Dynamic Mode Decompostion for Aquifer Thermal Energy Storage

Learning a linear model for control of ATES

S.C. van Muiden

Master of Science Thesis

TU Delft Delft University of Technology

Delf Center for Systems and Control

Dynamic Mode Decomposition for Aquifer Thermal Energy Storage

Learning a linear model for control of ATES

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

S.C. van Muiden

July 26, 2024

Faculty of Mechanical Engineering (3mE) · Delft University of Technology

Abstract

This thesis investigates the application of dynamic mode decomposition (DMD) for the modelling of aquifer thermal energy storage (ATES) systems, which are crucial for reducing the energy used for heating and cooling of buildings. ATES systems store thermal energy underground, using the natural temperature differences between seasons. The research aims to develop a linear model suitable for control purposes, specifically for integration into model predictive controllers (MPC).

DMD is a data-driven method for finding the discrete-time Koopman operator. The required data for DMD is gathered by running high-fidelity simulations in MODFLOW. To identify the Koopman operator, the data is lifted with spatial and time-delayed coordinates. Three DMD algorithms are applied to this lifted data. Firstly, DMD with control (DMDc), this is the most basic DMD algorithm for systems with control input. Secondly, we apply physics-informed DMD (piDMD). This algorithm enforces a local physical constraint. This means that points are only influenced by nearby points. Thirdly, a new DMD algorithm is developed. Gershgorin DMD (GeDMD) combines ideas from piDMD with a constraint on the Gershgorin norm in the DMD optimization to penalize instability. A stable and local system can now be learned.

The DMD models are evaluated on a multi-year prediction horizon and compared to a non-linear analytical ATES temperature model from the literature. The DMDc algorithm outperforms both the analytical model from the literature and the other two DMD algorithms, piDMD and GeDMD. PiDMD is able to correctly enforce the local structure in the model but creates unstable models. The GeDMD algorithm creates a stable and local model but does not reach the same performance as DMDc.

In conclusion, DMDc is able to learn a linear hybrid model of ATES that is usable in an MPC. If better predictions are desired at the cost of more model complexity, research into bi-linear DMD is recommended since this approximates the dynamics from the PDEs better. Also, deep DMD is recommended to discover more complex observables in order to find a better approximation of the Koopman operator. To validate the models, they should be tested more extensively on more challenging ATES conditions.

Acknowledgements

I would like to extend my gratitude to my thesis advisor, Mohammad Khosravi. Our weekly discussions were engaging and provided a thorough understanding of the mathematical theory behind DMD.

I am also very grateful to Martin Bloemendal, whose assistance in understanding and modelling ATES was invaluable. Thank you for your clear and quick responses to my numerous emails regarding ATES. Dankjewel!

Delft, University of Technology S.C. van Muiden
July 26, 2024

Acronyms

$V_{\mathbf{sto}}$ Storage capacity. 31, 32

$T^{\mathbf{C}}_{\mathbf{inj}}$ Cold well injection temperature. 32

$T^{\mathbf{H}}_{\mathbf{inj}}$ Hot well injection temperature. 32

AATM analytical ATES-well temperature model. 10, 37, 39–42

ARX autoregressive exogenous model. 10

ATES aquifer thermal energy storage. 1, 3, 4, 18, 31

DMD dynamic mode decomposition. 2, 13–15, 26

DMDc DMD with control. i, 16

EDMD extended DMD. 17

GeDMD Gershgorin DMD. i, 21

HDMD Hankel DMD. 17

HT-ATES high temperature ATES. 4

MPC model predictive controller. 1, 2, 6

PDE partial differential equation. 10, 31, 33, 34, 46

piDMD physics-informed DMD. i, 19

RC resistance capacitance. 11

RMSE root mean squared error. 37, 39

SVD singular value decomposition. 15, 16, 29

Table of Contents

Chapter 1

Introduction

Heating and cooling of buildings account for almost 50% of the final global energy consumption [1, 2]. In order to limit global warming it is important to reduce energy used in buildings. The high energy consumption of buildings is caused by the temporal mismatch in heating and cooling demand. Energy consumption is high in winter when buildings are heated, and high in summer when cooled. If the heat from the summer could be used in winter and vice versa for the cold, this could greatly reduce the energy consumption of buildings.

Aquifer Thermal Energy Storage is a way of storing this thermal energy underground. An aquifer thermal energy storage (ATES) system consists of two wells (boreholes) through which water can be pumped in and out of the ground. Typically, one well is considered a hot well, and the other is a cold well. In summer, cold water is extracted from the cold well and used to cool the building, the heated water from the building is injected back into the ground via the hot well. In winter this process is reversed. Due to the moderate climate and availability of sandy aquifers, the Netherlands is a very suitable place for ATES systems. There are already over 3000 systems in use in the Netherlands [3]. Chapter 2 gives more background on the working principles of ATES and its physics, as well as some simplified models from literature.

To make optimal use of the storage capabilities of ATES, advanced control methods such as model predictive controllers (MPCs) are required. These control methods rely on a model to predict how the system evolves over time. Therefore, models are required to predict the amount and temperature of the stored heat and cold. Since the storage of ATES is far below the surface, it is very expensive to measure the actual state of the system. In section 2-2, more details about the control and modelling challenges are provided.

