed in Figure 6.9. This is especially important for the voltages, since it starts at a point that is lower from the optimal values for V_1, which indicates that the bucket area is not sufficient since V_1 is the voltage that drives the increase of the bucket area.



(a) Magnetic field $\mathcal{B}(t)$

(b) Magnetic field ramp rate $\dot{\mathcal{B}}(t)$

Figure 6.6: Magnetic field programmes $\mathcal{B}(t)$ and ramp rate $\dot{\mathcal{B}}(t)$ for the 1.4 GeV and 2.0 GeV extraction energy operational beams

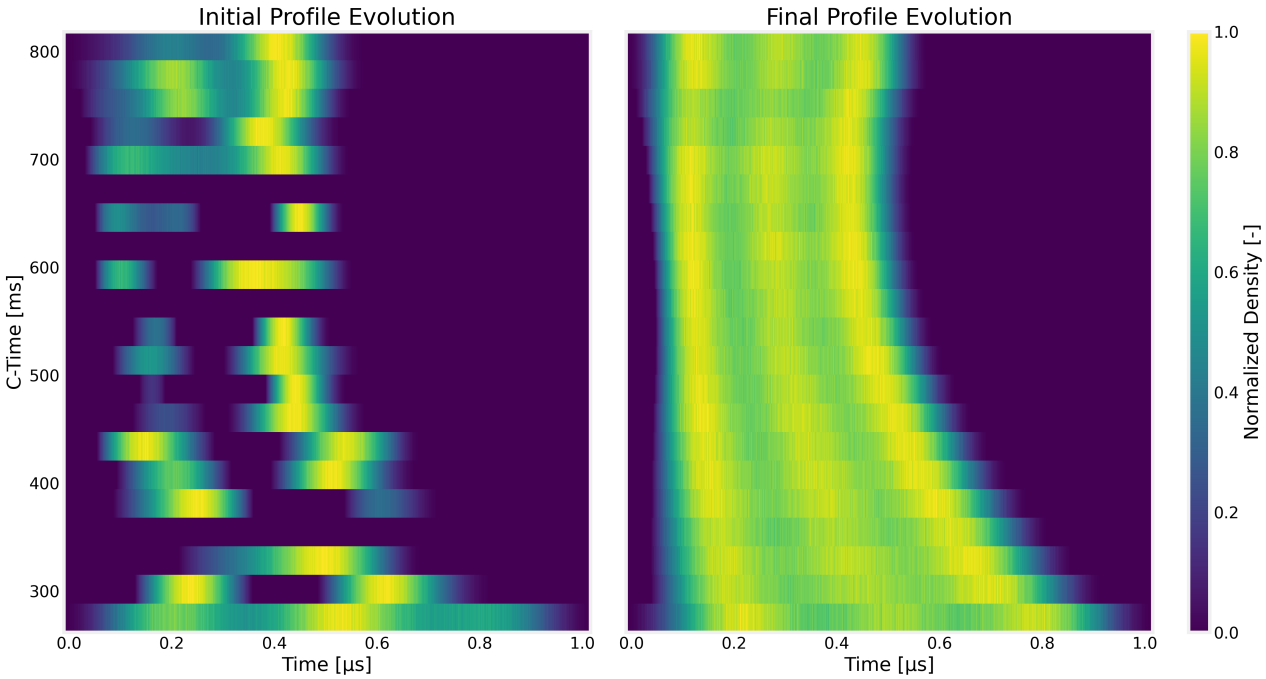
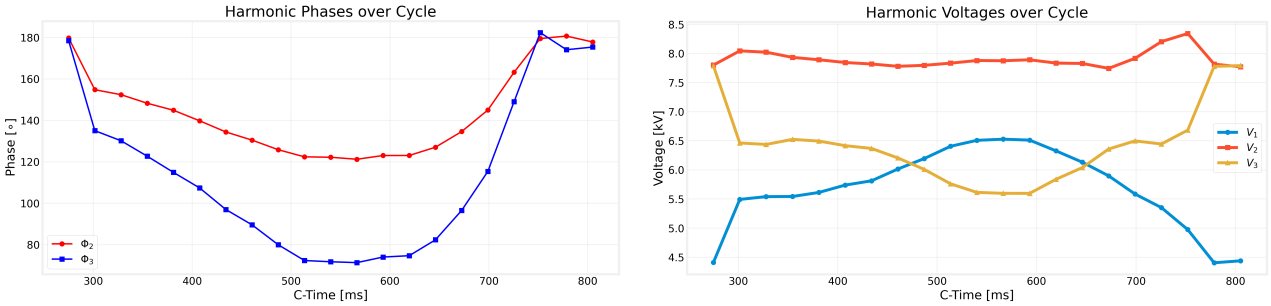


Figure 6.7: Initial and final waterfalls for the 1.4 GeV operational beam

Nevertheless, it can be seen as well in the Figure 6.7 that the agent is not able to achieve the fully optimal, symmetric and flat-topped profile shape in all the corrections. This can be identified in its majority as a higher density in the left and right lobes (mostly more in the right) throughout the waterfall plot. Furthermore, we can see at C-725 and C-750 that there is the highest discrepancy from the optimal profile shape (but it is not significant), and it coincides with the reduction to 0 T/s of the ramp rate. In general, this discrepancy could be for a couple of reasons: either the agent is not exploring the parameter space enough (which could be fixed by increasing the exploration σ of the parameters), or the KL divergence alone is not enough to drive the profile towards having matched and symmetric peaks (which could be fixed by adding additional reward shaping terms as mentioned in the outcome of Section 6.1.1). However, the space charge mitigation is still successful since the majority of the bunch density (except at the ends of the bucket of course) does not deviate by more than 30% of the maximum charge density at every point in the cycle.



(a) Phases Φ_2 and Φ_3 **(b)** Voltages V_1, V_2 and V_3

Figure 6.8: Initial and final phases Φ_2 and Φ_3 and voltages V_1, V_2 and V_3 for the 1.4 GeV operational beam

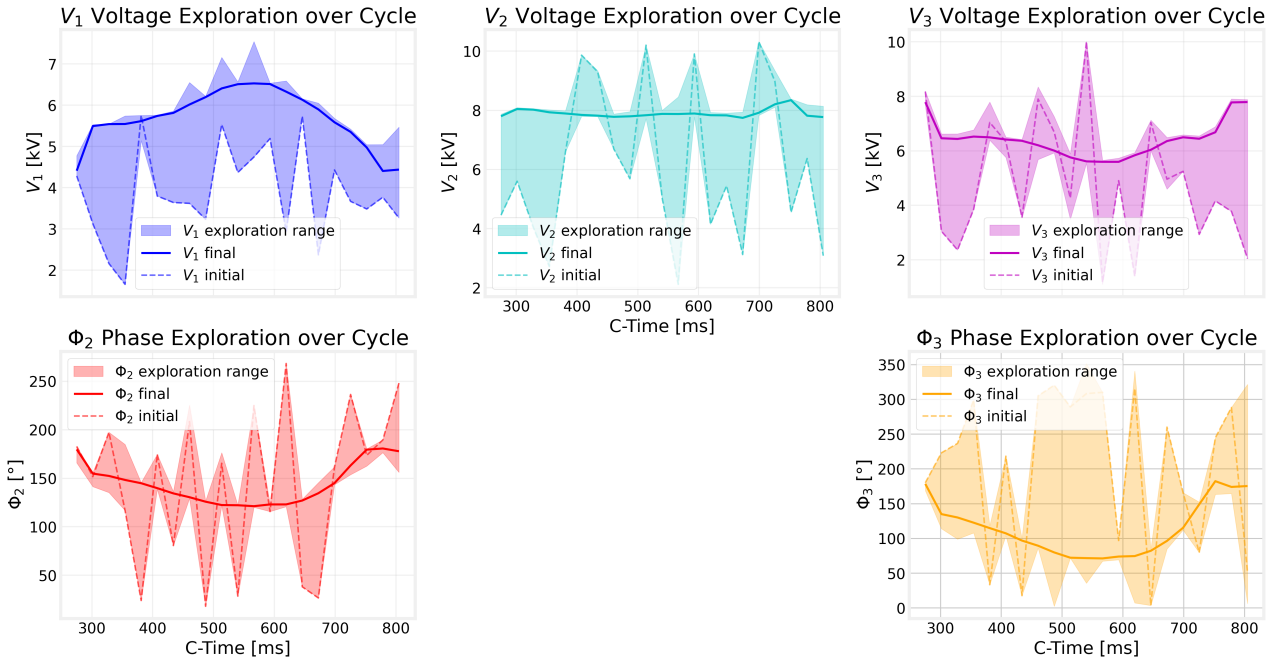


Figure 6.9: Parameters ranges explored by the agent for the 1.4 GeV operational beam

Meanwhile, for the 2.0 GeV beam the profile evolution displayed through the waterfall plots is shown in Figure 6.10. Again, the agent achieves the desired profile shape and the bunch lengthening is achieved throughout the cycle. The phases and voltages are shown in Figure 6.11, where again they progress smoothly with no jumps. The parameters ranges explored by the agent are shown in Figure 6.12 where again the agent is converging to the optimal parameters without going the direction opposite to where the optimal parameters lie. Convergence was reached within 20 corrections for the 2.0 GeV beam, but within 10 corrections almost the entirety of the cycle was converged, but 10 additional corrections were needed solely to cater to the portion of the cycle where the ramp rate is at its highest (C-625 to C-680).

A very interesting feature is that the agent does overcorrect by a larger extent for V_1 beyond the optimal correction when the bucket area is small due to the higher initial V_2 and V_3 and a high ramp rate. This is showing that the agent has learnt that higher voltages in counter-phase decrease the bucket area and that this effect is aggravated at a high $\dot{\mathcal{B}}(t)$, which is correct. This occurs for both the 1.4 and the 2.0 GeV beams,

indicating that the agent is likely overcorrecting at these points for V_1 since this is when the bucket area is at its lowest, so it is being conservative and increasing V_1, which is a desirable "safe operation" strategy which it learnt by itself (solely by requiring sufficient bucket area). Furthermore, this safety strategy is also enhanced by the fact that when V_2 and V_3 are seeded at values lower than the optimal values, the agent is able to achieve the optimal correction without overcorrecting at all practically as can be seen in the V_2 and V_3 plots in Figure 6.9 and Figure 6.12, as this would cause the bucket area to be reduced due to them primarily being in counter-phase with V_1 (or at least this is the attempt). This is one of the most beneficial features of reinforcement learning as it is learning safety strategies (in this particular problem) by itself without any explicit guidance other than the reward function.

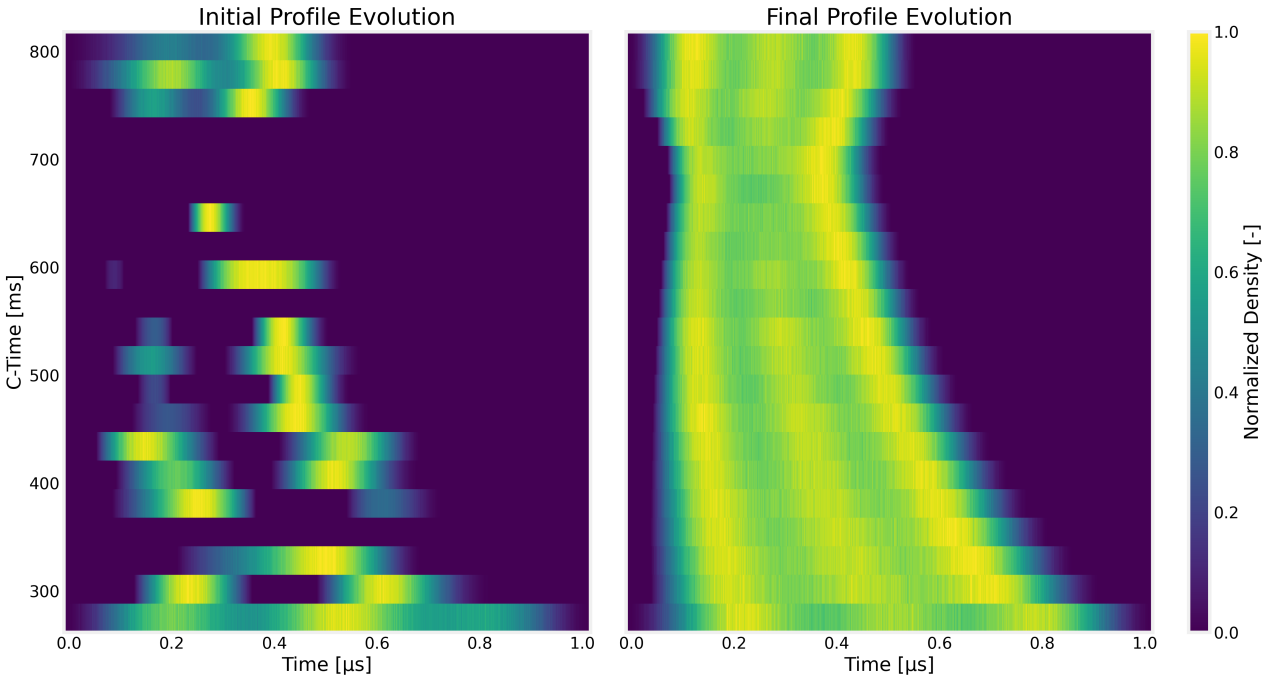


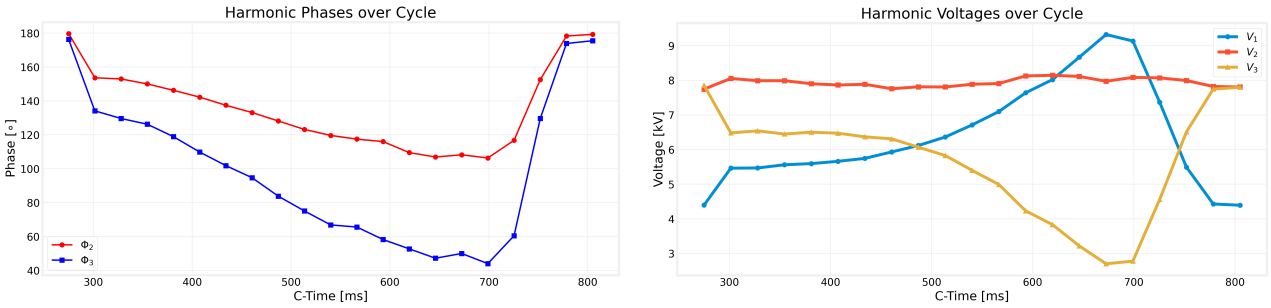
Figure 6.10: Initial and final waterfalls for the 2.0 GeV operational beam

However, it can be seen that the agent is again slightly struggling to make the profiles completely flat-topped at the optimal parameters. Primarily, for both cycles, it seems that V_2 is slightly too high to have the central peak be more prominent, despite the Φ_2 appearing to be optimal. Additionally, to consistently have the higher peak density on the right lobe of the profile, it would seem that Φ_3 (or the combination of both Φ_2 and Φ_3) should be slightly higher than the value that is being identified by the agent. This is likely due to the KL divergence measure and the way it is implemented as mentioned previously in the training results. Furthermore, retraining the agent with the modified KL divergence measure would not imply a significant redesign of the reward function, while likely improving the performance of the agent. If paired with a hyperparameter optimisation, the performance is sure to be improved.

Another behaviour that is consistent with not having the ideal profile shape is that whenever the $\dot{\mathcal{B}}(t)$ increases, the agent increases V_1 to compensate for the smaller bucket area, and also lowers V_3 to allow for a larger bucket area, but V_2 is kept practically constant for both cycles. This basically leads to the effect seen at the high $\dot{\mathcal{B}}(t)$ portion of the cycle where the waterfall plot loses its central peak, because the smaller V_3 while keeping V_2 just as high, will lead to the disappearance of the central loop in the inner separatrix while maintaining the two loops on the side. A more ideal behaviour would be to decrease both V_2 and V_3 (probably in different proportions, and assuming they are phased correctly in BLM) while increasing V_1 to whatever is required for sufficient bucket area. This could also be a feature related to the the reward combination of sufficient bucket area and the KL divergence method used, as there could be a local minima of the KL divergence with these types of profiles.

Operational Cycles Under Different Impedance Conditions

Doubling the PSB impedance significantly increases the induced voltages seen by the beam. Thus, it becomes important to analyse the robustness of the agent to these conditions. It was already seen that the agent learns to be robust to initialization conditions given by the fact that the initial guess is chosen at random in the range of the parameters and that this also happens in training. It can even progress when there is no profile, meaning no accelerating bucket, being formed by the matching process. Despite this shift, the agent remains robust and generalizes well across operating conditions. This behaviour is largely enabled by its recurrent memory in the same way as in the 1D case, as it conditions actions on recent profile evolution and the ramp-rate context



(a) Phases Φ_2 and Φ_3 **(b)** Voltages V_1, V_2 and V_3

Figure 6.11: Initial and final phases Φ_2 and Φ_3 and voltages V_1, V_2 and V_3 for the 2.0 GeV operational beam

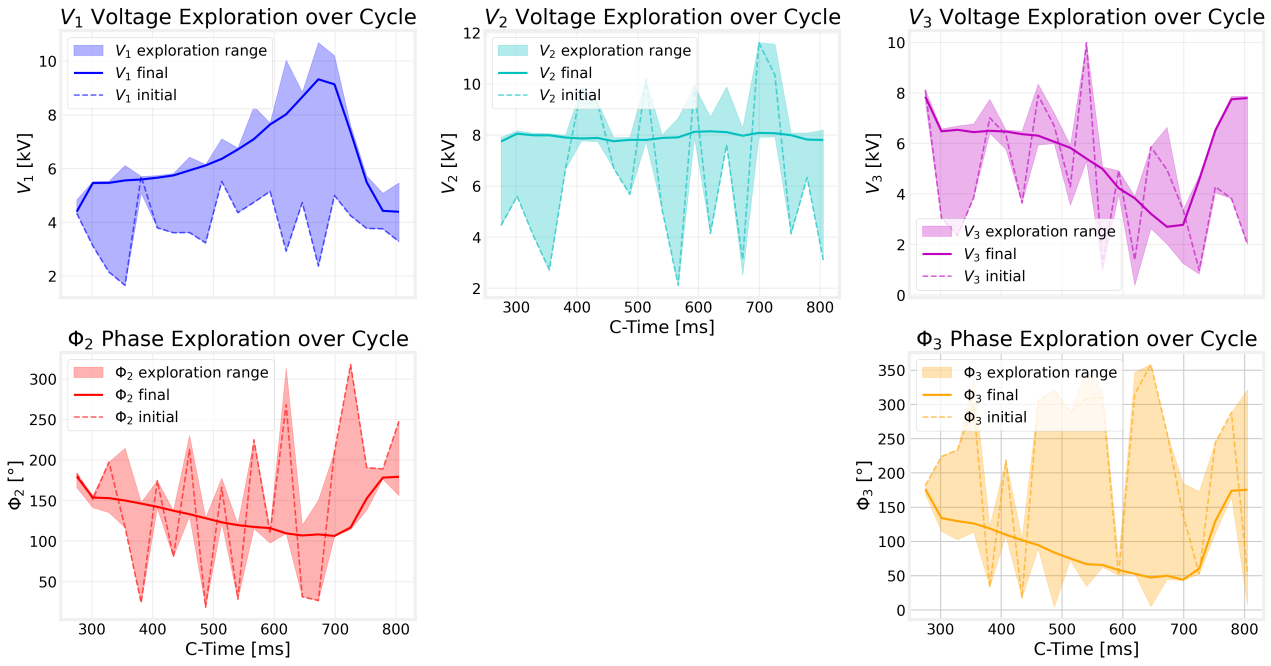


Figure 6.12: Parameters ranges explored by the agent for the 2.0 GeV operational beam

rather than on single-shot observations. As a result, the controller adapts coherently over the cycle instead of chasing instantaneous noise.