This thesis focuses on learning a model for ATES for control. The model should be able to predict how much useable thermal energy is stored in the ground. This information can then be combined with weather predictions and thermal models for buildings to control ATES optimally. For example, if not much heat is stored in the ground after a mild summer, and the winter is predicted to be very cold. Extra heat could be generated by using a heat pump

relatively cheaply when it is still warm outside. However, to be able to make this decision, one must know how much heat is actually stored in the ground.

The main goal of this thesis is to develop a control-oriented model for the output temperature of ATES. Here, we consider a control-oriented model as a model that can be implemented in an MPC. Therefore, a linear model would be optimal. Various methodologies and tools are developed for the analysis and control of linear dynamical systems, leading to more tractable approaches.

For learning this model, the dynamic mode decomposition (DMD) algorithm is chosen. The choice for DMD is made because it has been proven to work on high-dimensional systems and highly non-linear fluid flows. More information on dynamic mode decomposition (DMD) can be found in section 3-2. Learning a linear model for a highly non-linaer system may sound naive, however Koopman operator theory provides a framework where with the use of observables, non-linear dynamics can be lifted to a linear coordinate frame. More on Koopman operator theory can be found in section 3-1.

This thesis implements two existing DMD algorithms: DMD with control (subsection 3-3-1) and physics-informed DMD (subsection 3-3-3). A new DMD algorithm has also been developed and implemented. The new DMD algorithm is able to enforce stability and locality constraints. The algorithm is named Gershgorin DMD and is elaborated upon in subsection 3-3-3.

These models are trained using simulation data. The choice for simulation data is made since ATES data is hard to come by since temperature measurements of the subsurface are very expensive and even impossible for the full subsurface. The timescale for ATES is also very long. Decades would be required to gather enough input-output data to identify the dynamics correctly. More on the simulation strategy and the ATES system parameters can be found in chapter 5.

Chapter 2

Aquifer thermal energy storage

This chapter first explains the working principle of ATES in section 2-1. Then a more in-depth discussion about the physics governing the system dynamics follows in section 2-3. Finally three simplified models are explained in section 2-4.

2-1 Working principle

In this thesis the ATES system considered is the doublet as this is the most often implemented system. A doublet means that there are two wells, a hot and a cold one. Other systems exist with one or even three wells.

First, some ground properties need to be explained. The ground consists of multiple layers, roughly speaking they can be categorised into two groups: a permeable layer called *aquifer* and an impermeable layer called *aquitard*. Permeable means that water can flow easily through the ground. More precisely, it is hydraulically conductive. In the Netherlands, most often, the aquitard consists of unconsolidated clay or peat, which are not hydraulically conductive. The aquifer consists of sand which is hydraulically conductive [5]. In an ideal scenario, the ATES well is placed in a confined aquifer, which means that the aquifer's top and bottom are covered by the aquitard, ensuring that the stored heat or cold does not flow up or down, only sideways.

The wells are constructed by drilling down to the aquifer and placing a long perforated screen over the full length of the aquifer. In Figure 2-1, the aquifer is indicated in yellow, and the aquitard in brown. It can be seen that the well screen is placed in the aquifer between two layers of the aquitard. Here, water can be pumped in and out of the aquifer. Typically, one well is considered the hot well and the other the cold well. In summer cold water is extracted from the cold well and used to cool the building, the heated water is then injected back into the ground via the hot well. In winter, this process is reversed. These are the two operating modes: *hot injection* and *cold injection*. A third operating mode can be defined: *storage*, which is when there is no flow into or out of the well. Since the dynamics of the wells are

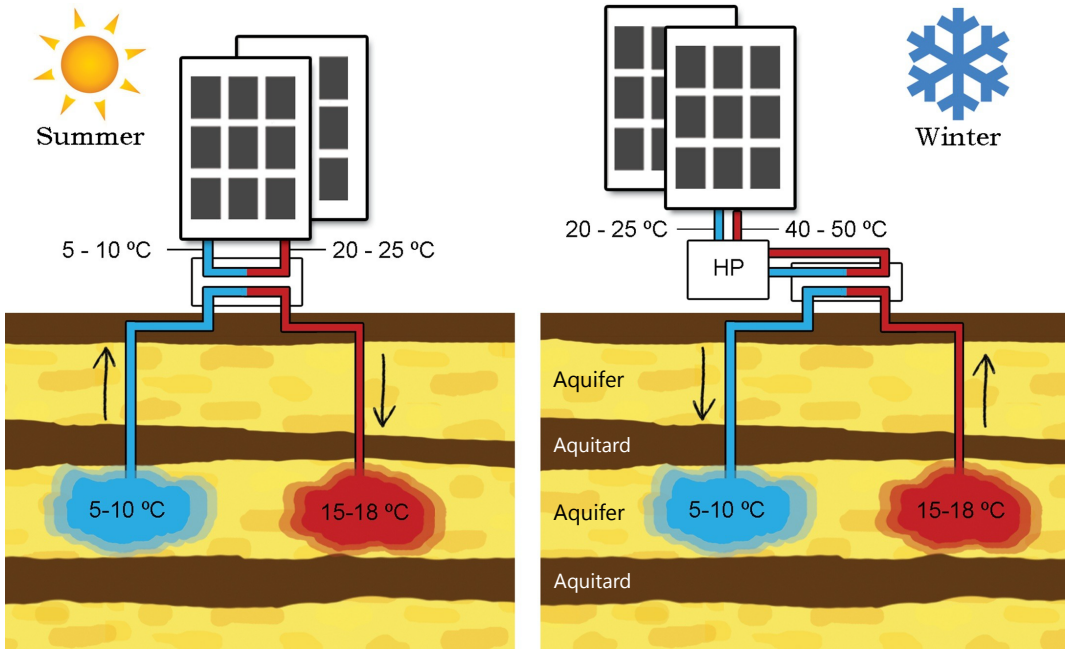

Figure 2-1: Basic working principle of an ATES system. Brown indicates the poorly permeable aquitard, and yellow indicates the good permeable sandy aquifer. **Left:** In summer, water is pumped out of the cold well and used via a heat exchanger to cool down the building, the heated water is pumped back into the hot well. **Right:** In winter the process is reversed, warm water is pumped out of the well and used to heat the building. A heat pump may be used to increase the heating temperature. The cold water is returned to the cold well [4].

very similar, we first zoom in on just one well. From this perspective, we call the operating modes *injection* and *extraction*.

The system is closed because no groundwater enters or leaves the system. After extracting the hot or cold, the extracted water is immediately returned to the opposing well. This means that the flow out of one well is always equal to the flow into the other. In other words, the flows are coupled.

The temperatures at which ATES operates are relatively low, the ambient groundwater temperature in shallow aquifers, where ATES systems are placed, is approximately 12 °C. The injection temperatures are mostly constant throughout the year. The cold injection temperature is very constant because it is at the outlet of a heat pump. The hot injection temperature can vary a bit more. If it is very warm outside and more cooling is required, the temperature increases. However, the temperature can never be higher than that of the space that it is cooling down. The minimal cold water injection temperature is 5 °C, and the maximum injection temperature (by law in the Netherlands) is 25 °C. At these temperatures, there is no disruption to the soil [6]. However, it is necessary to maintain an energy balance so no energy is extracted or put into the ground over a long period. Note that ATES is only a way of storing energy and should not be confused with geothermal energy. With ATES no energy is extracted from the ground, which is the case with geothermal energy. High temperature ATES (HT-ATES) is also researched where the hot injection temperatures can be as high as 90 °C. However, HT-ATES is not jet widely implemented [7].

The dimensions of an ATES system depend mainly on the local geohydrological conditions and the required storage volume. For example, in Amsterdam, the suitable aquifer is located at a depth of 70 to 200m, while in Utrecht, the appropriate aquifer is at 3 to 50m of depth [8, 9]. Not only does the depth of the aquifer influence the performance of the system, but the hydraulic conductivity and the heterogeneity of the aquifer have a great influence. The heterogeneity is how non-uniform the ground conditions are.

2-1-1 Inputs and outputs of ATES system

The inputs and outputs of an ATES well are the flow, the injection temperature and the extraction temperature. For the hot well, we have the following in and outputs

V_{in}^{H} the flow rate (or volume) into the hot well,
$\text{T}_{\text{inj}}^{\text{H}}$ the temperature of the water entering the well, and
$\text{T}_{\text{ext}}^{\text{H}}$ the temperature of the water extracted from the well.

Similar for the cold well

V_{in}^{C} the flow rate (or volume) into the cold well,
$\text{T}_{\text{inj}}^{\text{C}}$ the temperature of the water entering the cold well, and
$\text{T}_{\text{ext}}^{\text{C}}$ the temperature of the water extracted from the cold well.

Since the systems are coupled $V_{\text{in}}^{\text{H}} = -V_{\text{in}}^{\text{C}}$, to make it easier in the rest of this thesis, only V^{in} is used ($V^{\text{in}} = V_{\text{in}}^{\text{H}} = -V_{\text{in}}^{\text{C}}$). The units of V^{in} can change depending on the use, it is either a volume flow or a fixed added volume, which in discrete-time are proportional to each other with the sampling time.

2-2 ATES control and modelling challenge

For effective control and optimal utilization of the storage capabilities of ATES, it is important to know how much heat and cold is stored in the ground. Since the thermal storage of ATES occurs far below the surface, measuring the actual state of the system is very expensive and challenging. This is because it involves drilling down to the depth of the aquifer, which is costly. Moreover, even after drilling, temperature measurements can only be taken at the specific locations where the wells and temperature sensors are situated. Obtaining a full grid of measurements across the entire storage volume is practically impossible due to these constraints.

Therefore, predictive models are required to estimate the amount of stored heat and cold. While a direct estimation of the storage volume can be useful, it is more critical to predict the temperature of the water extracted from the ground. The key information needed is when the temperature of the extracted water will start to drop off significantly. Typically, injection temperatures remain fairly constant, resulting in relatively stable extraction temperatures until the storage capacity is nearly depleted. At this point, the extraction temperature declines rapidly (for hot storage) or increases rapidly (for cold storage), as shown in Figure 2-2. This change in temperature marks a significant drop in heating or cooling efficiency. If this drop-off can be predicted in advance, more heat could be stored during the summer or more cold during the winter to maximize the storage capacity and ensure optimal system performance. Accurate predictions allow for proactive adjustments, ensuring that the ATES system operates efficiently throughout the year.

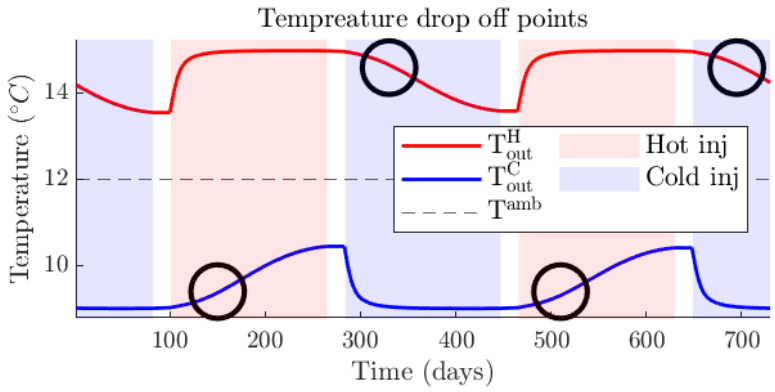

Figure 2-2: Temperature drop-off points where system efficiency decreases.