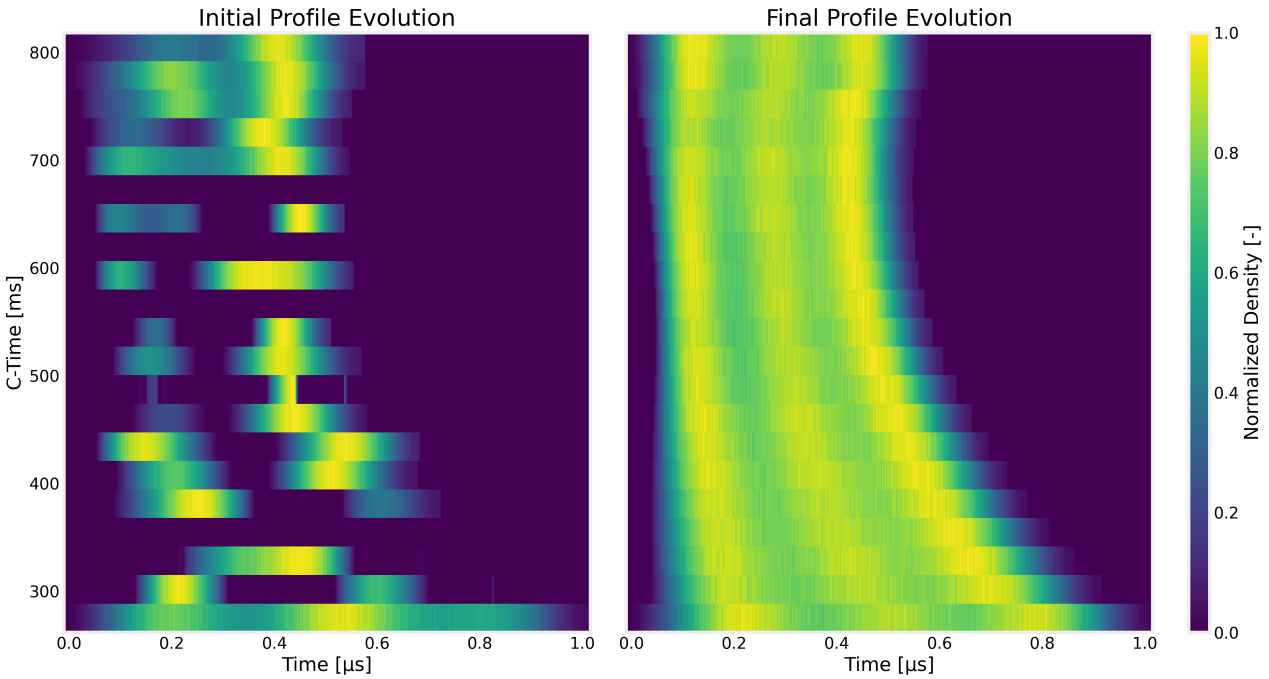
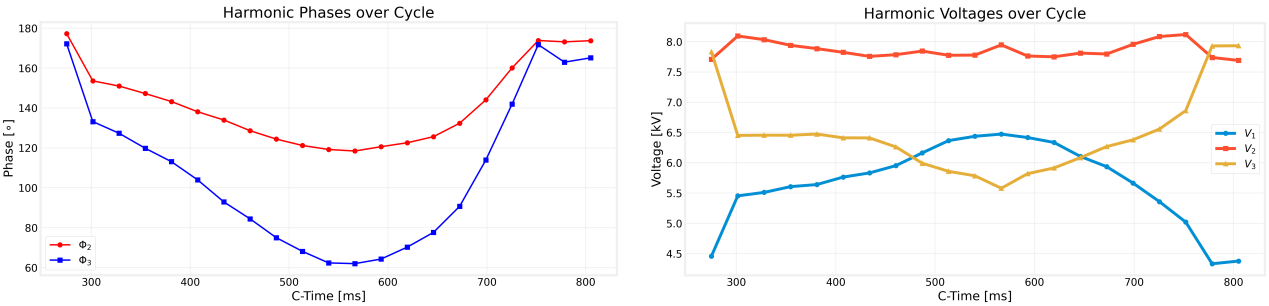


Figure 6.13: Initial and final waterfalls for the 1.4 GeV operational beam under double the PSB impedance (with servoloops)

For the 1.4 GeV beam, the dominant response is a phase adaptation for h_3 in order to compensate for the phase shift caused by the increased impedance rather than a change in bucket area. The waterfalls in Figure 6.13 remain close to the desired shape, and the phases and voltages in Figure 6.14 show a systematically lower Φ_3 across the cycle, with a minimum roughly 10° lower at the highest $\dot{\mathcal{B}}$. The voltage trajectories, including V_1, do not indicate a compensatory increase in bucket area other than a more jagged change in V_3 at the highest $\dot{\mathcal{B}}$. The agent instead resolves the impedance-induced distortion through a phase shift. This confirms that the learned policy leverages temporal context to make minimal, targeted adjustments.



(a) Phases Φ_2 and Φ_3

(b) Voltages V_1, V_2 and V_3

Figure 6.14: Initial and final phases Φ_2 and Φ_3 and voltages V_1, V_2 and V_3 for the 1.4 GeV operational beam under double the PSB impedance (with servoloops)

That said, the same mild profile asymmetries observed earlier persist here: the KL-divergence-based term still biases the solution toward slight left/right imbalances, visible in the waterfalls and consistent with the discussion in the operational-case results. This points to the KL computation as the primary source of residual asymmetry, rather than a limitation in the controller's ability to generalize under higher impedance. Nevertheless, the agent converges again within 10 corrections for the 1.4 GeV beam.

For the 2.0 GeV beam, the higher and differently structured $\dot{\mathcal{B}}(t)$ segments pose a tougher challenge. In the window from approximately C-600 to C-680, Figure 6.17 shows oscillations in V_2 and V_3 that are mirrored by fluctuations in Φ_2 and Φ_3. These modulations manifest as non-optimal profiles in Figure 6.16. While space-charge constraints are less stringent in this portion of the cycle, maintaining stable control is still desirable. The behavior suggests that, under large ramp-rate transients unique to the 2.0 GeV programme, the KL-driven objective and observation design can encourage small corrective oscillations. Even so, outside this narrow

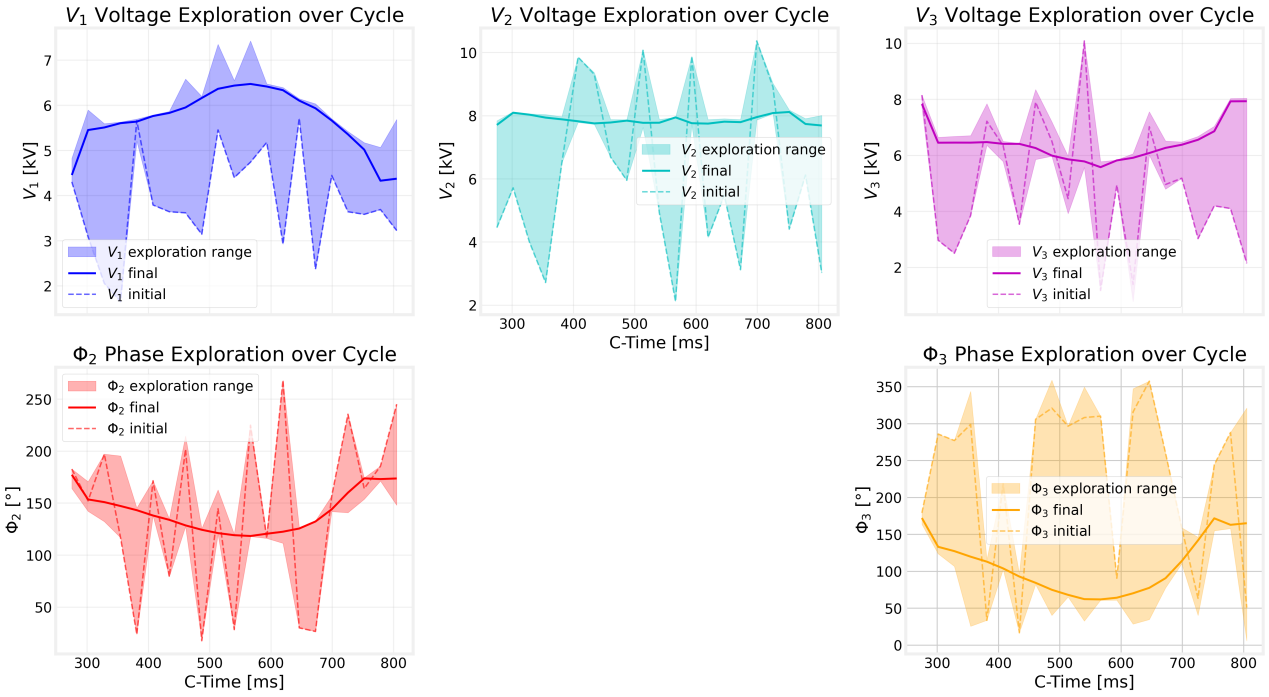


Figure 6.15: Parameters ranges explored by the agent for the 1.4 GeV operational beam under double the PSB impedance (with servoloops)

window the controller remains stable and effective, indicating that the recurrent policy continues to generalise across the doubled-impedance regime.

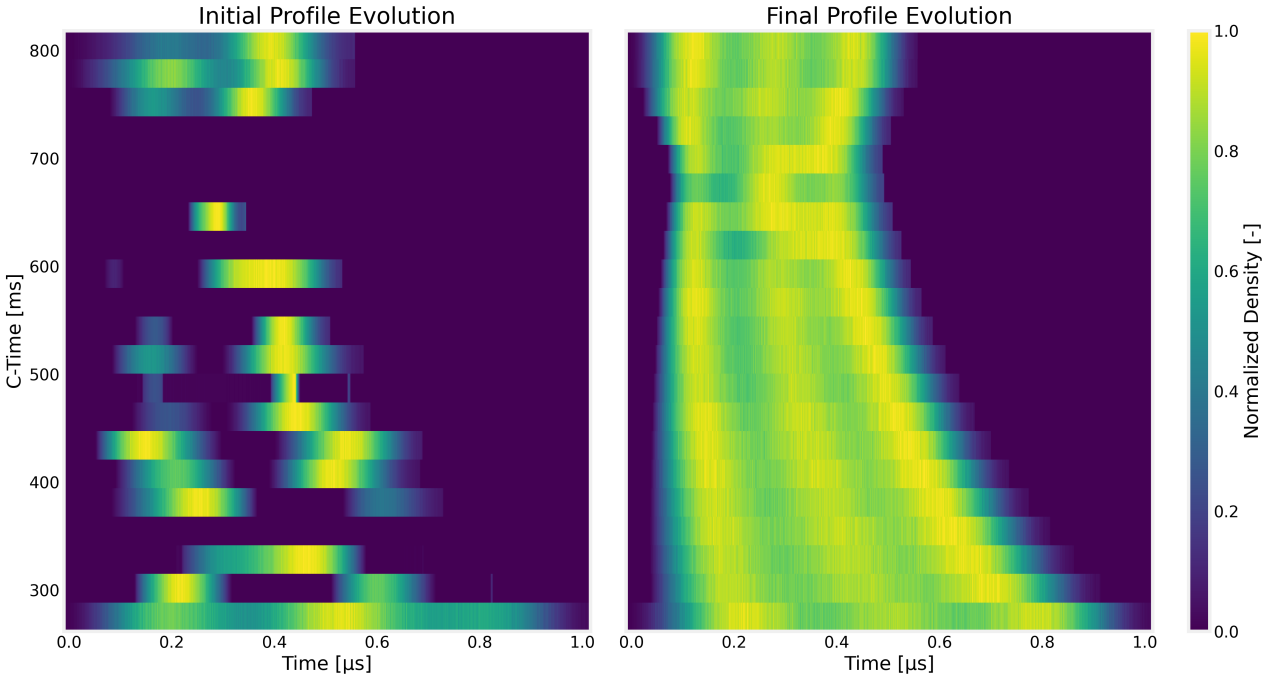
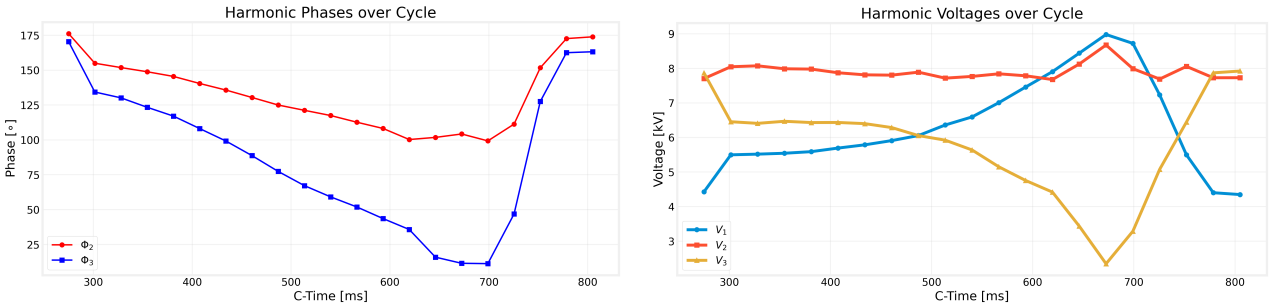


Figure 6.16: Initial and final waterfalls for the 2.0 GeV operational beam under double the PSB impedance (with servoloops)

Exotic Cycles

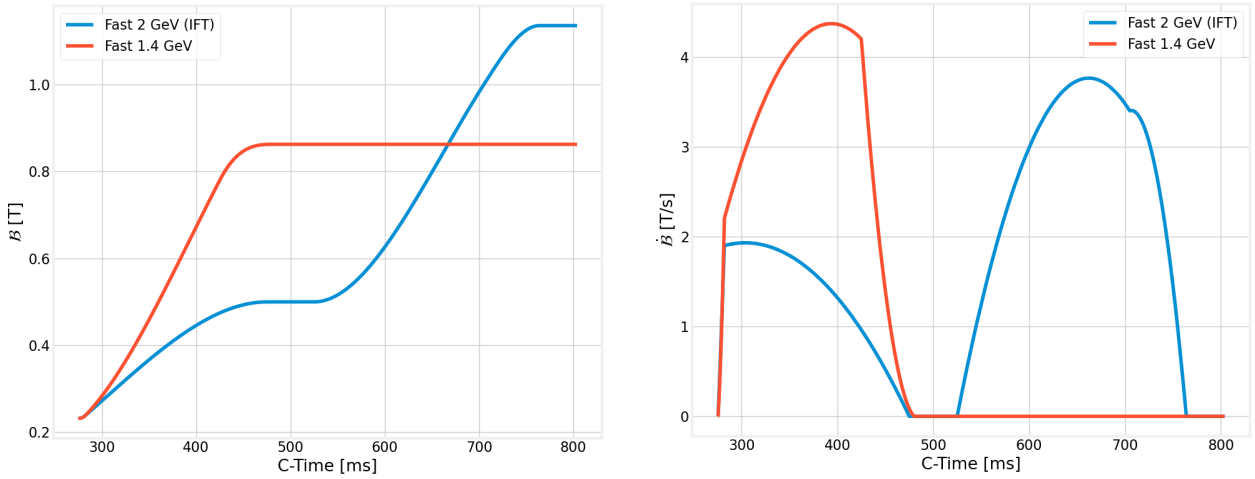
Finally we move on to the last test, the exotic cycles. They are trying to minimize the amount of time that the beam spends at low energies, while simultaneously satisfying the requirements on the RMS current that the dipole magnets can handle. As such the 1.4 GeV Fast Cycle given its very high ramp rate, requires a cycles with low ramp rates before and after it to be used in the supercycle to let the magnets cool down and to avoid saturating the magnets. Meanwhile, for the 2.0 GeV Intermediate Flat Top cycle, the ramp rate is lower than the peak 4.37 T/s encountered for the 1.4 GeV Fast Cycle, but still requires an intermediate flat top to avoid



(a) Phases Φ_2 and Φ_3 **(b)** Voltages V_1, V_2 and V_3

Figure 6.17: Initial and final phases Φ_2 and Φ_3 and voltages V_1, V_2 and V_3 for the 2.0 GeV operational beam under double the PSB impedance (with servoloops)

saturating the magnets, which is why the ramp rate temporarily drops to 0 T/s to allow the magnets to cool down. The magnetic field programmes and ramp rates for the two cycles are shown in Figure 6.18.



(a) Magnetic field $\mathcal{B}(t)$ **(b)** Magnetic field ramp rate $\dot{\mathcal{B}}(t)$

Figure 6.18: Magnetic field programmes $\mathcal{B}(t)$ and ramp rate $\dot{\mathcal{B}}(t)$ for the 1.4 GeV Fast and 2.0 GeV Intermediate Flat Top cycles

It can be seen in Figure 6.19 that the agent has failed to properly set the optimal parameters for triple-harmonic bunch lengthening when in the presence of a very large acceleration rate. This is mainly because of the range of $\dot{\mathcal{B}}(t)$ that is not seen during training and the agent cannot extrapolate to such high ramp rates. Besides the failure of the agent in achieving the triple-peaked bunch profile, or a fully flattened double-harmonic profile at that, an instructive safety mechanism emerges at the very high $\dot{\mathcal{B}}(t)$ of the 1.4 GeV Fast Cycle. When the agent cannot extrapolate to such high ramp rates, it drives $V_3 \to 0$ already anticipating a smaller bucket area over the affected interval, prioritizing sufficient V_1 to maintain the accelerating bucket area rather than risking instability. It is also using all the V_{total} at its disposal, but refuses to reduce V_2 to increase bucket area further, which not what we want given that the system is also badly phased in double-harmonic operation now, exhibiting peaked profiles. Perhaps by training at higher acceleration rates it would be possible to learn more optimal behaviour here, but that is only speculation. Regardless, after the acceleration, it is able to maintain the constant triple peaked structure that is desired, but at this point there is no acceleration, so there is no real difficulty.