The main goal of this thesis is to develop a prediction model for the output temperature of ATES that is well-suited for control. To effectively control and optimize ATES, the model must be suitable for implementation in a MPC. A linear model is particularly desirable because various tools and methodologies are established for analyzing and controlling linear systems, making it easier to leverage existing knowledge and techniques. Linear models also enable the use of convex optimization techniques within an MPC.

A second objective of the model is to predict interactions between neighboring wells. When ATES systems are placed close together, they can interfere with each other. This interference can reduce storage efficiency. However, if the wells are placed strategically, the efficiency can also increase [10]. If these interactions are accurately modelled, control strategies can be optimized to either capitalize on the benefits of nearby systems or minimize the energy loss if the system cannot be placed optimally [11]. A solution is to model the full temperature distribution underground. This is a very high-dimensional modelling problem, which is where DMD excels, capturing the important modes in a high-dimensional system.

2-3 ATES physics

To simulate ATES, it is important to understand the physics behind it. Two main processes are the driving force behind the ATES dynamics: groundwater flow and heat transport.

2-3-1 Groundwater flow

The first step is computing how water flows underground. This is done using Darcy's Law. Darcy's law defines the rate of water flow through porous media. It states that the rate of flow per unit of time is proportional to the rate of change of fluid pressure (the hydraulic gradient) with distance [12, 13]. In other words, how much water will flow through a certain area is proportional to the pressure difference and the conductivity of the flow area. Darcy's law for flow through a channel with constant crosse section area is

$$Q = KA\frac{\Delta h}{L}, \text{ with } h = \frac{p}{\rho g}, \tag{2-1}$$

where

Q is the flow ($m^3\,d^{-1}$),
K is the hydraulic conductivity ($m\,d^{-1}$),
A is the area (m^2),
h is the pressure head (m),
L is length (m), and
p is the pressure (Pa).

In one dimension, Darcy's law is equivalent to Ohms Law: $q = -k\frac{\partial h}{\partial x}$ and $i = -\sigma\frac{\partial V}{\partial x}$. In other words, the flow of current/fluid is proportional to the voltage/pressure difference with the hydraulic/electric conductivity.

Rewriting this equation in differential form, or over an infinitesimally small area and length for all three dimensions, Darcy's Law becomes:

$$q_i = -\mathrm{K}_i \frac{\partial h}{\partial i} \quad \text{with } \mathrm{K}_i = \frac{\kappa_i \rho g}{\mu} \quad \text{for } i = x, y, z\ , \tag{2-2}$$

where

q_i is the specific discharge ($m\,d^{-1}$),
K_i is the hydraulic conductivity ($m\,d^{-1}$),
h is the pressure head (m),
κ_i is the intrinsic permeability (m^2),
μ is the dynamic viscosity ($kg\,m^{-1}\,d$), and
ρ is the density ($kg\,m^{-3}$).

Note that the used time unit is days (d), a deviation is made from the SI units because the dynamics of ATES are so slow that this is a much more suitable timescale than seconds.

Now, we have the flow related to the head for each dimension separately. A mass balance is performed to couple these equations. In a given volume, the mass does not change. More precisely, the sum of the flow into and out of the control volume is zero. This is shown in Figure 2-3, in equations this becomes:

$$\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} + \frac{\partial q_z}{\partial z} = 0. \tag{2-3}$$

This equation assumes steady-state flow, no storage and no sink or source. Storage is a process where water is released by the aquifer. When the pressure becomes lower, the ground behaves like a sponge. If these phenomena are taken into account, the equation becomes:

$$\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} + \frac{\partial q_z}{\partial z} = S_s \frac{\partial h}{\partial t} - q', \tag{2-4}$$

where S_s is the specific storage and q' is the source or sink discharge.

To obtain the groundwater flow equation, Darcy's Law (2-2) and the mass balance (2-4) are combined to

$$\begin{aligned} \frac{\partial K_x \frac{\partial h}{\partial x}}{\partial x} + \frac{\partial K_y \frac{\partial h}{\partial y}}{\partial y} + \frac{\partial K_z \frac{\partial h}{\partial z}}{\partial z} &= S_s \frac{\partial h}{\partial t} - q', \\ K_x \frac{\partial^2 h}{\partial x^2} + K_y \frac{\partial^2 h}{\partial y^2} + K_z \frac{\partial^2 h}{\partial z^2} &= S_s \frac{\partial h}{\partial t} - q', \end{aligned} \tag{2-5}$$

here, the variables that depend on the temperature are indicated in yellow, the state variables are indicated in red and the inputs in blue. The properties of water (that are influenced by

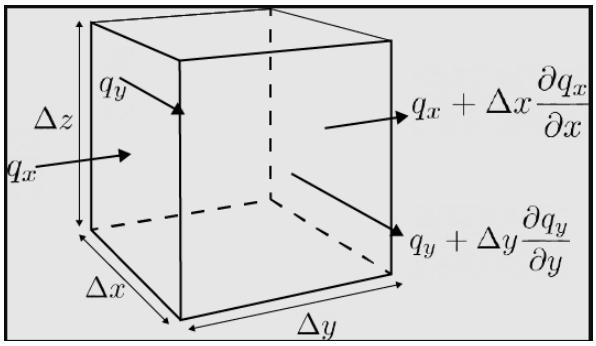

Figure 2-3: Ground water control volume/mass balance [5].

temperature) are captured in the hydraulic conductivity K_i, which depends on the density and viscosity, which both depend on temperature [5, 13–15].