This behavior is visible in the waterfalls of Figure 6.19 and the trajectories in Figure 6.20. As also indicated by the exploration plot in Figure 6.21, V_1 overshoots in the highest-$\dot{\mathcal{B}}(t)$ region and Φ_2 does not follow the correct nominal trend. Attempts to normalize the ramp rate by the maximum gradient (4.37 T/s) did not yield convergence at these points, consistent with the fact that such ramp rates were out of distribution during training. This is acceptable in practice since the 1.4 GeV Fast Cycle is a research case characterized by a very small bucket area rather than an operational cycle, and as mentioned before it requires cycles before and after it to compensate for the large ramp rates to let the magnets cool down[3].

Meanwhile, for the 2.0 GeV Fast Cycle with an intermediate flat top, the agent achieves the desired bunch

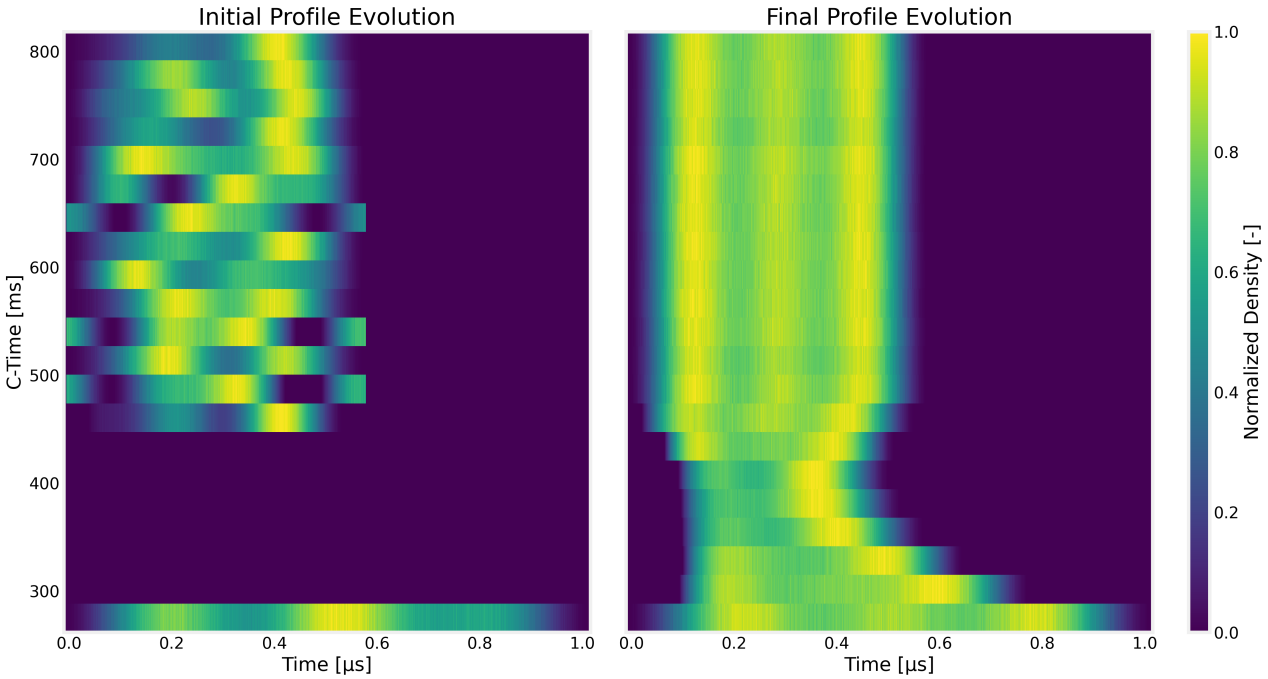
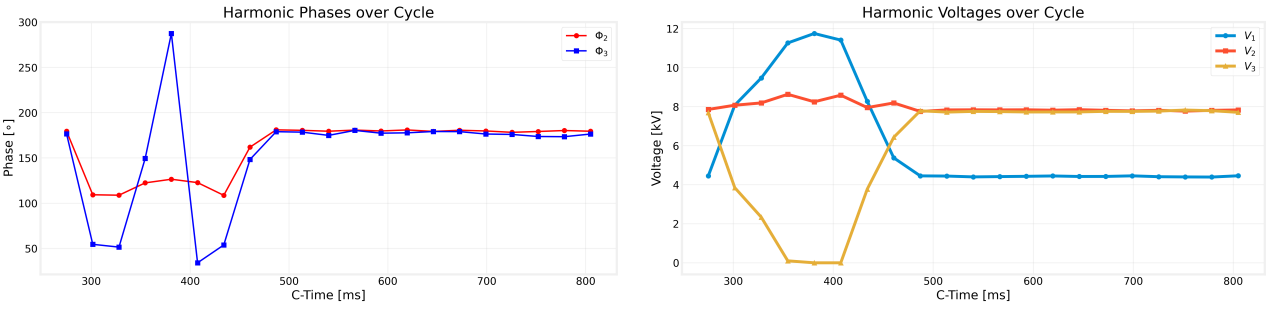


Figure 6.19: Initial and final waterfalls for the 1.4 GeV Fast Cycle



(a) Phases Φ_2 and Φ_3

(b) Voltages V_1, V_2 and V_3

Figure 6.20: Initial and final phases Φ_2 and Φ_3 and voltages V_1, V_2 and V_3 for the 1.4 GeV Fast Cycle

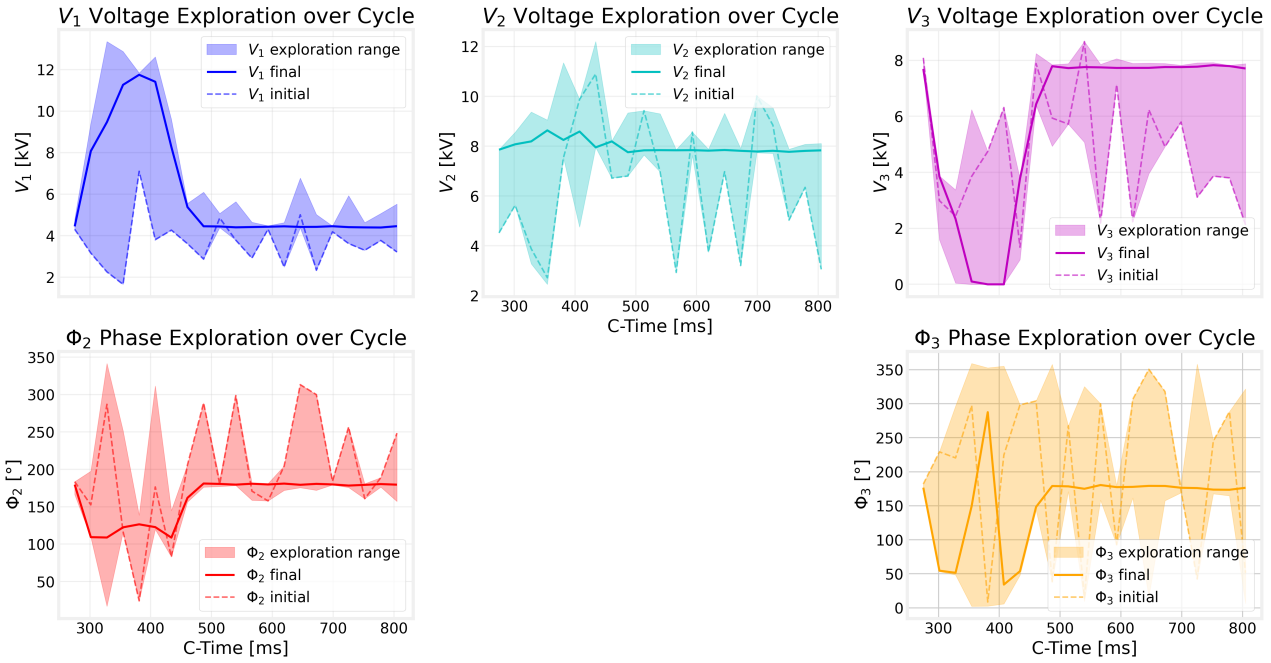


Figure 6.21: Parameters ranges explored by the agent for the 1.4 GeV Fast Cycle

shape across the cycle because the peak $\dot{\mathcal{B}}(t)$ remains within the envelope seen during training. The waterfalls in Figure 6.22 show good agreement with the target profiles; the phase and voltage trajectories in Figure 6.23 are smooth and consistent with the flat-top structure; and the exploration plot in Figure 6.24 indicates efficient progression toward the operating region without unnecessary excursions, with the same overshoot of V_1 seen previously in all cycles where V_2 or V_3 were initialized at a high value.

Aside from this, we can see the effect of the intermittent flat top on the bunch profiles, as the waterfall plot shows that the bunch length grows again at this point, consistent with the lack of acceleration during this plateau. Following this, the agent is able to maintain the desired bunch shape across most of the cycle, with the phases and voltages following the correct nominal trend. It still can be seen how at the second acceleration portion of the cycle, the bunch density in the middle of the bunch fades away due to the decrease in V_3.

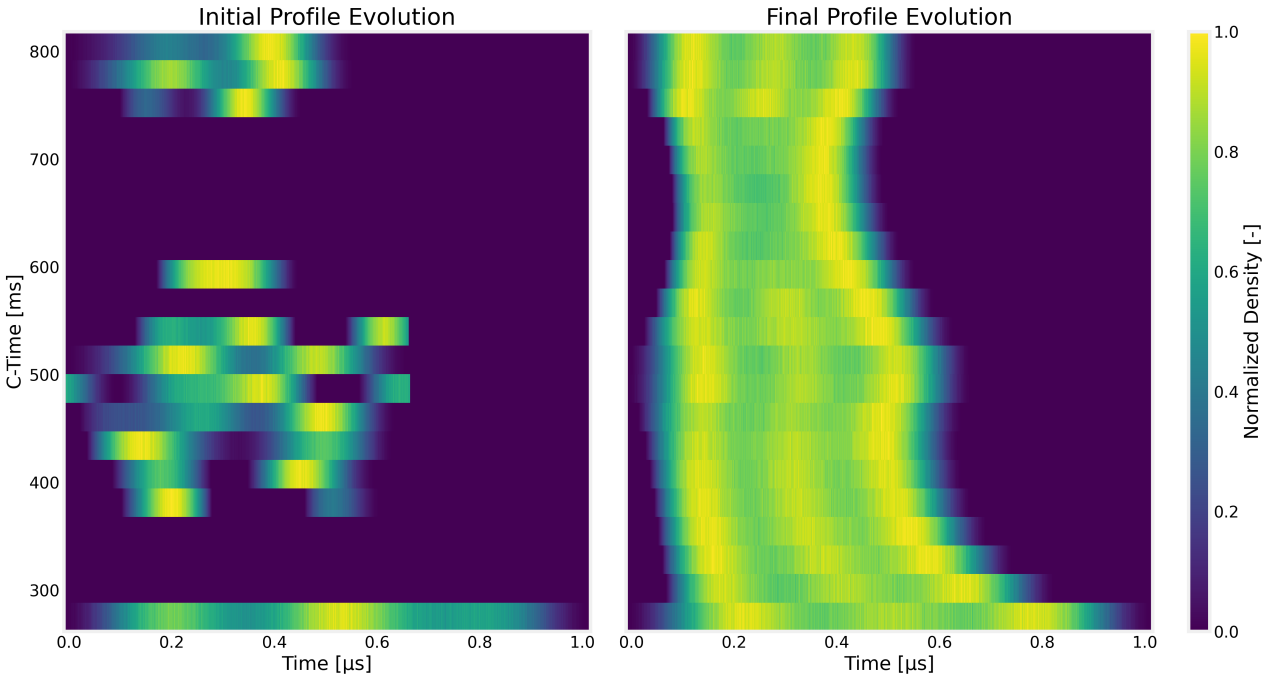
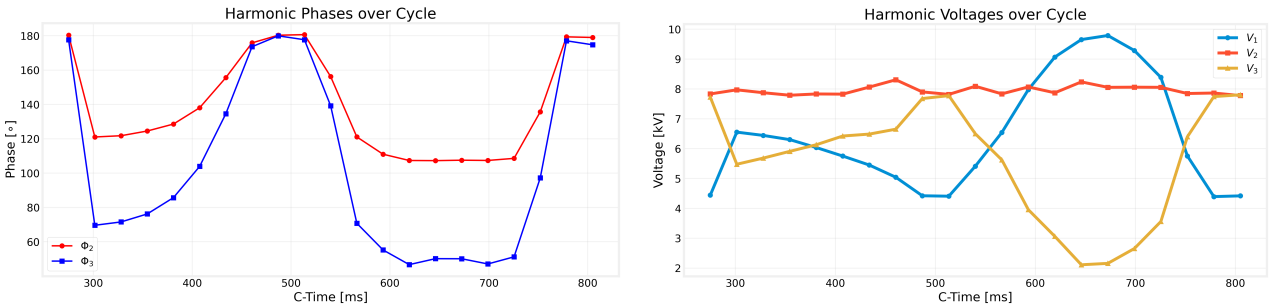


Figure 6.22: Initial and final waterfalls for the 2.0 GeV Fast Cycle with intermediate flat top



(a) Phases Φ_2 and Φ_3 **(b)** Voltages V_1, V_2 and V_3

Figure 6.23: Initial and final phases Φ_2 and Φ_3 and voltages V_1, V_2 and V_3 for the 2.0 GeV Fast Cycle with intermediate flat top

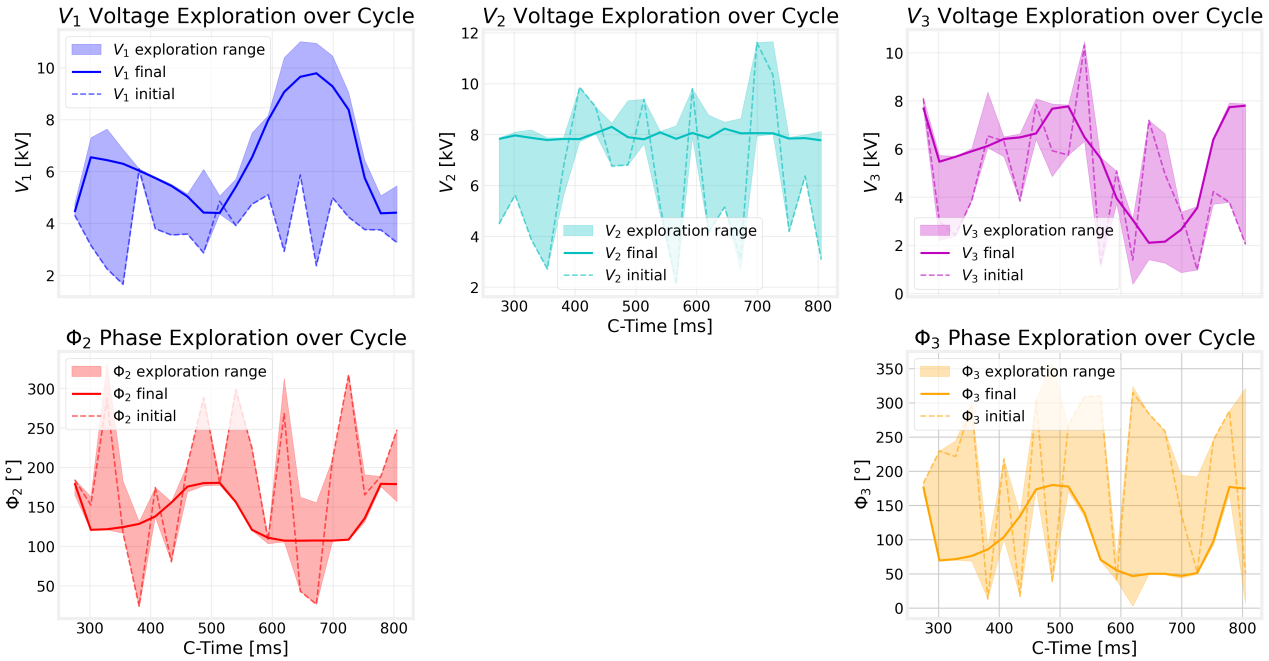


Figure 6.24: Parameters ranges explored by the agent for the 2.0 GeV Fast Cycle with intermediate flat top

Overall, these results show that the agent can cater to magnetic-field programmes with substantially more structure, given that the $\dot{\mathcal{B}}(t)$ is contained within the training regimen of the model. The recurrent memory enables it to contextualise corrections over time and handle flat tops and rapid ramps, with only the most extreme, out-of-distribution ramp rates causing performance to degrade into conservative, safety-preserving behaviour, but with high charge concentrations in some cases. As a result of this analysis, it would be beneficial to perform a hyperparameter optimisation to improve the performance of the agent under all conditions, but before, to also modify the KL divergence method to better account for the peak equalisation without requiring the standard deviation method.

6.1.3. PSB Implementation Roadmap

The transition from simulation-based training to operational deployment in the PSB requires a structured approach that bridges the gap between the controlled training environment and the complex realities of accelerator operation. Given that there was not enough time to test the triple-harmonic optimiser in the PSB, this roadmap outlines the practical steps and possibilities for implementing both the triple-harmonic corrector and the online training capabilities that tie into it, drawing on the developed infrastructure and lessons learned during validation for the 1D case.

Triple-Harmonic Corrector Implementation

The corrector implementation represents the most immediate deployment pathway, designed for operational reliability and operator confidence. The core architecture leverages the established LSTM-TD3 framework with adaptations for real-time PSB integration through the PyDA control system interface, as was the case for the double-harmonic Φ_2 corrector.

The preliminary triple-harmonic corrector operates through a similar multi-stage pipeline that begins with the computation point selection. Here it was already identified that there is potential for improvement given that the code does not correctly capture the peaks in the approximate synchronous phase. Nevertheless, given that now we have full parameter control over the longitudinal dynamics of the beam in the PSB, we can set the Δt to our liking and perform the optimisation with arbitrary precision. In practice this would likely cause issues due to how much the solution could vary over a short timescale, inducing oscillations and unwanted effects, so this should be limited to a reasonable number.