The relevant states for modelling and learning the dynamics of the system in these equations are the flow in all three directions in q_x, q_y and q_z directions. The vector containing the flow in all three directions is notated as q.

2-3-2 Heat transport

Now that we know how water flows underground, it is time to investigate how heat flows underground and how this is affected by the flow. There are four processes governing the ground(water) temperature:

- Thermal retardation, the process where water transfers its heat to the ground.
- Heat conduction, heat is conducted away similar to how heat flows in solids.
- Dispersion, mixing that occurs due to flow and widens the spread of heat.
- Advection and groundwater flow (ambient and source/sinks). Heat is carried away with the flowing water.

These four processes are described in the heat transport equation with again the variables that depend on the temperature indicated in yellow, the state variables indicated in red and the inputs in blue [5]:

$$\underbrace{\left(1 + \frac{(1-\rho_s)\, c_s}{\theta c_f \rho_f}\right) \frac{\partial(\theta T)}{\partial t}}_{\text{Thermal retardation}} = \nabla \left[\left(\underbrace{\theta \left(\frac{\lambda_b}{\theta c_f \rho_f} \right)}_{\text{Heat conduction}} + \underbrace{\alpha \frac{q}{\theta}}_{\text{Dispersion}} \right) \nabla T \right] - \underbrace{\nabla(qT) - q_s' T_s}_{\text{flow (sink/source)}}, \quad (2\text{-}6)$$

where

θ is the porosity (-),
ρ_s, ρ_f is the density of the solid and fluid ($\mathrm{kg\,m^{-3}}$),
c_s, c_f is the specific heat capacity of the solid and fluid ($\mathrm{J\,kg^{-1}\,K^{-1}}$),
λ_b is the bulk thermal conductivity of the aquifer ($\mathrm{W\,m^{-1}\,K^{-1}} = \mathrm{kg\,m\,d^{-3}\,K^{-1}}$),
T_s is the source temperature (K),
q is the specific discharge, the flow from the groundwater equation ($\mathrm{m\,d^{-1}}$),
q' is the source or sink discharge ($\mathrm{m\,d^{-1}}$), and
α is the dispersivity tensor (m).

The temperature (T) and flow (q) are the states of interest in this equation and they vary with time. In this equation, the properties of water are assumed to be constant, while in reality, they also depend on the temperature.

2-3-3 Variable properties of water

The viscosity and density of water are a nonlinear function of the temperature. For the viscosity, an empirical relation with temperature is used [16]:

$$\mu(T) = 239.4 \cdot 10^{-7} \cdot 10^{\left(\frac{248.37}{T+133.15}\right)}, \tag{2-7}$$

note that in this equation, the unit of temperature is Celsius (°C).

The nonlinear density-temperature relation can be modelled using [16]:

$$\begin{aligned} \rho &= \rho_0 \exp\left[\beta_T (T - T_0) + \beta_P (P - P_0)\right], \\ &\text{with} \\ \beta_T &= \frac{1}{\rho}\left(\frac{\partial \rho}{\partial T}\right)_{c,P}, \text{ and} \\ \beta_P &= \frac{1}{\rho}\left(\frac{\partial \rho}{\partial P}\right)_{C,T}, \end{aligned} \tag{2-8}$$

where

T is the temperature (K),
P is the pressure ($\mathrm{kg\,m^{-1}\,d^{-1}}$),
T_0 is the reference temperature (K),
P is the pressure at the reference temperature ($\mathrm{kg\,m^{-1}\,d^{-1}}$),
β_T is the volumetric expansion coefficient for temperature (-), and
β_P is the volumetric expansion coefficient for pressure (-).

For the low-temperature ATES systems that we are modelling, a linearisation can be made of (2-8) [8]:

$$\rho(T) = \rho_0 + \frac{\partial \rho}{\partial T}(T - T_0) + \frac{\partial \rho}{\partial \ell}(\ell - \ell_0), \tag{2-9}$$

where $l = h_0 - z$ is the vertical distance from the reference head h_0 at height l_0.

2-4 Current models for ATES

Besides the very detailed model based on the partial differential equations (PDEs), some simplified analytical models already exist. Three simplified models formulated for control purposes are discussed here.

2-4-1 Basic energy model

The simplest formulated model is a basic energy model. This model considers only the energy aspect of ATES with as input the energy flux added or extracted and as state the amount of energy stored. This can be modelled as a one-dimensional autoregressive exogenous model (ARX), resulting in [17]:

$$Q_{s,k+1} = AQ_{s,k} + Q_{\mathrm{aq},k} = AQ_{s,k} + Bu_k, \tag{2-10}$$

where

- A is a lumped coefficient of losses $A \in [0, 1)$,
- $Q_{s,k}$ is the amount of stored energy,
- $Q_{\mathrm{aq},k}$ is the inlet or outlet of energy,
- B is the input coefficient, and
- u_k is the model input, the flow rate into the well.