Furthermore, the profile acquisition follows the same procedure as listed in Section 5.1.6, and we use the same `MultiHistoryBuffer` class manages this complexity, tracking observations and actions over the maximum history length, which in this case is 10 previous states and actions. During each correction iteration, the system performs batch inference across all computation points, generating normalised parameter adjustments that are then converted back to physical voltage and phase settings. A critical component is the KL-divergence tracking system that monitors profile quality at each computation point throughout the correction process. Rather than simply applying the final parameter set and finishing the optimisation, the corrector maintains a record of the best-performing parameters for each C-time based on achieved KL values. After 20 correction iterations, it applies this optimised parameter combination, effectively performing a form of online optimisation that adapts to the specific conditions of each cycle.

The parameter conversion system handles the translation between the normalised action space used by the neural networks and the physical machine settings. Two mirrored functions apply transformations to convert the internal representation to the voltage amplitudes and phase settings required by the RF systems and likewise for introducing the current voltages and phases into the state. This architecture,together with the bounded action space, ensures that corrections remain within safe operational bounds while providing sufficient flexibility for optimisation.

Safety mechanisms are embedded throughout the corrector implementation. Parameter bounds are enforced at multiple levels, from the action clipping in the neural networks to final validation before machine setting. The system can revert to synchronous-phase estimates if optimisation fails, and operators retain full manual override capability through the standard PSB control interfaces.

Online Training Framework

Meanwhile, it is also possible to perform online training and it represents the ultimate evolution of the system, enabling continuous learning and adaptation directly from operational data. This approach transforms the PSB itself into a distributed training environment where each C-time becomes an independent learning context, in the same way that previously we parallelized environments to speed up training. The only difference is that now the model is pretrained and ensured to work at least to a certain extent before being put to train online.

The training architecture mirrors the parallel environment structure used in simulation, but with real beam profiles replacing the synthetic data. Each ”environment” corresponds to a specific time window in the magnetic cycle, allowing the agent to build up its understanding for different acceleration phases. The temporal resolution can be adjusted from the standard 20 ms intervals down to 5 ms or even 2 ms for fine-grained control, limited primarily by the acquisition system capabilities and how much variability there is in the actions taken by the agent and whether it is safe.

The online learning loop follows a structured episode-based approach. Each episode begins with parameter initialization based on the current machine state, followed by a sequence of profile acquisitions, neural network inferences, and parameter updates. The system waits for beam response between corrections, building up temporal context through the LSTM memory mechanisms. Episode termination can be triggered by convergence criteria, maximum correction limits, or operator intervention, but in this case there is no desire to "converge " in a terminal sense. Instead, a certain amount of steps is used so the agent always has an optimisation horizon in mind. Furthermore, to cater to the online learning approach, the Prioritized Experience Replay buffer would now be implemented in a way that it automatically accumulates transitions from all C-times and magnetic field programmes (when they are used). Unlike simulation training where environments can be reset arbitrarily, online training must respect the operational schedule and beam availability. The system therefore can save time by saving the experience buffers between episodes and updating the actor and critic networks with the latest experiences, enabling learning to continue across different beam types and operational conditions.

The reward structure adapts the simulation-based approach to real operational constraints. Bucket area calculations use real-time magnetic field measurements and RF settings (but still using the Solfege module to do so quickly), while profile quality metrics are computed from actual beam measurements. The system can incorporate operational preferences, such as prioritizing stability over perfect flattening, or adapting to different beam intensities and emittances by modifying the reward functions used. Additionally, a key advantage of online training is the ability to learn from rare operational conditions that are difficult to simulate accurately. Injection transients, noisy acquisitions, and unusual instabilities all become part of the training data, potentially improving the system's robustness beyond what is achievable through simulation alone. The continuous learning also enables adaptation to slow drifts in machine behaviour (if there are any). Ideally one would have an implementation that includes a mechanism for safe exploration during initial online learning such as low intensities at first and requiring extensive monitoring to ensure beam stability until it could be trusted. Naturally this would only be done after the corrector is proven to work, at least partially.

In conclusion, both the corrector and online training implementations benefit from the verification work performed in simulation and it should be repeated again once the model is optimised even further through reward function tweaks or hyperparameter optimisation. The parameter normalization, safety bounds, and convergence criteria developed during the simulation phase transfer directly to the operational environment.

7

Conclusions & Recommendations

This chapter synthesises the outcomes of the work and highlights their implications for operational longitudinal control in the PSB. It first assesses how the research questions and project goals were addressed and how the stated requirements were satisfied. It then outlines practical recommendations for future development and concludes by highlighting the main contributions and the outlook for deploying and generalising the methodology.

7.1. Evaluation of the Project

This section evaluates how well the research questions were answered, whether the project goals were achieved, and the extent to which the requirements in Chapter 3 were met by the CBAM and LSTM-TD3 agents.

7.1.1. Research Questions

Question R.Q.1 and Question R.Q.2 were addressed by developing the CAE, the CBAM model, and the LSTM-TD3 agent, and by benchmarking them against manually phased programmes. The automated methods consistently provided strong approximations to the optimal phase in the PSB. Whereas in some cases they exceeded manual performance, in others they were slightly inferior, but still within acceptable tolerances and seeding the system in the correct neighbourhood of the optimal Φ_2(BLM) phase when not providing optimal profile. The performance in the CBAM model was higher than that of the unoptimised LSTM-TD3 agent and demonstrated certain robustness with the decaying corrections method in the PSB. For triple-harmonic operation in particular, the methods identified effective phases and amplitudes for all control parameters, albeit slightly deviated from the optimal ones, but this might be due to the KL divergence measure and the way it is implemented. The profile analysis in Section 2.4 and the validation studies confirmed that the performance targets defined in the requirements can be achieved relative to manual baselines. In double-harmonic optimisation, better performance was shown to be very likely for the optimised LSTM-TD3 agent by comparing the simulated behaviours of the unoptimised agent with its performance in the PSB, yet this still remains to be proven true.

Question R.Q.3 concerned identifying the maximal latency which was determined to be less than 30 seconds in Chapter 3 to achieve sub-super-cycle latency to be able to be efficient with at least 1 slot inside of the super-cycle. In the introductions of Section 5.1.7 and Section 5.2.6 it was mentioned that preprocessing, not inference, dominates runtime in the form of B-spline smoothing and re-interpolation to an RF period, taking roughly 1 second per profile. However, model inference is sub-second, and end-to-end acquisition, preprocessing, inference, and correction still remain below one super-cycle ($< 30\,\mathrm{s}$). Further latency reductions are feasible by adopting more efficient, tensorized smoothing using kernels and re-interpolation techniques. This would allow using 2 slots in the cycle to optimize, and reduce the time taken by a factor of 2.

Question R.Q.4 and Question R.Q.6 examined performance breadth and convergence under representative operating conditions. Validation spanned multiple cycles covering the operational range. While full cycle optimisation and safety constraints were not tested at full operational intensities (especially for the unoptimised LSTM-TD3 agent), comparisons to manual programmes showed that the automated methods perform well and sometimes improve upon manual settings currently used in the PSB. Notably, the 5D agent rapidly converges toward effective phases and amplitudes in simulation, which is a process that would take considerably longer using manual optimisation.

Question R.Q.5 focused on generalisation. Combining verification and validation, the optimised LSTM-TD3 agent extrapolated across impedances and initialisation conditions and across injection transients which are practically random. As for the initialisation conditions and impedances, the CBAM model was particularly sensitive to being seeded in BSM as this caused the false belief of being in BLM at low emittances. This was also the case for the unoptimised LSTM-TD3 agent. However, the CBAM model handled injection transients more reliably in the tests performed than the unoptimised LSTM-TD3 agent, which over-corrected in such cases much like it did in simulation, highlighting the value of the optimised and threshold-using variant which showcases a high degree of robustness and generalisation capabilities.

Safety-related Question R.Q.7 through to Question R.Q.9 were investigated through verification for all models and, in part, through validation incidents with the unoptimised LSTM-TD3 agent. Robustness varied: the optimised double- and triple-harmonic LSTM-TD3 agents were robust to impedance and initialisation changes,

whereas the unoptimised LSTM-TD3 agent and the CBAM model were less so. Safety mechanisms were exercised during a failed TOF-cycle optimisation near C-700 led by the unoptimised LSTM-TD3 agent, where sanitisation reverted to the synchronous phase estimate $\phi_{s_{approx}}$ to avoid beam loss and longitudinal oscillations at extraction. Additionally, random, noisy acquisitions from the digital cards were identified as a risk needing fixes and tests to be performed on the cards themselves, but also highlight the need for thresholding to remove noisy profile components, though a full testing was not possible due to time constraints. The triple-harmonic LSTM-TD3 agent also exhibited conservative, self-taught behaviour, increasing bucket area when higher-harmonic voltages risked reducing it and when high $\dot{\mathcal{B}}(t)$ were encountered, it would set $V_3 = 0$ to avoid not having enough V_1 to accelerate the bunch and achieve the desired bucket area.

Question R.Q.10 was addressed by specifying how parameters are applied in the PSB via device control and the Python API (Section 3.1) and by documenting the data acquisition and preprocessing pipeline (Section 5.1.6). This was fundamental to be able to implement any type of model in the PSB for profile-based parameter control of any type. Meanwhile, Question R.Q.11 was explored by using normalised parameters to generalise to other accelerators and beam types. The approach is promising and partially validated in simulation; scripts for double-harmonic bunch lengthening in LEIR were prepared but could not be tested due to a mixture of incompatible acquisition hardware and time constraints.

Furthermore, Question R.Q.12 was answered by a validation methodology discussed in Section 3.3.1 as a consequence of the requirement analysis performed that selects representative beam types spanning the operational range of the PSB and compares model performance against manually phased Φ_2 programs. Also, Question R.Q.13 was demonstrated by proving that the LSTM-TD3 methodology extends to triple-harmonic operation allowing for memory-augmented control of RF parameters. This approach was initially validated through Bayesian optimization (Section 2.4) and then reframed as an RL optimization problem. Through this method, it proved that the agent can learn meaningful and safe behaviour that meets performance objectives while simultaneously introducing self-taught safety practices not available with traditional methods.

Finally, Question R.Q.14 remains an open question: cross-accelerator testing was not possible within the time and hardware constraints at LEIR, nor was the optimised LSTM-TD3 model developed to be able to do so at the time that LEIR stopped its run. Nonetheless, the methodology scales across impedances and initial conditions. Scripts for LEIR are ready and can be exercised once acquisition cards are upgraded.

7.1.2. Project Goals

Having analysed the research questions, now it is important to assess which goals were completed in this project as a result of the investigation and testing performed. The development of the optimised LSTM-TD3 agent achieved Goal P.G.1 through Goal P.G.4 by improving upon CBAM with more accurate phasing (Φ_2, Φ_3) and voltage settings (V_1, V_2, V_3), as evidenced in Section 5.1.7 and Section 5.2.6. These goals are, however, only partially realized operationally because the optimised agent could not be tested on the real machine within the available time, despite showing the best simulated performance, convergence under varying impedances and initial conditions, and learned safety behaviour for triple-harmonic operation.

The verification and validation-related goals enabled progress on Goal P.G.5, Goal P.G.6 and Goal P.G.7. Verification across impedances and initial conditions supported Goal P.G.5 and Goal P.G.7, while validation advanced Goal P.G.6. As with earlier goals, the absence of on-machine testing for the final optimised solution and the triple-harmonic optimizer limits these to partial completion. Furthermore, safety mechanisms were validated during the failed TOF cycle case with the unoptimised LSTM-TD3 agent, where sanitization prevented beam loss and oscillations at extraction, contributing to Goal P.G.8 and Goal P.G.10. The CBAM model showed some robustness to noisy acquisitions by returning toward optimal settings after erroneous corrections, but its decayed-correction scheme restricted full recovery if noise recurred or struck mid-run due to corrections not being large enough. Thresholding and bounded corrections in the LSTM-TD3 agent are expected to mitigate this, though not yet tested. Consequently, Goal P.G.7, Goal P.G.8, and Goal P.G.10 are partially achieved.

Goal P.G.9 was met by integrating device control via the Python API and embedding the models in operational scripts. These scripts produce the monitoring plots used throughout Section 5.1.7 and Section 5.2.6, improving visibility and interpretability, and enabling Goal P.G.11. A preliminary triple-harmonic control script was also prepared, but without PSB tests, this counts as partial progress for the 5D case. Training and integration procedures for CBAM, LSTM-TD3, and the 5D agent were documented (Chapter 4) and with the code released in public repositories, fulfilling Goal P.G.12 [46, 44, 45, 48].

Goal P.G.13 could not be tested due to time and LEIR hardware constraints, though the methodology was shown to scale across impedances and initial conditions, so high hopes were had for the LEIR implementation. Finally, Goal P.G.14 was achieved in design by relying on normalized parameters to facilitate generalization across accelerators and RF system capabilities, with performance validated in simulation but not yet across machines. Profile evaluation for double- and triple-harmonic operation in Section 2.4 completed Goal P.G.15 for the 5D problem, and benchmarking against manual programs was provided for the double-harmonic phasing task.

In summary, most goals were achieved fully or partially. The principal exception is Goal P.G.13, which awaits hardware availability at LEIR. Evidence indicates the methodology scales across impedances and initial conditions and is ready for broader validation.

7.1.3. Requirement Completion

The requirements completion is assessed in the traceability matrix in Table 7.1, where the requirements are matched to whether or not their verification criteria outlined in Table 3.2 were met. Additionally, each verification criteria is discussed below with a discussion on why it was achieved, partially achieved or not achieved. This helps guide future developments in the automatization process.

Table 7.1: Verification Criteria Completion: ✓= met, × = not met, ≙ partially met

Ver. ID	Verification Description	Status (model)
	Functional Requirements Verification Completion	
VER-F-001	Demonstration of BLM achievement for all supported beam types within ±3 degrees of optimal phase, or at least visual comparison with existing operational phase setting to compare performance.	≙(CBAM)
VER-F-002	Verification of being able to perform only profile-based optimisation without requiring specific machine parameters.	✓(CBAM & LSTM-TD3 agent)
VER-F-003	Confirmation of triple-harmonic peak matching within 5% tolerance for all three peaks.	✓(LSTM-TD3 agent)
VER-F-004	Characterisation of the performance of the achieved solution with validation beam types and intensities across the full operational range.	≙(CBAM & LSTM-TD3 agent)
VER-F-005	Validation of fallback mechanisms when convergence is not achieved, ensuring system defaults to flattest achieved profile.	× (LSTM-TD3 agent)
	Performance Requirements Verification	
VER-P-001	Confirming super-cycle response time with full optimisation cycle completed within 30 seconds.	✓(CBAM & LSTM-TD3 agent)
VER-P-002	Convergence testing demonstrating BLM achievement within 30 cycles for double-harmonic optimisation.	≙(CBAM)
VER-P-003	Convergence testing demonstrating profile flattening within 50 cycles for triple-harmonic optimisation.	× (LSTM-TD3 agent)
VER-P-004	Long-term stability verification over 30 cycles after initial convergence (this necessitates convergence to be achieved first).	× (LSTM-TD3 agent)
	Safety Requirements Verification	
VER-S-001	Beam loss monitoring confirming losses remain below 20×10^{10} ppb limit during optimisation.	≙(CBAM)
VER-S-002	Testing of automatic reversion to synchronous phase for double-harmonic optimisation and returning to safe settings for triple-harmonic optimisation.	≙(CBAM & LSTM-TD3 agent)
VER-S-003	Validation of parameter bounds enforcement when corrections indicate exceeding predefined operational limits.	✓(CBAM & LSTM-TD3 agent)
VER-S-004	Verification of operator override capabilities with successful manual parameter control.	✓(CBAM & LSTM-TD3 agent)
VER-S-005	Reporting tool performance for all operational beams in simulation with impedance models included.	✓(CBAM & LSTM-TD3 agent)
	Integration Requirements Verification	
VER-I-001	End-to-end testing with setting parameters and checking successful parameter application through the Java front-end.	✓(N/A)
VER-I-002	Data flow validation from measurement systems to parameter settings with graphical demonstration of the complete data pipeline.	✓(N/A)
VER-I-003	Performance testing under realistic operational conditions including typical machine variations.	≙(CBAM & LSTM-TD3 agent)

Continued on next page

Table 7.1 – continued from previous page

Ver. ID	Verification Description	Status (model)
VER-I-004	Performance testing using noisier data coming from the Wall Current Monitor (WCM) to verify robustness.	↿(CBAM & LSTM-TD3 agent)
VER-I-005	Recovery testing demonstrating automatic recovery from transient faults within two super-cycles.	↿(CBAM & LSTM-TD3 agent)
	Operational Requirements Verification	
VER-O-001	User interface validation covering autonomy monitoring, diagnostics, logging, and manual override: UI displays convergence, parameters, and metrics; logs captured and exportable; operator takeover and hand-back succeed.	× (N/A)
VER-O-002	Documentation review checklist completed covering training framework, parameters, and PSB integration procedures.	✓(N/A)

1. **Functional Requirements Verification Completion**:

   - **VER-F-001: Partial.** Good approximation of the optimal phase for the PSB was achieved by the CBAM model, but the LSTM-TD3 agent was able to achieve better results in simulation, while its fully realized performance is yet to be tested. The CBAM model lacked complete convergence BLM phasing, but did improve upon the manually phased programs or achieved almost the same performance, but did lack perfect performance in splittings.

   - **VER-F-002: Met.** The models were able to perform only profile-based optimisation without requiring specific machine parameters and only used normalised parameters.

   - **VER-F-003: Partial**: The LSTM-TD3 agent was able to achieve good performance in simulation for the operational and exotic magnetic fields, but it was not able to be tested in the PSB due to time constraints. Additionally, the peak variations are a little higher than that what was indicated in ACC-002, but this could be remedied with a slight modification of the KL divergence methodology. As such, the criterion is only partially fulfilled.

   - **VER-F-004: Partial.** The models were able to achieve good performance in simulation for the validation beam types, but the performance was not entirely optimal in the PSB for the CBAM, but it was not able to be tested for the full range of intensities due to time constraints. Meanwhile, the LSTM-TD3 agent was able to achieve better performance than the CBAM model in simulation, but its fully realized performance is yet to be tested in the PSB for both the double and triple-harmonic optimisation.

   - **VER-F-005: Not Met.** This was not able to be performed as it is referring to looking through the buffer and then providing the parameters that yielded the lowest KL divergence. However, this was not possible due to lack of time.

   - **VER-F-006: Partial.** The models were able to perform simultaneous 5D optimisation with all of V_1, V_2, V_3, Φ_2, Φ_3 adjusted and correctly applied. However, this was not performed in the PSB due to time constraints, but it is known to work in simulation.

2. **Performance Requirements Verification**:

   - **VER-P-001: Met.** The models were able to achieve super-cycle response time with full optimisation cycle completed within 20 seconds for a time-step of 20 and 25 ms for both models in double-harmonic optimisation.

   - **VER-P-002: Partial.** This was partially fulfilled by the CBAM model in double-harmonic optimisation with the decaying solution method, but it lacked true convergence that is likely to be achieved by the optimised LSTM-TD3 agent.

   - **VER-P-003: Not Met.** Since the triple-harmonic optimisation was not performed in the PSB, this requirement was not met. However, simulation results show that it is able to converge within 20 cycles for several magnetic field ramps (but not the 1.4 GeV Fast cycle) and impedances.

   - **VER-P-004: Not Met.** This was not able to be tested with the optimised LSTM-TD3 agent for double- and triple-harmonic optimisation due to time constraints, but the unoptimised agent was unable to achieve consistent long-term stability.

3. **Safety Requirements Verification**:

   - **VER-S-001: Partial.** This criterion was met by the CBAM model when tested with an injection intensity of 80×10^{10} ppb, but it was not able to be tested with the optimised LSTM-TD3 agent for double and triple-harmonic optimisation due to time constraints. Furthermore, the unoptimised LSTM-TD3 agent was unable to achieve this criterion since it did lose 20×10^{10} ppb for the 2025 BCMS cycle.

   - **VER-S-002: Met.** The unoptimised LSTM-TD3 agent led to a solution for the TOF cycle with large phase changes over one Δt (20 ms), and the sanitization procedure was able to revert to the initial estimate of $\phi_{s_{approx}}$ to not cause beam loss and longitudinal bunch oscillations at extraction, this requirement was met for double-harmonic optimisation. However, it was not able to be tested with the optimised LSTM-TD3 agent for triple-harmonic.

   - **VER-S-003: Met.** The models and corrections were designed in such a way that they could never exceed the operational limits.

   - **VER-S-004: Met.** The operator of the tool can always simply interrupt the script and run the sanitization procedure to revert to the initial estimate of $\phi_{s_{approx}}$. Furthermore, there are already mechanisms in place to remove intensity from a cycle if it is causing too much beam loss. However,

   - **VER-S-005: Met.** The extensive verification practices for all models were performed in Section B.1 and B.2 to characterize the performance and know what to expect in the PSB. Furthermore, this also allowed the tentative extrapolation of the optimised LSTM-TD3 performance to the PSB by comparing the simulation results of the unoptimised agent with its performance in the PSB.

4. **Integration Requirements Verification**:

   - **VER-I-001: Met.** The tool was able to be integrated into the PSB through Python scripts making use of PyDA to set parameters. However, this was not performed for the in validation procedures for the triple-harmonic optimisation due to time constraints, but it is known to work.

   - **VER-I-002: Met.** The data flow was validated from the measurement systems, to the profile selection, reinterpolation and smoothing, and finally to the inference and parameter setting with graphical demonstration of the complete data pipeline.

   - **VER-I-003: Partial.** The models were able to achieve approximations of BLM phasing in the PSB for the CBAM model, but since the point of this criterion is to ensure reliability, it is only partially met since it was susceptible to extraordinary, noisy acquisitions from the digital cards. However, this is expected to be mitigated by the use of the thresholding mechanism in the optimised LSTM-TD3 agent.

   - **VER-I-004: Partial.** The models were able to partially recover from transient faults caused by the noisier acquisitions from the digital cards that produced erroneous corrections. However, the CBAM model was not able to completely recover due to the decayed corrections and the unoptimised LSTM-TD3 agent was able to "recover", but to the wrong phasing due to its lack of performance. This is expected to be fixed with the optimised LSTM-TD3 agent.

   - **VER-I-005: Partial.** Again, like the previous criterion, it is only partially met due to the same reasons.

5. **Operational Requirements Verification**:

   - **VER-O-001: Not Met.** This was not able to be developed into a UCAP node with a GUI due to time constraints. However, the scripts do provide a good interface for the operator to monitor the optimization process, to revert to the initial estimate of $\phi_{s_{approx}}$, and if manual override is needed, it can be done by simply interrupting the script, manually setting the parameters using the existing Java front-end, and then resuming the script from where it was interrupted.

   - **VER-O-002: Met.** The documentation was completed (partially through this work) covering the training framework, parameters, and PSB integration procedures [46, 44, 45, 48].

This completes the traceability matrix for the verification and validation criteria, where it can be seen that 9 of the 22 criteria were met, while 9 were partially met and awaiting final testing, and 4 were not met. The ones that were not met relate mostly to the lack of time to perform the final testing for the 5D, triple-harmonic optimizer, while the remaining one discusses more the development of a GUI to aid visualization and interaction with the models rather than a script. As a result, there are still tasks to be performed until the agent can become operational, but the simulation results are promising.

7.2. Recommendations

This section outlines prioritized directions for future work across three areas: the CAE, the data acquisition and preprocessing pipeline, and the LSTM-TD3 optimization. CBAM-specific improvements are not emphasized because it already provides a good approximation to the optimal phase in the PSB and is surpassed in simulation by the optimised LSTM-TD3 agent, whose full operational performance remains to be tested. This is also not discussed since the CBAM model is inherently trying to make a regression network fit all different parameter combinations of $V_{total}, r_2, \dot{\mathcal{B}}, FF$ and σ by looking solely at their profile and minimizing its loss based on a single correction, which is not realistic in practice. In the presence of the aforementioned limitations, it is better to continue with the development of better LSTM-TD3 agents rather than trying to improve the CBAM model.

7.2.1. Future Directions for CAE Development

The current limitations stem more from excessive compression than from architectural constraints. Increasing the latent dimensionality (for example to 128, 256, or 512) is likely to preserve more information and improve reconstruction fidelity, at the cost of less compression. Hierarchical or progressive variants that learn multiple compression scales could better capture both global shape and local features, while physics-informed regularization terms that encode beam-dynamics constraints may steer reconstructions toward physically plausible profiles. This would make the CAE kind of an enhanced downsampling black-box to aid training. Meanwhile, adding attention mechanisms could preserve long-range correlations that standard convolutional layers under-represent. Until reconstruction quality reaches operational standards, the CAE remains most valuable academic analysis tool to see that profiles can be mapped to correction.

7.2.2. Data acquisition and preprocessing

Selecting better computation points is a critical point to enable better solutions for all the optimization methodologies and the double and triple-harmonic problems. This could potentially be made more principled by moving from approximate synchronous-phase heuristics to an energy-gain or magnetic field ramp rate $\dot{\mathcal{B}}$ criterion, which would also allow generalization to the triple-harmonic case. A smoothed estimate of energy gain per turn (which would basically be a scaled $\dot{\mathcal{B}}$) would allow defining points based on any peaks or discontinuities located in this curve. However, this could cause missing points in double-harmonic operation as in the case of ISOLDE, as the voltages have to be considered to understand how this estimate changes. So maybe the solution is to use the same methodology, but simply smooth the approximate synchronous phase in a more robust manner or not use the approximate synchronous phase, but a more sophisticated estimate.

Regardless, this is not possible for triple-harmonic operation, so a compromise must be made to select points based solely on the energy gain. This strategy is expected to be more robust, but might lack performance in double-harmonic around splittings where computation points improve performance. In parallel, a fast separatrix estimator can replace approximate phase in both training and initialization. Estimating the synchronous phase under double-harmonic operation (and using the central stable fixed point for triple-harmonic) would yield better initial conditions and more reliable early corrections.

7.2.3. LSTM-TD3 Optimization

For the double-harmonic, optimised LSTM-TD3 agent, operational testing is the most immediate priority. Running the optimised LSTM-TD3 agent on the real system will validate the simulation-based analysis made in Section 5.2.6 and extrapolated to the optimised and thresholded model, while also allowing to establish the performance relative to CBAM. Additionally, it is important to perform initial tests with the 5D model to surface any architectural adjustments needed to account for behaviors not captured in simulation. Ideally this is done before any type of hyperparameter optimization is done, which certainly should be done to improve performance further while also implementing a different and better performant KL divergence measure.

Subsequently, narrowing parameter ranges and removing low-impact hyperparameters can concentrate the optimization on the most influential factors and extract the remaining performance margin for the double-harmonic agent, as was done for the CBAM model. Also, a sensitivity study (e.g., ANOVA with Taguchi designs) would quantify hyperparameter effects and provide statistical confidence in configuration choices, but in any case we can always perform tests to verify and validate the training outcomes.

Finally, it is fundamental that triple-harmonic optimization undergoes a dedicated hyperparameter optimization (either 1 or 2) rather than inheriting settings from the double-harmonic case. This is mainly because if the objectives and important features differ, tailored hyperparameters are expected to improve performance, eliminate the slight profile tilt observed during verification, and increase reliability for broader deployment. It is also very important that once a trained and validated model is developed, that it is put to train in the PSB including potential online training such that the RL model adapts to the real system and increases PSB performance. This is only possible due to the objective value that can be extracted directly from the profile.

7.3. Conclusions

In conclusion, this work introduced and optimised an open-source reinforcement learning framework tailored to longitudinal beam control. In particular, the integration and acceleration of prioritized experience replay for

LSTM-TD3, the use of Huber loss to stabilize training under Q-network overestimation, and multi-environment parallelization substantially improved data throughput and learning stability. This makes it a very strong RL framework for generalized problem solving that would be enhanced by memory. The fact that it has been improved from the original open source code in terms of computational efficiency, capabilities, learning speed and stability make it a valuable contribution to the ML community, and also to the ML for particle accelerators.

A generalized, profile-centric framework was developed for double-harmonic BLM phasing that relies solely on beam-profile geometry and general RF parameters rather than machine-specific settings. The approach met the stated performance targets and generalized across impedances, initialization conditions, and beam types in simulation. The steps remaining are the pending validation in the PSB, for both the double-harmonic and triple-harmonic cases, while the same methodology is positioned for transfer to other accelerators such as LEIR.

For triple-harmonic space-charge mitigation, we established a training framework in which the agent learned safety strategies from the reward structure alone and extrapolated this to more complex behaviours. This is in contrast to the double-harmonic optimizer and is only possible due to the higher complexity and dimensionality and control of the action space and its effect on beam profiles. Furthermore, the normalized RF parameters in the state provided essential context, enabling the agent to capture the interdependencies between parameters in a quicker manner and profile geometry without binding the solution to a specific accelerator.

Finally, we implemented a profile acquisition and correction pipeline for the PSB that aligns center-profile acquisitions throughout the cycle. This infrastructure not only underpins the deployed controllers but also enables future model development and diagnostic analyses. Together, all these contributions provide both enablers and methodologies for autonomous longitudinal control in accelerators, with clear paths to operational deployment and cross-accelerator generalization.

References

[1] Takuya Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: (2019). URL: https://github.com/pfnet/optuna/.

[2] S Albright et al. "New Longitudinal Beam Production Methods in the CERN Proton Synchrotron Booster". In: *12th International Particle Accelerator Conference* (2021). DOI: 10.18429/JACoW-IPAC2021-THPAB183.

[3] Simon Albright and Asvesta Foteini. "Magnetic cycle optimisation in the CERN PS booster ". In: *16th International Particle Accelerator Conference* (2025). ISSN: 2673-5490. DOI: 10.18429/JACoW-IPAC25-TUPS026. URL: https://meow.elettra.eu/81/pdf/TUPS026.pdf.

[4] Simon Albright et al. *BLonD: Beam Longitudinal Dynamics Code*. 2025. URL: https://gitlab.cern.ch/blond/BLonD.

[5] Simon Albright et al. "Comissioning the New LLRF System of the CERN PS Booster". In: *13th International Particle Accelerator Conference* (2022), pp. 2060–2063. ISSN: 2673-5490.

[6] Simon Albright et al. "Two-Dimensional Longitudinal Painting at Injection into the CERN PS Booster". In: *JACoW* HB2023 (2024), pp. 563–566. DOI: 10.18429/JACOW-HB2023-THBP38.

[7] Samuel Alvernaz and Julian Togelius. "Autoencoder-augmented Neuroevolution for Visual Doom Playing". In: *2017 IEEE Conference on Computational Intelligence and Games, CIG 2017* (July 2017), pp. 1–8. DOI: 10.1109/CIG.2017.8080408. URL: https://arxiv.org/abs/1707.03902v1.

[8] F Antoniou et al. "The PS Booster Alignment Campaign and a New Tune Control Implementation After the LHC Injectors Upgrade at CERN". In: (2021). DOI: 10.18429/JACoW-HB2021-MOP14. URL: https://edms.cern.ch/document/1505902/..

[9] F Asvesta and H Bartosik. "Resonance Driving Terms From Space Charge Potential". In: *CERN Internal Note* (2019).

[10] Foteini Asvesta et al. "Pushing High Intensity and High Brightness Limits in the CERN PSB after the LIU Upgrades". In: *68th Advanced Beam Dynamics Workshop for High-Intensity High-Brightness Hadron Beams* 2023 (2024), pp. 458–461. DOI: 10.18429/JACOW-HB2023-THBP09. URL: https://cds.cern.ch/record/2900904.

[11] Adrien Bardes, Jean Ponce, and Yann LeCun. "VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning". In: *ICLR 2022 - 10th International Conference on Learning Representations* (May 2021). URL: https://arxiv.org/abs/2105.04906v3.

[12] D Barrientos et al. " A New Beam Loading Compensation and Blowup Control System Using Multi-Harmonic Digital Feedback Loops in the CERN Proton Synchrotron Booster". In: *13th International Particle Accelerator Conference* (2022), pp. 907–910. DOI: 10.18429/JACoW-IPAC2022-TUPOST024. URL: https://arxiv.org/abs/1910.06643.

[13] S. Bettoni et al. "Temporal profile measurements of relativistic electron bunch based on wakefield generation". In: *Physical Review Accelerators and Beams* 19.2 (Feb. 2016), p. 021304. ISSN: 24699888. DOI: 10.1103/PHYSREVACCELBEAMS.19.021304/FIGURES/3/MEDIUM. URL: https://journals.aps.org/prab/abstract/10.1103/PhysRevAccelBeams.19.021304.

[14] Ka Heng Chan, Aida Mustapha, and Mohammed Ahmed Jubair. "Comparative Analysis of Loss Functions in TD3 for Autonomous Parking". In: *Journal of Soft Computing and Data Mining* 5.1 (June 2024), pp. 1–14. ISSN: 2716621X. DOI: 10.30880/jscdm.2024.05.01.001.

[15] Alexander Wu. Chao et al. *Handbook of Accelerator Physics and Engineering*. 3rd ed. World Scientific, Feb. 2023. ISBN: 9789811269196.

[16] M Cieslak-Kowalska et al. "Evolution of High Intensity Beams in the CERN PS Booster After H- Injection and Phase Space Painting". In: *7th International Particle Accelerator Conference* (2016).

[17] P Elson et al. "Introducing Python as a Supported Language for Accelerator Controls at CERN". In: *ICALEPCS*. JACOW Publishing, Geneva, Switzerland, Mar. 2021, pp. 236–241. ISBN: 978-3-95450-221-9. DOI: 10.18429/JACOW-ICALEPCS2021-MOPV040. URL: https://en.wikipedia.org/wiki/.

[18] Phil Elson. "PyDA: The future of device access with Python Background: Python in ATS". In: *IPP MD Days*. Feb. 2025. URL: https://indico.cern.ch/event/1488714/contributions/6283874/attachments/3007457/5302626/IPP%20MD%20Days%202025_%20PyDA.pdf.

[19] M Filippova et al. *Proceedings of the CERN–Accelerator–School course: Introduction to Particle Accelerators*. 2nd Edition. 2022. URL: https://indico.cern.ch/event/1356988/.

[20] V. Forte, E. Benedetto, and M. McAteer. "CERN Proton Synchrotron booster space charge simulations with a realistic model for alignment and field errors". In: *Physical Review Accelerators and Beams* 19.12 (Dec. 2016), p. 124202. ISSN: 24699888. DOI: 10.1103/PHYSREVACCELBEAMS.19.124202/FIGURES/27/MEDIUM. URL: https://journals.aps.org/prab/abstract/10.1103/PhysRevAccelBeams.19.124202.

[21] Peter I. Frazier. "A Tutorial on Bayesian Optimization". In: *arXiv* (July 2018). URL: https://arxiv.org/abs/1807.02811v1.

[22] Scott Fujimoto, Herke Van Hoof, and David Meger. "Addressing Function Approximation Error in Actor-Critic Methods". In: *35th International Conference on Machine Learning, ICML 2018* 4 (Feb. 2018), pp. 2587–2601. URL: https://arxiv.org/abs/1802.09477v3.

[23] Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. "An Efficient Approach for Assessing Hyperparameter Importance". In: *31st International Conference on Machine Learning*. 2014.

[24] Leandro Intelisano. "Loss of Landau Damping in Double Harmonic RF Systems". In: (May 2023).

[25] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *arXiv* (2015).

[26] Abbas Jafar and Myungho Lee. "Comparative Performance Evaluation of State-of-the-Art Hyperparameter Optimization Frameworks". In: *Transactions of the Korean Institute of Electrical Engineers* 72.5 (May 2023), pp. 607–619. ISSN: 22874364. DOI: 10.5370/KIEE.2023.72.5.607.

[27] M. Janga Reddy and D. Nagesh Kumar. "Evolutionary algorithms, swarm intelligence methods, and their applications in water resources engineering: A state-of-the-art review". In: *H2Open Journal* 3.1 (Jan. 2020), pp. 135–188. ISSN: 26166518. DOI: 10.2166/H2OJ.2020.128/699810/H2OJ2020128.PDF.

[28] Michal Krupa and Marek Gasior. "A New Wall Current Transformer for Accurate Beam Intensity Measurements in the Large Hadron Collider". In: *Energies 2023, Vol. 16, Page 7442* 16.21 (Nov. 2023), p. 7442. ISSN: 1996-1073. DOI: 10.3390/EN16217442. URL: https://www.mdpi.com/1996-1073/16/21/7442/htm%20https://www.mdpi.com/1996-1073/16/21/7442.

[29] Alexandre Lasheen and Simon Albright. *SOLFEGE*. Geneva, 2025. URL: https://gitlab.cern.ch/rf-br/solfege.

[30] Shyh-yuan Lee. *Accelerator Physics*. 4th. World Scientific Publishing Company, 2018. URL: www.worldscientific.com.