The value of the input coefficient can be computed based on the pumping rate

$$B = \rho_w c_w \Delta T_{\mathrm{aq}} \tau, \tag{2-11}$$

with τ the sampling time and ΔT_{aq} is defined as the temperature difference between the hot and cold well.

This model is very simple, which makes it great for control but not very realistic. It is not possible to determine the temperature of the stored energy. The model does not even separate the stored cold and heat.

2-4-2 Analytical ATES-well temperature model

For a more accurate model [18, 19] propose the following analytical ATES-well temperature model (AATM)

$$\begin{aligned}
V^{\mathrm{H}}_{k+1} &= V^{\mathrm{H}}_k + \left(s^{\mathrm{H}}_k - s^{\mathrm{C}}_k\right) V^{\mathrm{in}}_k, \\
V^{\mathrm{C}}_{k+1} &= V^{\mathrm{C}}_k + \left(s^{\mathrm{C}}_k - s^{\mathrm{H}}_k\right) V^{\mathrm{in}}_k, \\
\mathrm{T}^{\mathrm{H}}_{k+1} &= \frac{V^{\mathrm{H}}_k}{V^{\mathrm{H}}_k + s^{\mathrm{H}}_k V^{\mathrm{in}}_k} \mathrm{T}^{\mathrm{H}}_k + \frac{s^{\mathrm{H}}_k V^{\mathrm{in}}_k}{V^{\mathrm{H}}_k + s^{\mathrm{H}}_k V^{\mathrm{in}}_k} \mathrm{T}^{\mathrm{in}}_k - \alpha \frac{\mathrm{T}^{\mathrm{H}}_k - \mathrm{T}^{\mathrm{amb}}_k}{V^{\mathrm{H}}_k + s^{\mathrm{H}}_k V^{\mathrm{in}}_k}, \\
\mathrm{T}^{\mathrm{C}}_{k+1} &= \underbrace{\frac{V^{\mathrm{C}}_k}{V^{\mathrm{C}}_k + s^{\mathrm{C}}_k V^{\mathrm{in}}_k}}_{\text{RHS 1}} \mathrm{T}^{\mathrm{C}}_k + \underbrace{\frac{s^{\mathrm{C}}_k V^{\mathrm{in}}_k}{V^{\mathrm{C}}_k + s^{\mathrm{C}}_k V^{\mathrm{in}}_k}}_{\text{RHS 2}} \mathrm{T}^{\mathrm{in}}_k - \alpha \underbrace{\frac{\mathrm{T}^{\mathrm{C}}_k - \mathrm{T}^{\mathrm{amb}}_k}{V^{\mathrm{C}}_k + s^{\mathrm{C}}_k V^{\mathrm{in}}_k}}_{\text{RHS 3}},
\end{aligned} \tag{2-12}$$

where

s^{H} is a binary variable ($s_k^{\mathrm{H}} \in \{0,1\}$) indicating the ATES operating mode. $s_k^{\mathrm{H}} = 1$ if hot water is being injected,
s^{C} is a binary variable ($s_k^{\mathrm{C}} \in \{0,1\}$) indicating the ATES operating mode. $s_k^{\mathrm{C}} = 1$ if cold water is being injected,
V^{H} & V^{C} are the stored volumes of the hot and cold well respectively (m^3),
T^{H} & T^{C} is the temperature of the stored volumes of the hot and cold well respectively (K),
V^{in} & T^{in} is the volume of water injection at timestep k with temperature T^{in}, and
α is a loss term, $\alpha \in (0,1)$[1].

The first term on the right-hand side (RHS) is the percentage of the old volume that is present at time step $k+1$. The term RHS 2 represents the percentage of the new volume. The sum of the old volume and the new volume is always equal to 1: RHS 1 + RHS 2 = 1. The third term (RHS 3) is the temperature difference divided by the new volume. The physical interpretation is that the loss of temperature is now proportional to the temperature difference. Thus, a higher temperature difference results in a higher temperature loss. But inversely proportional to the volume, which is to be expected since the larger the volume the less the loss since the ratio of area over volume is better.

This is a very good interpretable model but comes at the cost of being a nonlinear hybrid model. Thus for control, it is harder to use since the MPC optimization is non-convex.

2-4-3 Electrical circuit analogy

In modelling systems, an electrical circuit analogy is often employed, representing the system with resistors and capacitors, known as an resistance capacitance (RC) model. This model consists of a network of nodes interconnected by resistors and coupled to the ground through capacitors, as illustrated in Figure 2-4. Each node represents a spatial point in the aquifer where the temperature is considered a state variable. Specifically, the temperature at node 0 (T^0) corresponds to the temperature within the well, while the temperature at the final node reflects the ambient temperature ($\mathrm{T}^{\mathrm{amb}}$). During injection, the well temperature is considered to be the input, thus $\mathrm{T}^0 = \mathrm{T}^{\mathrm{in}}$. During extraction, the well temperature changes to the output $\mathrm{T}^0 = \mathrm{T}^{\mathrm{out}}$. During both operating modes, the flow rate V^{in} is also an input. For a 3-node RC model, the dynamics during the injection phase are described by the following equations [20]:

$$\begin{aligned}
\text{Node 1:} \quad & \tfrac{\mathrm{T}_k^0 - \mathrm{T}_k^1}{R_{10}} + \tfrac{\mathrm{T}_k^2 - \mathrm{T}_k^1}{R_{21}} + \rho c_p V_k^{\mathrm{in}} \left(\mathrm{T}_k^0 - \mathrm{T}_k^1\right) &&= C_1 \tfrac{\mathrm{T}_k^1 - \mathrm{T}_{k-1}^1}{\Delta\tau}, \\
\text{Node 2:} \quad & \tfrac{\mathrm{T}_k^1 - \mathrm{T}_k^2}{R_{21}} + \tfrac{\mathrm{T}_k^3 - \mathrm{T}_k^2}{R_{32}} + \rho c_p V_k^{\mathrm{in}} \left(\mathrm{T}_k^1 - \mathrm{T}_k^2\right) &&= C_2 \tfrac{\mathrm{T}_k^2 - \mathrm{T}_{k-1}^2}{\Delta\tau}, \\
\text{Node 3:} \quad & \tfrac{\mathrm{T}_k^{\mathrm{amb}} - \mathrm{T}_k^3}{R_{g3}} + \tfrac{\mathrm{T}_k^2 - \mathrm{T}_k^3}{R_{32}} + \rho c_p V_k^{\mathrm{in}} \left(\mathrm{T}_k^2 - \mathrm{T}_k^3\right) &&= C_3 \tfrac{\mathrm{T}_k^3 - \mathrm{T}_{k-1}^3}{\Delta\tau},
\end{aligned} \tag{2-13}$$

[1] This range is provided in [18], in [19] they elaborate on how to find this parameter. However, they find values for α slightly higher than 1. When fitting our ATES data in chapter 6, we find values for α up to 100. This could be because the sampling time has a great influence on this parameter; however, this is not clearly mentioned in either paper.

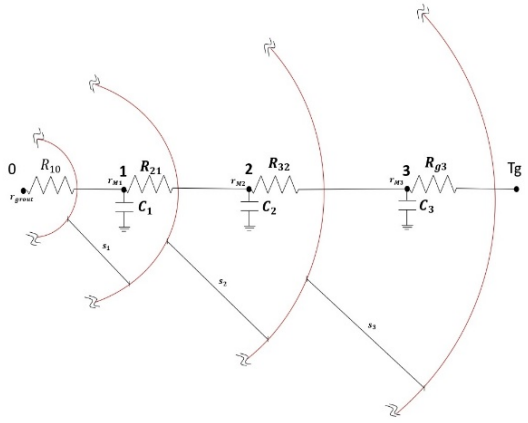

Figure 2-4: 3 node RC model [20].

During extraction, the dynamics become:

$$\begin{aligned}
\text{Node 0:} \quad & \frac{\mathrm{T}_k^1 - \mathrm{T}_k^0}{R_{10}} && + \rho c_p V_k^{\text{in}} \left(\mathrm{T}_k^1 - \mathrm{T}_k^0\right) && = 0 \\
\text{Node 1:} \quad & \frac{\mathrm{T}_k^0 - \mathrm{T}_k^1}{R_{10}} + \frac{\mathrm{T}_k^2 - \mathrm{T}_k^1}{R_{21}} && + \rho c_p V_k^{\text{in}} \left(\mathrm{T}_k^2 - \mathrm{T}_k^1\right) && = C_1 \frac{\mathrm{T}_k^1 - \mathrm{T}_{k-1}^1}{\Delta\tau} \\
\text{Node 2:} \quad & \frac{\mathrm{T}_k^1 - \mathrm{T}_k^2}{R_{21}} + \frac{\mathrm{T}_k^2 - \mathrm{T}_k^2}{R_{32}} && + \rho c_p V_k^{\text{in}} \left(\mathrm{T}_k^3 - \mathrm{T}_k^2\right) && = C_2 \frac{\mathrm{T}_k^2 - \mathrm{T}_{k-1}^2}{\Delta\tau} \\
\text{Node 3:} \quad & \frac{\mathrm{T}_k^{\text{amb}} - \mathrm{T}_k^3}{R_{g3}} + \frac{\mathrm{T}_k^2 - \mathrm{T}_k^3}{R_{32}} && + \rho c_p V_k^{\text{in}} (\mathrm{T}_k^{\text{amb}} - \mathrm{T}_k^3) && = C_3 \frac{\mathrm{T}_k^3 - \mathrm{T}_{k-1}^3}{\Delta\tau}
\end{aligned} \tag{2-14}$$

where

T_k^i is the temperature of node i at time step k (C),
$R_{i,i-1}$ is the resistance between node i and $i-1$, $R_{i,i-1} \in \mathbb{R}^+$ (Ω),
C_i is the capacitance of node i, $C_i \in \mathbb{R}^+$ (F),
ρ is the density of water ($\mathrm{kg\,m^{-3}}$), and
c_p is the specific heat capacity of water ($\mathrm{J\,kg^{-1}\,K^{-1}}$).

Due to the choice of backward Euler integration, this model requires solving a linear system of equations for each time step. For a dimension of 3, this is still relatively fast (especially if it is possible to compute an analytical inverse of $\mathrm{A}(V_{k+1}^{\text{in}})$). However, using this system in an MPC is very challenging since it is very non-convex to have to solve multiple consecutive systems of linear equations.