[31] Timothy P. Lillicrap et al. "Continuous control with deep reinforcement learning". In: *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings* (Sept. 2015). URL: https://arxiv.org/abs/1509.02971v6.

[32] Yunhan Lin et al. "Efficient TD3 based path planning of mobile robot in dynamic environments using prioritized experience replay and LSTM". In: *Scientific Reports 2025 15:1* 15.1 (May 2025), pp. 1–16. ISSN: 2045-2322. DOI: 10.1038/s41598-025-02244-z. URL: https://www.nature.com/articles/s41598-025-02244-z.

[33] Xiaomin Liu et al. "Value Distribution DDPG With Dual-Prioritized Experience Replay for Coordinated Control of Coal-Fired Power Generation Systems". In: *IEEE Transactions on Industrial Informatics* 20.6 (June 2024), pp. 8181–8194. ISSN: 19410050. DOI: 10.1109/TII.2024.3369712.

[34] Cedric Lombard et al. "Improved Antiproton Production Beam at CERN". In: *14th International Particle Accelerator Conference* (2023). Ed. by JACoW Publishing. ISSN: 2673-5490. DOI: 10.18429/JACoW-IPAC2023-TUPM075.

[35] E Meier et al. "Artificial Intelligence Systems for Electron Beam Parameters Optimisation at the Australian Synchrotron LINAC ". In: *1st International Particle Accelerator Conference*. 2010, pp. 1770–1772.

[36] Lingheng Meng, Rob Gorbet, and Dana Kuli´c Kuli´c. "Memory-based Deep Reinforcement Learning for POMDPs". In: *IEEE/RSJ International Conference on Intelligent Robots and System* (2021). URL: https://github.com/benelot/pybullet-gym.

[37] Steven Mortier. "Exploring top-quark and Z boson coupling in top pair production in association with a Z boson". In: (July 2020). DOI: 10.13140/RG.2.2.14191.76962. URL: https://www.researchgate.net/publication/355586020_Exploring_top-quark_and_Z_boson_coupling_in_top_pair_production_in_association_with_a_Z_boson.

[38] n_TOF collaboration. "The neutron Time-Of-Flight facility, n TOF, at CERN (I): Technical Description the n TOF collaboration a". In: *CERN Document Server* (2015). URL: https://cds.cern.ch/record/1514680/files/n_TOF-PUB-2013-001.pdf.

[39] Matthias Nabinger et al. "The International Muon Collider Collaboration". In: *12th International Particle Accelerator Conference* (2021), pp. 3792–3795. ISSN: 2673-5490. URL: http://www.desy.de/.

[40] M Neroni et al. "Recent Updates in the Impedance Characterization of the CERN PS Booster Finemet RF System". In: *15th International Particle Accelerator Conference* (2024), pp. 3120–3123. DOI: 10.18429/JACoW-IPAC2024-THPC52.

[41] Keiron O'Shea and Ryan Nash. "An Introduction to Convolutional Neural Networks". In: *International Journal for Research in Applied Science and Engineering Technology* 10.12 (Nov. 2015), pp. 943–947. DOI: 10.22214/ijraset.2022.47789. URL: https://arxiv.org/abs/1511.08458v2.

[42] H Okita et al. "Consideration of Triple-Harmonic Operation for the J-PARC RCS". In: *12th International Particle Accelerator Conference*. 2021. ISBN: 9783954502141. DOI: 10.18429/JACoW-IPAC2021-WEPAB177. URL: https://blond.web.cern.ch.

[43] Qing Pan et al. "An interpretable 1D convolutional neural network for detecting patient-ventilator asynchrony in mechanical ventilation". In: *Computer Methods and Programs in Biomedicine* 204 (June 2021), p. 106057. ISSN: 0169-2607. DOI: 10.1016/J.CMPB.2021.106057.

[44] Anibal Luciano Pastinante. *Double Harmonic Dataset Generator and CNN Trainer*. 2025. URL: https://gitlab.cern.ch/apastina/double-harmonic-dataset-generator-and-cnn-trainer.

[45] Anibal Luciano Pastinante. *Double Harmonic LSTM-TD3 RL Agent*. 2025. URL: https://gitlab.cern.ch/apastina/double-harmonic-lstm-td3-rl-agent.

[46] Anibal Luciano Pastinante. *Double Harmonic PSB Corrector-CNN and LSTM-RL*. 2025. URL: https://gitlab.cern.ch/apastina/double-harmonic-psb-corrector-cnn-and-lstm-rl.

[47] Anibal Luciano Pastinante. *Multi-Harmonic Particle Tracking*. 2025. URL: https://github.com/apastinante/multi_harmonic_tracking/tree/main.

[48] Anibal Luciano Pastinante. *Triple Harmonic LSTM-TD3 RL Agent Trainer*. 2025. URL: https://gitlab.cern.ch/apastina/triple-harmonic-lstm-td3-rl-agent-trainer.

[49] Ken Peach, P. Wilson, and B. Jones. "Accelerator science in medical physics". In: *British Journal of Radiology* 84.SPEC. ISSUE 1 (Dec. 2011). ISSN: 00071285. DOI: 10.1259/BJR/16022594. URL: https://www.researchgate.net/publication/221870520_Accelerator_science_in_medical_physics.

[50] Danilo Quartullo. "Simulations of RF beam manipulations including intensity effects for CERN PSB and SPS upgrades". PhD thesis. Feb. 2019. URL: https://cds.cern.ch/record/2658075/files/CERN-THESIS-2019-006.pdf.

[51] Antonin Raffin. *Getting SAC to Work on a Massive Parallel Simulator: An RL Journey With Off-Policy Algorithms (Part I)*. 2025. URL: https://araffin.github.io/post/sac-massive-sim/.

[52] Antonin Raffin et al. "Stable-Baselines3: Reliable Reinforcement Learning Implementations". In: *Journal of Machine Learning Research* 22 (2021), pp. 1–8. URL: https://github.com/DLR-RM/stable-baselines3..

[53] Aravind Rajeswaran et al. "Towards Generalization and Simplicity in Continuous Control". In: (). URL: https://sites.google.com/view/simple-pol.

[54] Ryan Roussel et al. "Bayesian Optimization Algorithms for Accelerator Physics". In: *Physical Review Accelerators and Beams* 27.8 (Dec. 2023). ISSN: 24699888. DOI: 10.1103/PhysRevAccelBeams.27.084801. URL: https://arxiv.org/abs/2312.05667v2.

[55] Jeyabharathy Sadaiyandi et al. "Stratified Sampling-Based Deep Learning Approach to Increase Prediction Accuracy of Unbalanced Dataset". In: *Electronics (Switzerland)* 12.21 (Nov. 2023). ISSN: 20799292. DOI: 10.3390/ELECTRONICS12214423.

[56] Ramprasaath R. Selvaraju et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization". In: *International Journal of Computer Vision* 128.2 (Oct. 2016), pp. 336–359. DOI: 10.1007/s11263-019-01228-7. URL: http://arxiv.org/abs/1610.02391%20http://dx.doi.org/10.1007/s11263-019-01228-7.

[57] Ghada Sokar and Pablo Samuel Castro. "Mind the GAP! The Challenges of Scale in Pixel-based Deep Reinforcement Learning". In: *Google DeepMind* (May 2025). URL: http://arxiv.org/abs/2505.17749.

[58] Helga Timko et al. "Beam Longitudinal Dynamics Simulation Suite BLonD". In: (June 2022). URL: https://arxiv.org/abs/2206.08148.

[59] Laurens Van Der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605.

[60] Shuhei Watanabe. "Tree-Structured Parzen Estimator: A Tutorial Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance". In: *arXiv* (2023). URL: https://github.com/nabenabe0928/tpe/tree/single-opt..

A

Theory of Bayesian Optimization

Bayesian Optimization (BO) is a powerful strategy for the global optimization of expensive-to-evaluate black-box functions. It is particularly well-suited for problems where the objective function lacks an analytical form, is non-convex, and each evaluation is computationally costly, such as the tuning of the multi-component objective (or loss) function $\mathcal{L}(\mathbf{w})$ for the BLM phasing problem described in Section 2.4. The concept of BO revolves around building a probabilistic surrogate model of the objective function we are evaluating and to use this model to intelligently select the most promising points to evaluate next, thereby finding the optimum in as few iterations as possible.

A.1. Optimizing Weights for Loss Function

In the context of the problem defined for double harmonic operation in Section 2.4, it attempted to minimize the errors between the "true" Φ_2 phase deviations, $\Delta\Phi_2^{opt}$, normalized by π (such that the function was bounded between 0 and 1) and the losses approximated by $\mathcal{L}(\mathbf{w})$. The errors were broken into 2 parts: the deviation of the prediction of the minimum of $\mathcal{L}(\mathbf{w})$ and that of the true function $\Delta\Phi_2^{opt}$ ($\Delta_1 = |\min_{\Phi_2}(\mathcal{L}(\mathbf{w})) - \min_{\Phi_2}(\Delta\Phi_2^{opt})|$, and the correct prediction of the sign of the gradient at all points such that $\mathcal{L}(\mathbf{w})$ could leverage gradient information when close to the optimum ($\Delta_2 = |sign(\frac{d}{d\Phi_2}\mathcal{L}(\mathbf{w})) - sign(\frac{d}{d\Phi_2}\Delta\Phi_2^{opt})|$.

The algorithm proceeds iteratively, following two main steps [21]:

1. Surrogate Modeling with Gaussian Processes

The unknown objective function $\mathcal{L}(\mathbf{w})$, which maps the 6-dimensional weight vector $\mathbf{w}$ to a scalar loss value, is modeled by a Gaussian Process (GP). A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution. It is completely specified by its mean function $m(\mathbf{w})$ and covariance (kernel) function $k(\mathbf{w}, \mathbf{w}')$:

$$\mathcal{L}(\mathbf{w}) \sim \mathcal{GP}(m(\mathbf{w}), k(\mathbf{w}, \mathbf{w}')) \tag{A.1}$$

Given a set of t observations (the "training data") $\mathcal{D}_{1:t} = \{\mathbf{w}_i, \mathcal{L}_i\}$, the GP provides a posterior predictive distribution for the function value at any new candidate point $\mathbf{w}_*$. This posterior is also Gaussian, characterized by a mean $\mu(\mathbf{w}_*|\mathcal{D}_{1:t})$ and variance $\sigma^2(\mathbf{w}_*|\mathcal{D}_{1:t})$, which quantify the model's prediction and its uncertainty, respectively.

In this work, the kernel function was chosen as a combination of a Matern Kernel($\nu = 2.5$) and a Spectral Mixture Kernel to effectively capture both smooth (as the metrics were expected to be somewhat smooth, without the kink located at $\min_{\Phi_2}(\Delta\Phi_2^{opt})$) and periodic trends in the objective landscape as can be seen in the true functions in Figure 2.28 and 2.29. These are further defined below since the behaviour and assumptions of a Gaussian Process are governed by its covariance function, or kernel, $k(\mathbf{w}, \mathbf{w}')$. This kernel essentially defines the similarity between two data points $\mathbf{w}$ and $\mathbf{w}'$.

Kernel Functions

Matern Kernel

The Matern kernel is a stationary kernel well-suited for modelling functions that are smooth but not necessarily infinitely differentiable. It is parameterised by a smoothness parameter ν, which controls the differentiability of the resulting function. The general form of the Matern kernel is:

$$k_{\text{Matern}}(\mathbf{w}, \mathbf{w}') = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu} r}{\ell} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu} r}{\ell} \right) \tag{A.2}$$

where:

- $r = \|\mathbf{w} - \mathbf{w}'\|$ is the Euclidean distance between the inputs,
- $\Gamma(\cdot)$ is the Gamma function,
- $K_\nu(\cdot)$ is the modified Bessel function of the second kind,
- ℓ is the lengthscale parameter, which determines how quickly the correlation decays with distance,
- σ^2 is the output variance (signal amplitude) parameter.

Common choices for ν are $\nu = 3/2$ or $\nu = 5/2$, which yield once- and twice-differentiable functions, respectively. The code in this work used $\nu = 5/2$.

Spectral Mixture Kernel

The Spectral Mixture (SM) kernel is a flexible kernel designed to model complex, periodic patterns by approximating its spectral density with a mixture of Q Gaussians [21]. This allows it to capture multiple periodicities and their harmonics. The kernel is derived by taking the inverse Fourier transform of a Gaussian mixture on the frequency spectrum:

$$k_{\text{SM}}(\mathbf{w}, \mathbf{w}') = \sum_{q=1}^{Q} \alpha_q \exp\left(-2\pi^2\tau^2\sigma_q^2\right) \cos(2\pi\tau\mu_q) \tag{A.3}$$

where:

- $\tau = |\mathbf{w} - \mathbf{w}'|$ is the distance between inputs,
- For each component q:
  - α_q is the mixture weight (the magnitude),
  - μ_q is the mean of the Gaussian in the frequency domain (defining the central periodicity),
  - σ_q is the variance of the Gaussian (defining the lengthscale or "width" of the periodicity).

The SM kernel is highly expressive and is particularly effective for signals that are a superposition of multiple quasi-periodic components, making it a good choice given the periodicity of the Φ_2 domain and $\Delta\Phi_2^{opt}$.

2. Selection of Next Query Point via Acquisition Function

The acquisition function $\alpha(\mathbf{w}; \mathcal{D}_{1:t})$, which leverages the GP posterior, balances the exploration of uncertain regions with the exploitation of known promising areas to propose the next point $\mathbf{w}_{t+1}$ to evaluate:

$$\mathbf{w}_{t+1} = \arg\max_{\mathbf{w}\in\mathcal{A}} \alpha(\mathbf{w}; \mathcal{D}_{1:t}) \tag{A.4}$$

The Upper Confidence Bound (UCB) acquisition function was employed:

$$\alpha_{\text{UCB}}(\mathbf{w}; \mathcal{D}_{1:t}) = -\mu(\mathbf{w}|\mathcal{D}_{1:t}) + \beta_t\sigma(\mathbf{w}|\mathcal{D}_{1:t}) \tag{A.5}$$

where the negative mean $-\mu(\mathbf{w})$ promotes exploitation (minimization) and the term $+\beta_t\sigma(\mathbf{w})$ encourages exploration. The parameter β_t controls this trade-off and was scheduled to decay over iterations, favoring more exploitation as the model becomes more certain.

The selected candidate $\mathbf{w}_{t+1}$ is then evaluated on the true, expensive objective function (in this case, by calculating the meta-loss $\Delta = \Delta_1 + \Delta_2$ over a batch of beam profiles), the dataset is updated $\mathcal{D}_{1:t+1} = \mathcal{D}_{1:t} \cup \{\mathbf{w}_{t+1}, \mathcal{L}_{t+1}\}$, and the GP surrogate is retrained. This loop continues until a convergence criterion or a maximum number of iterations N_{iter} is reached, yielding an optimized weight vector $\mathbf{w}^*$.

A.2. Optimizing KL Divergence

In the case for the triple harmonic optimization of the profile itself, a similar technique is employed, but now via the Optuna framework to optimize the five-dimensional parameter space governing the triple harmonic RF system of the PSB in simulation [1]. The objective is to minimize the Kullback-Leibler (KL) Divergence L_{KL} of the resulting longitudinal bunch profile $\lambda_t(t)$, which serves as a sole metric for potential well flatness. The parameters optimized are:

- $\mathbf{w}_1$: Total RF voltage, $V_{\text{total}} \in [6, 20]$ kV
- $\mathbf{w}_2$: Second harmonic voltage ratio, $r_2 = V_2/V_1 \in [0.0, 5.0]$
- $\mathbf{w}_3$: Third harmonic voltage ratio, $r_3 = V_3/V_1 \in [0.0, 5.0]$
- $\mathbf{w}_4$: Second harmonic phase, $\phi_2 \in [0, 2\pi]$ rad
- $\mathbf{w}_5$: Third harmonic phase, $\phi_3 \in [0, 2\pi]$ rad

Sampling and Optimization Strategy

Optuna's default `TPESampler` (Tree-structured Parzen Estimator) was used. This sampler models the probability distributions $l(\mathbf{w})$ and $g(\mathbf{w})$ of the parameters $\mathbf{w}$ that yield losses below and above a given quantile threshold, respectively. It then proposes new candidate points by maximizing the ratio $l(\mathbf{w})/g(\mathbf{w})$, effectively focusing the search on the most promising regions of the parameter space. This is defined mathematically in Section 4.8.

Objective Function and Constraint Handling

The objective function $\mathcal{L}(\mathbf{w}) = L_{\text{KL}}(\lambda_t)$ is evaluated by running a full BLonD simulation for a given parameter set $\mathbf{w}$. To ensure physical feasibility, a hard constraint based on the calculated RF bucket area is imposed *before* the simulation is run. If the bucket area is insufficient for stable capture, the trial is pruned and a high penalty is returned, significantly accelerating the optimization process by avoiding futile simulations.

B

Double-Harmonic Optimization: Verification Plots

B.1. CBAM Model

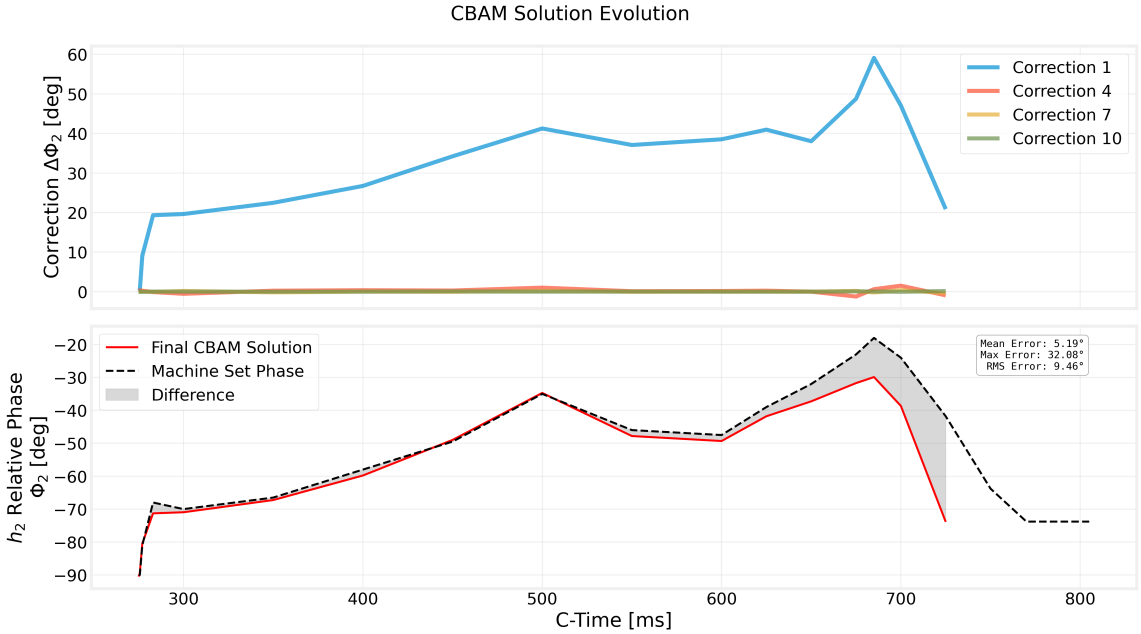


Figure B.1: AD Ring 2 phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

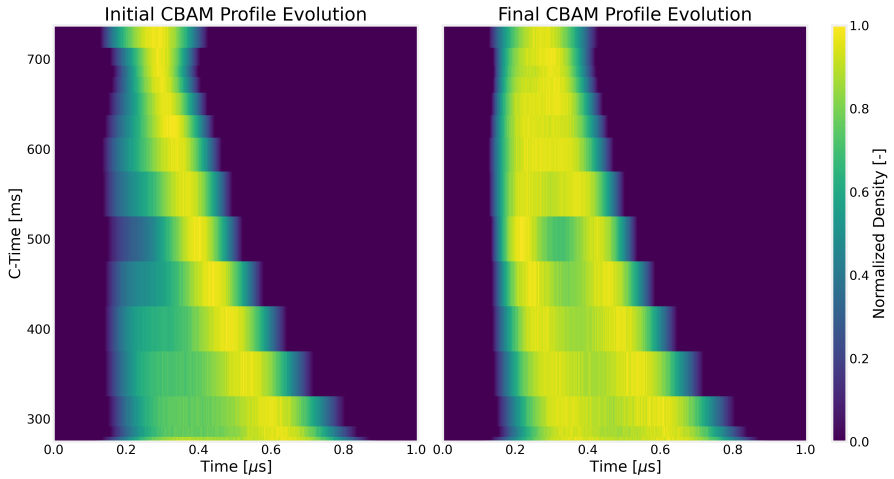


Figure B.2: AD Ring 2 normalized longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

B.2. LSTM-TD3 Agent

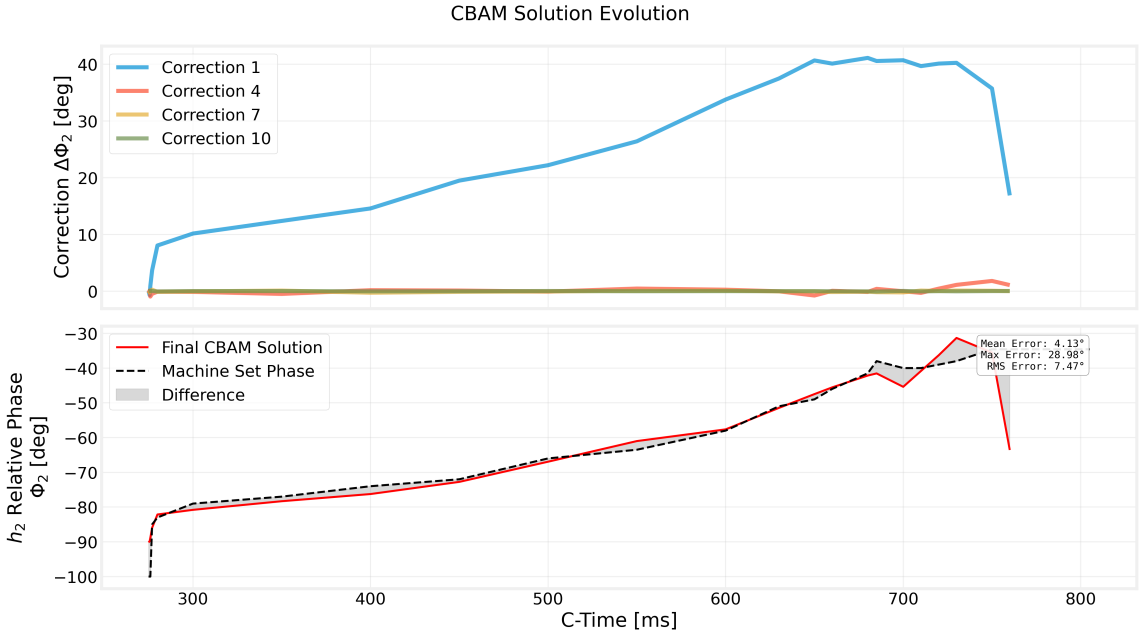


Figure B.3: AD Ring 3 phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

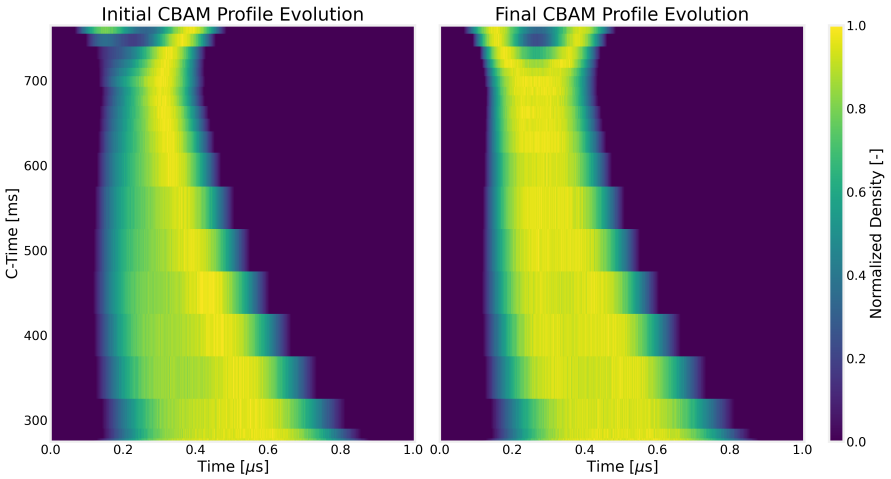


Figure B.4: AD Ring 3 normalized longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

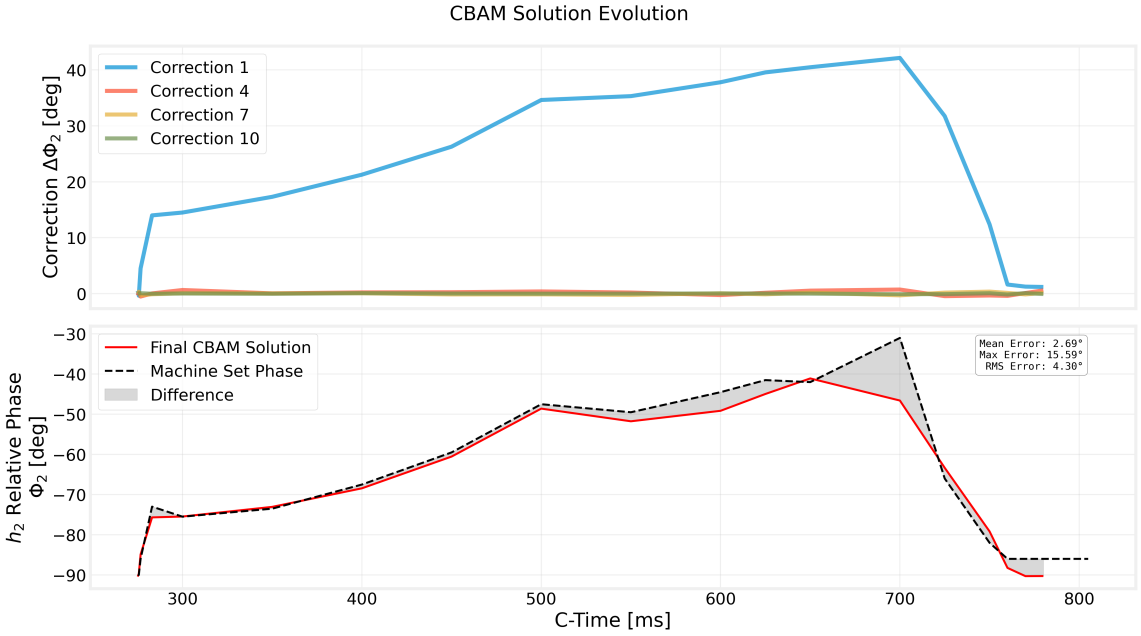


Figure B.5: EAST Ring 2 phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

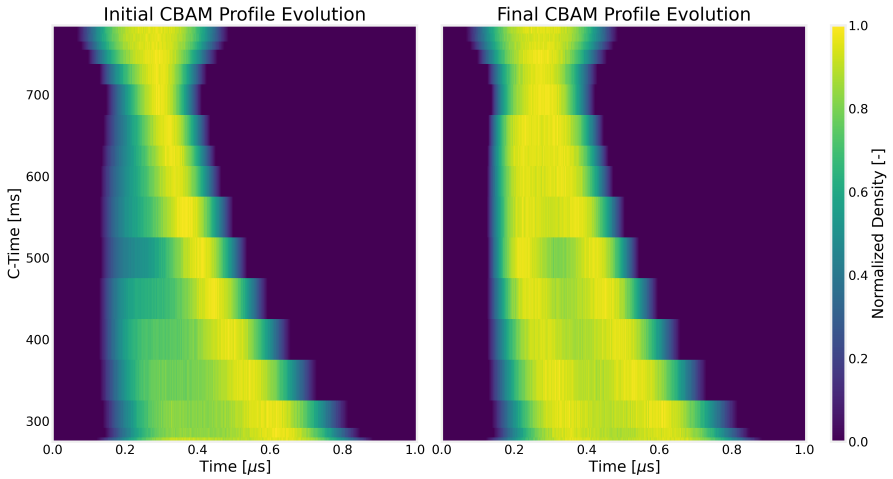


Figure B.6: EAST Ring 2 normalized longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

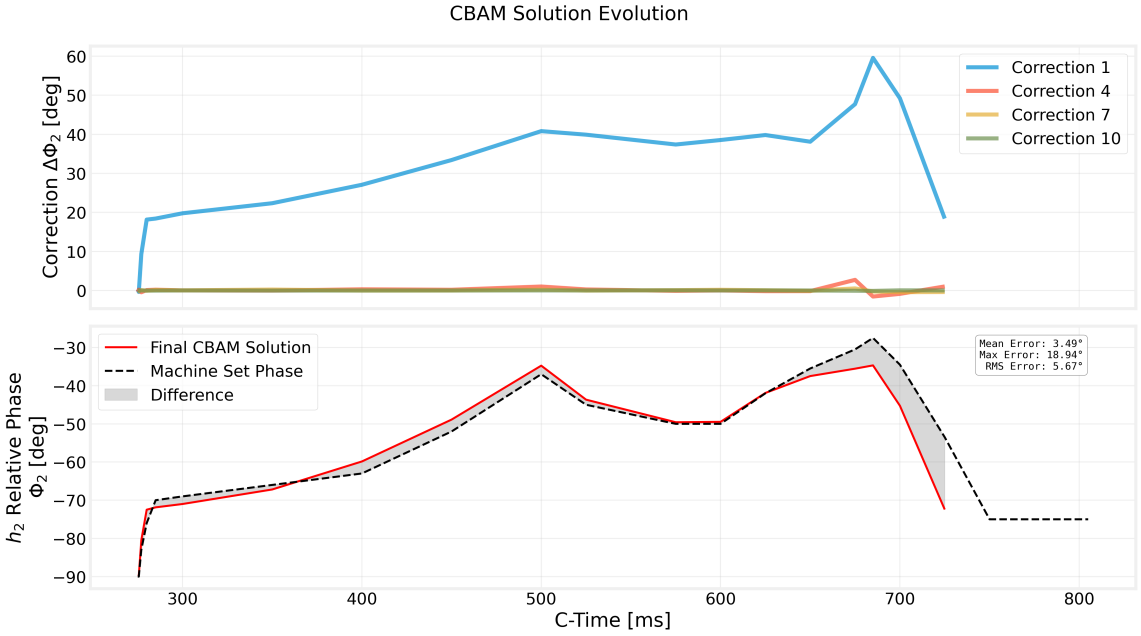


Figure B.7: EAST Ring 3 phase-correction evolution over 10 decaying-weight steps (top) and final CBAM Φ_2 solution versus machine set phase (bottom).

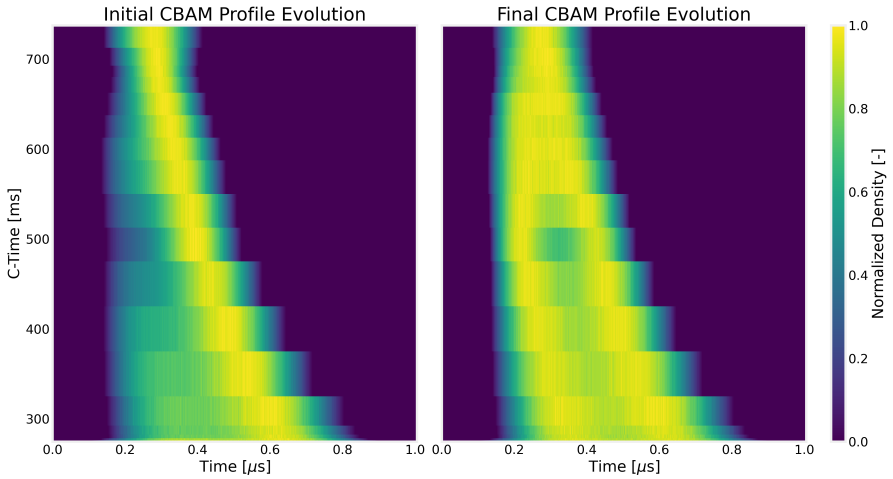


Figure B.8: EAST Ring 3 normalized longitudinal profile evolution: initial (left) and final after CBAM corrections (right).

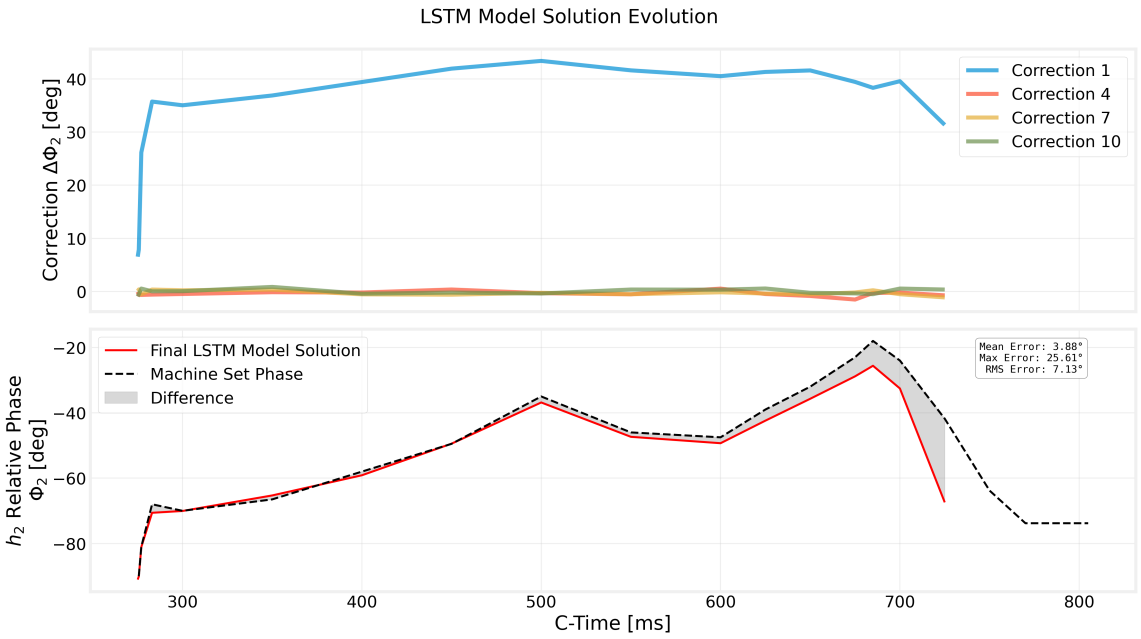


Figure B.9: AD (Ring 2) phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

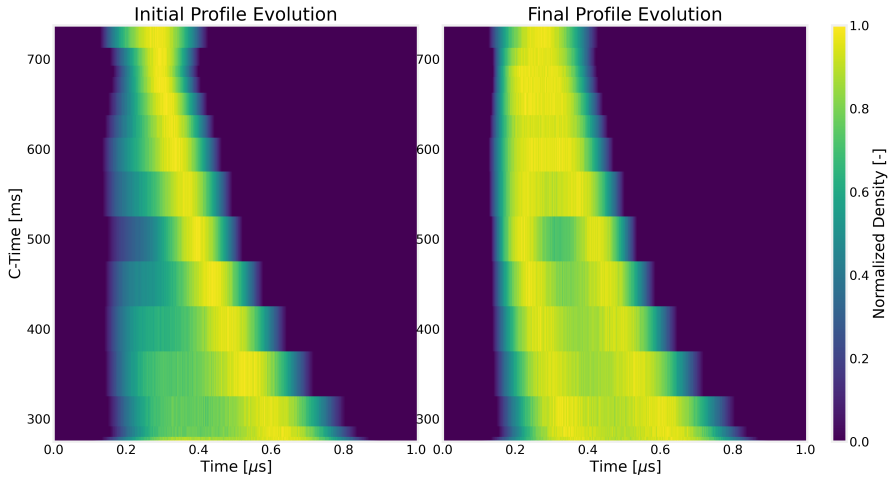


Figure B.10: AD (Ring 2) normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

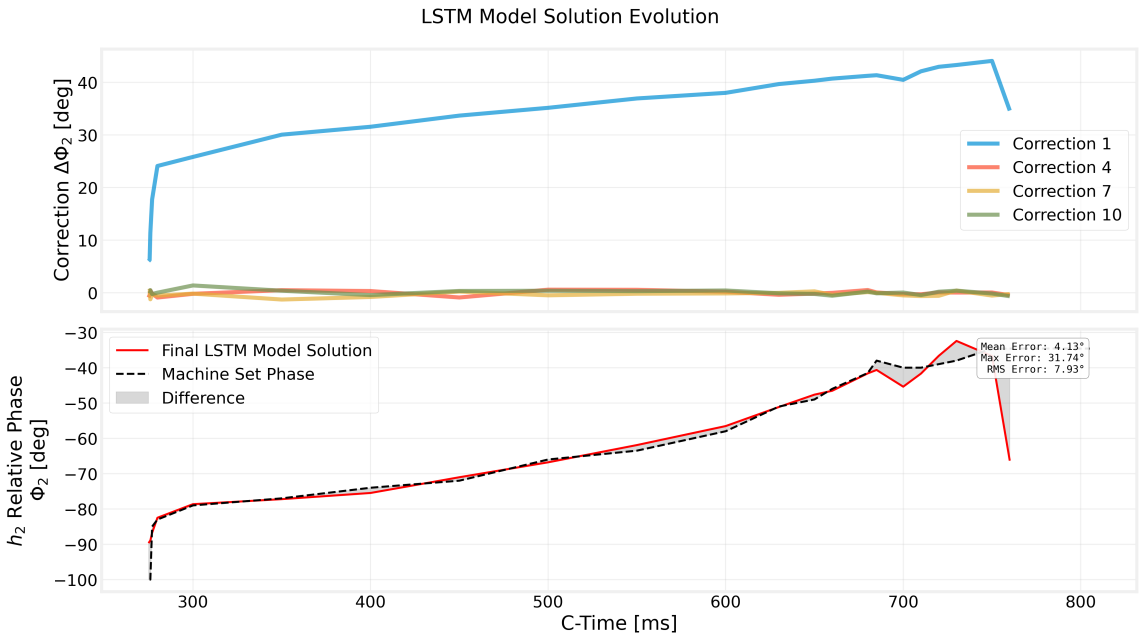


Figure B.11: AD (Ring 3) phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

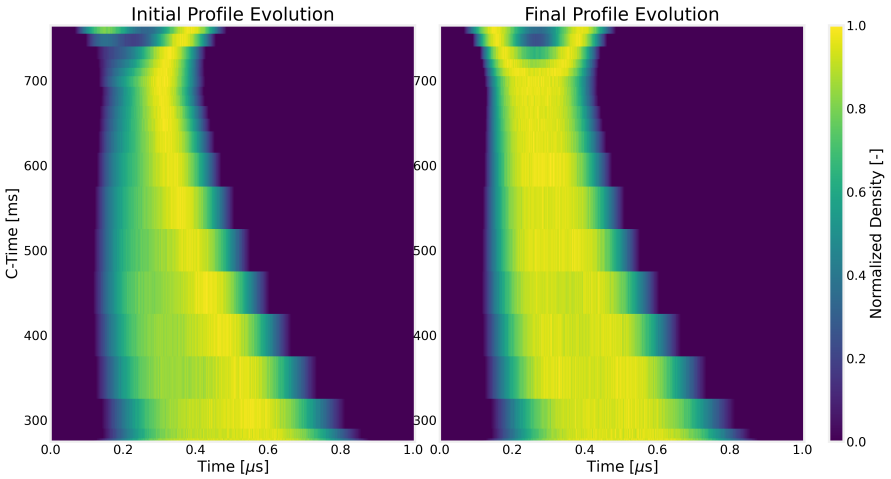


Figure B.12: AD (Ring 3) normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

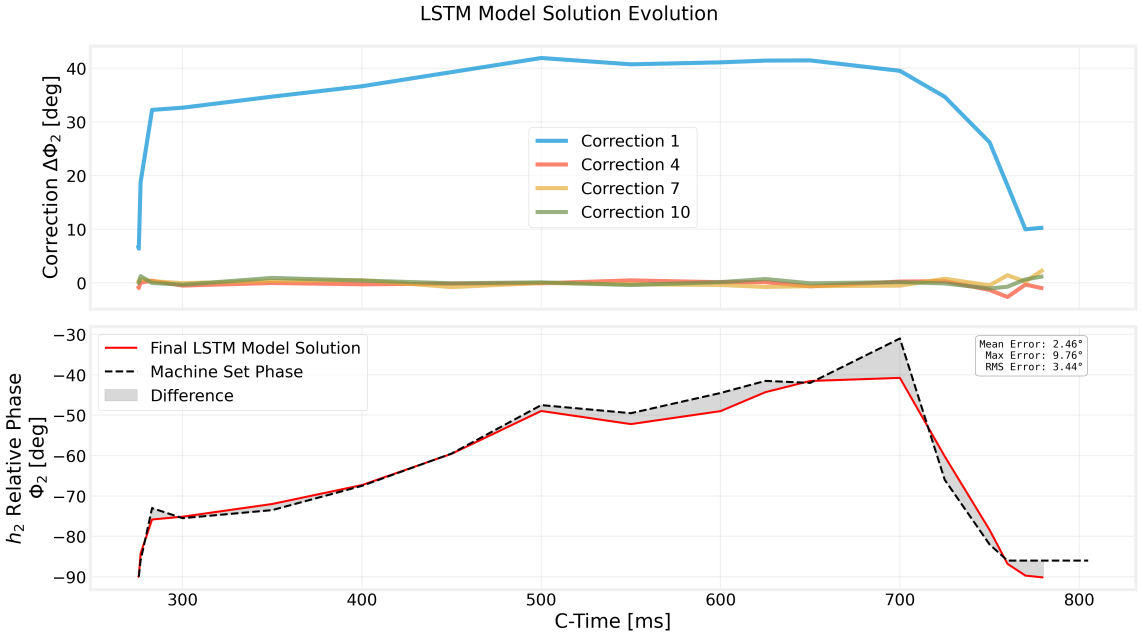


Figure B.13: EAST (Ring 2) phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

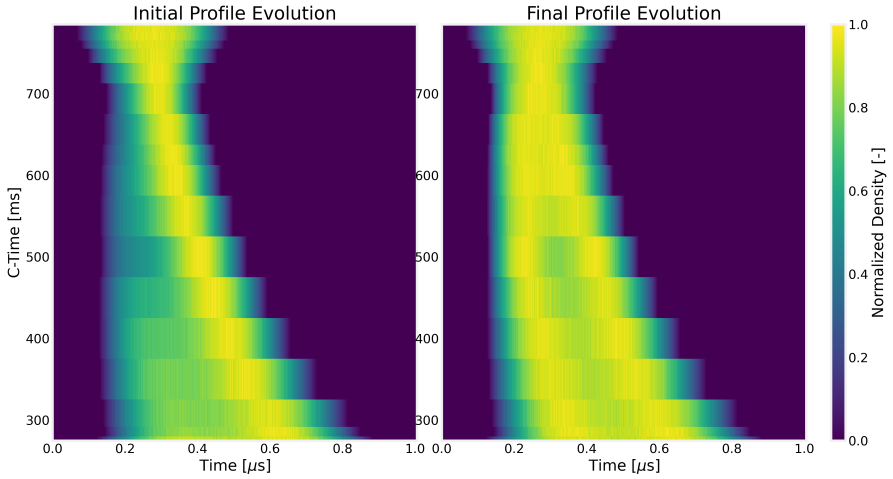


Figure B.14: EAST (Ring 2) normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

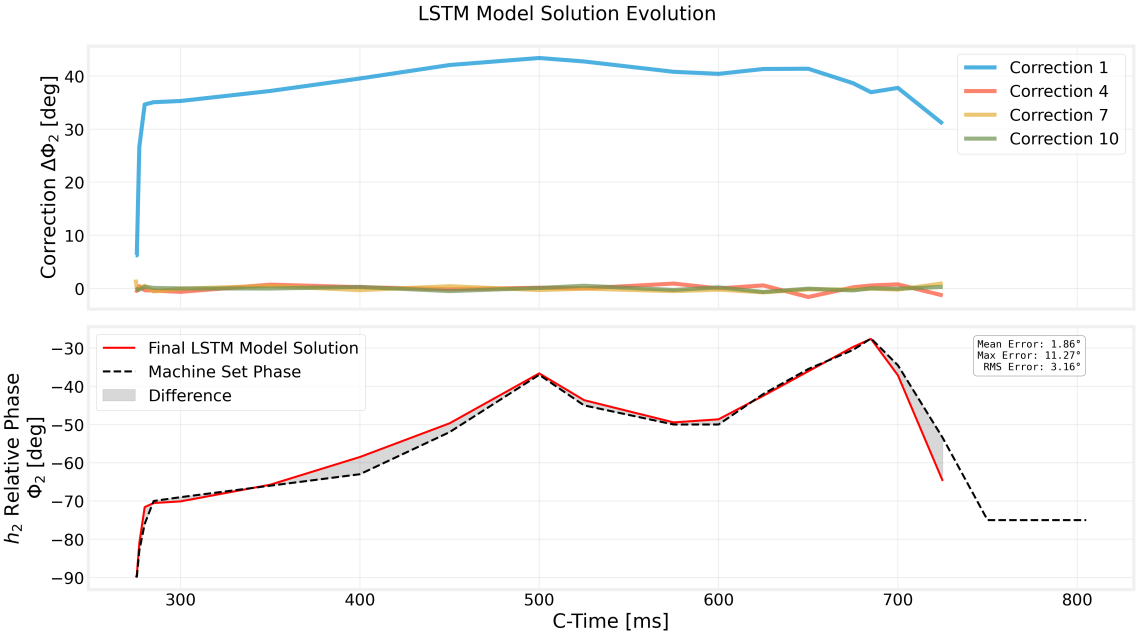


Figure B.15: EAST (Ring 3) phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom).

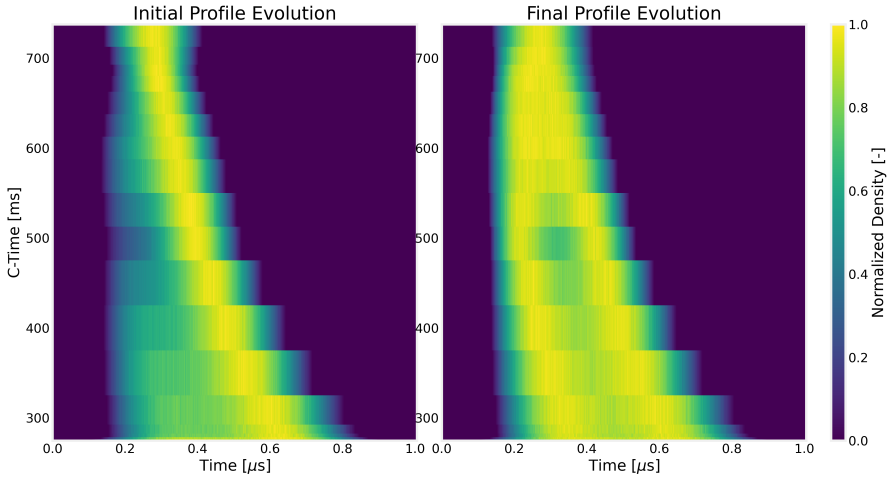


Figure B.16: EAST (Ring 3) normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right).

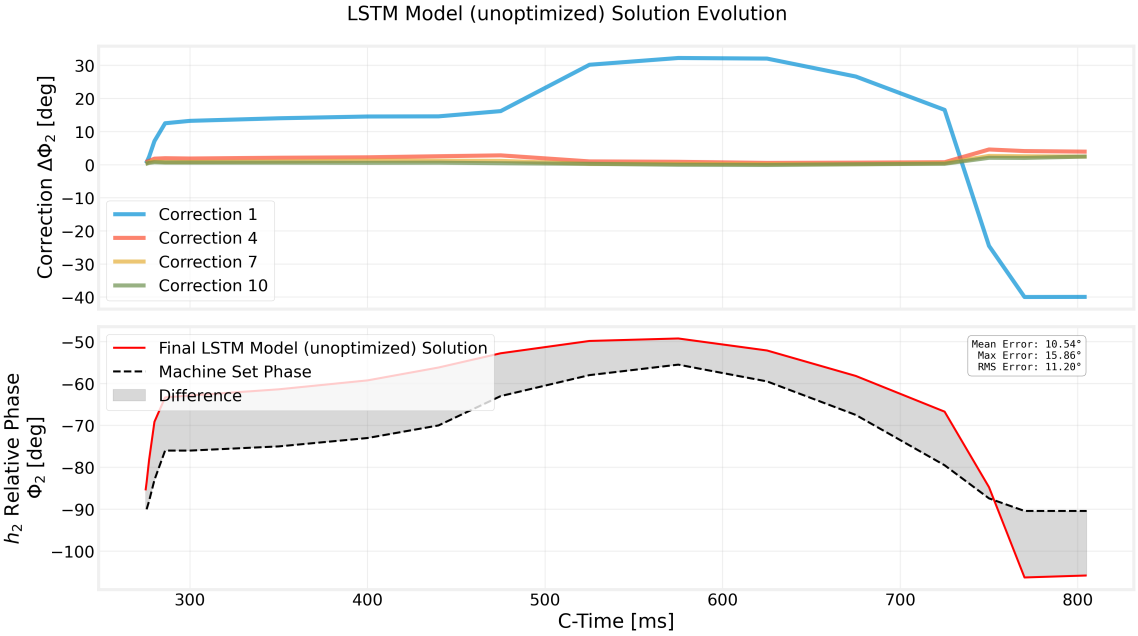


Figure B.17: ISOLDE phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom) for unoptimized model tested in the PSB.

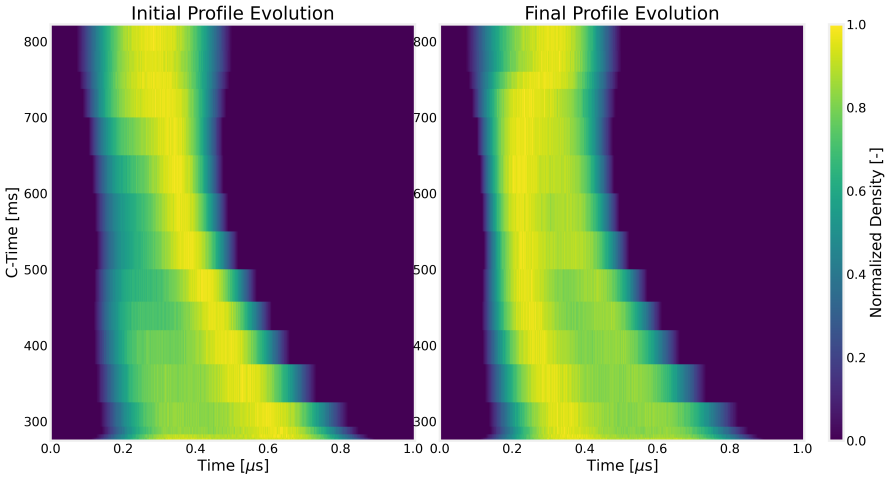


Figure B.18: ISOLDE normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right) for unoptimized model tested in the PSB.

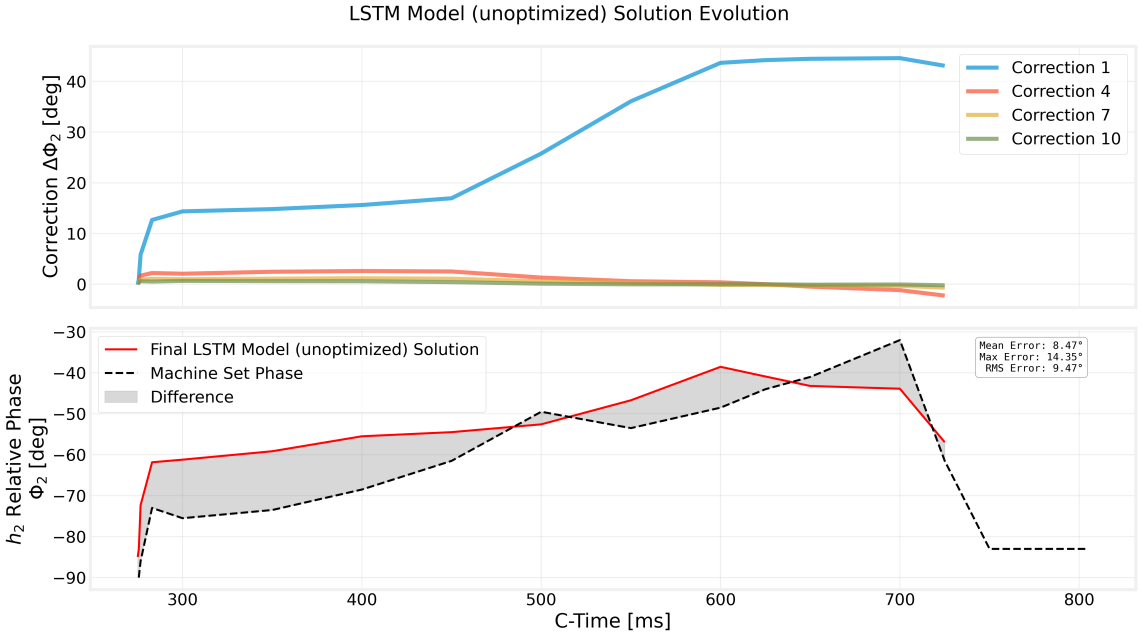


Figure B.19: TOF phase-correction evolution over 10 corrections (top) and final LSTM-TD3 Φ_2 solution versus machine set phase (bottom) for unoptimized model tested in the PSB.

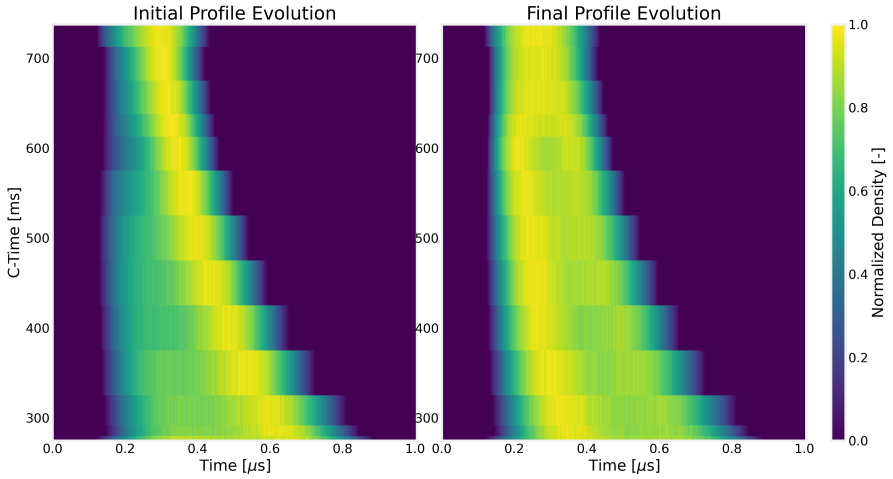


Figure B.20: TOF normalized longitudinal profile evolution: initial (left) and final after LSTM-TD3 corrections (right) for unoptimized model tested in the PSB.