Chapter 3

Data driven modelling using the Koopman operator

Dynamic mode decomposition (DMD) is a data-driven method for analyzing and identifying linear system dynamics. It is able to extract dynamic information from high-dimensional systems (e.g. fluid flow). The extracted dynamic modes can be used to describe the underlying physical principals or to reduce the order of the system by only looking at the dominant modes [21, 22]. The method was first proposed by Schmid in 2010 [22]. The main goal of DMD is to find the linear transformation that moves the system discretely forward in time. Thus finding the A-matrix in $\mathrm{x}_{k+1} = \mathrm{A}\mathrm{x}_k$. Koopman operator theory provides the theoretical framework for using DMD on non-linear dynamics. This is explained in section 3-1. Next, the DMD algorithm is explained in section 3-2, and extensions to the algorithm are discussed in section 3-3.

3-1 Koopman operator theory

The main idea behind Koopman operator theory is that one can lift every nonlinear system to a system with linear dynamics, which might be of infinite dimension. For the analysis of linear dynamical systems, various methodologies and tools are developed, leading to more tractable approaches. Accordingly, it is of specific interest to employ those techniques for nonlinear dynamics.

Consider an autonomous discrete-time nonlinear dynamical system as

$$\mathrm{x}_{k+1} = \mathrm{f}(\mathrm{x}_k), \tag{3-1}$$

where k denotes the time step, $\mathrm{x}_k \in \mathbb{R}^n$ is the state vector at time instant k, and $f : \mathbb{R}^n \to \mathbb{R}^n$ is the vector field characterizing the dynamics. When the dynamics are linear, the vector field $\mathrm{f}(\cdot)$ is a linear map, and the system is as follows:

$$\mathrm{x}_{k+1} = \mathrm{A}\mathrm{x}_k \tag{3-2}$$

with $\mathrm{A} \in \mathbb{R}^{n \times n}$.

Let us define a scalar-valued function $g(\mathrm{x})$ that takes as input the full state vector of the system and returns a scalar $g : \mathbb{R}^n \to \mathbb{R}$. These functions are called *observables* (in literature, also called lifting or measurement functions). Using these observable functions, a vector function can be defined containing all observables of the system $\mathrm{g} : \mathbb{R}^n \to \mathbb{R}^{\overline{n}_\mathrm{g}}$ as

$$\mathrm{g}(\mathrm{x}) := \begin{bmatrix} g_1(\mathrm{x}) \\ g_2(\mathrm{x}) \\ \vdots \\ g_{\overline{n}_\mathrm{g}}(\mathrm{x}) \end{bmatrix} \tag{3-3}$$

where $\overline{n}_\mathrm{g} \in \{1, 2, \ldots\} \cup \{\infty\}$. A new lifted state vector is defined $\mathrm{z}_k \in \mathbb{R}^{\overline{n}_\mathrm{g}}$ as $\mathrm{z}_k = \mathrm{g}(\mathrm{x}_k)$. The Koopman operator is the linear map that moves the dynamics forward in time $\mathrm{z}_{k+1} = \mathcal{K}\mathrm{z}$. In this new coordinate system, the dynamics are linear. However, the dimension of the system can be infinite.

$$\begin{aligned} \mathrm{z}_{k+1} &= \mathrm{g}(\mathrm{x}_{k+1}) \\ &= \mathrm{g}(\mathrm{f}(\mathrm{x}_k)) \\ &= \mathcal{K}\mathrm{g}(\mathrm{x}_k) \\ &= \mathcal{K}\mathrm{z}_k \ . \end{aligned} \tag{3-4}$$

Roughly speaking, we have the commutative diagram below

$$\begin{array}{ccc} \mathrm{x}_k & \xrightarrow{\mathrm{f}(\mathrm{x}_k)\ \text{(non linear dynamics)}} & \mathrm{x}_{k+1} \\ \mathrm{g}(\mathrm{x}_k)\downarrow & & \uparrow \mathrm{g}^{-1}(\mathrm{z}_{k+1}) \\ \mathrm{z}_k & \xrightarrow{\mathcal{K}\mathrm{z}_k\ \text{(linear dynamics)}} & \mathrm{z}_{k+1} \end{array}$$

The challenge now is finding a truncated (finite dimension) approximation of the Koopman operator. DMD is an algorithm that does exactly that.

3-2 Dynamic mode decomposition

The data used with the DMD algorithm has to be structured in a specific way. Each measurement in time of the state vector x_k will be referred to as a snapshot. These snapshots are stored in two data matrices $\mathrm{X} \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{k}-1}$ and $\mathrm{X}^+ \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{k}-1}$, where n_x are the number of states and n_k the number of snapshots available. The second matrix X^+ contains the $1\Delta t$ time-shifted snapshot of the first matrix X, they are structured in the following way:

$$\mathrm{X} = \begin{bmatrix} | & | & & | \\ \mathrm{x}_0 & \mathrm{x}_1 & \cdots & \mathrm{x}_{n_\mathrm{k}-1} \\ | & | & & | \end{bmatrix}, \quad \text{and} \quad \mathrm{X}^+ = \begin{bmatrix} | & | & & | \\ \mathrm{x}_1 & \mathrm{x}_2 & \cdots & \mathrm{x}_{n_\mathrm{k}} \\ | & | & & | \end{bmatrix}. \tag{3-5}$$

It is not necessary for the snapshots in the data matrix to be consecutive snapshots as long as, in the time-shifted data matrix, the corresponding consecutive time step is at the right place. It is thus possible to use multiple experiments in the training data.

Rewriting the linear dynamics of the system $x_{k+1} = Ax_k$ to account for the full data matrices becomes $X^+ = AX$. Finding the best predictor $\hat{A}$ for the linear map A comes down to solving the following optimization problem:

$$\hat{A} = \underset{A}{\operatorname{argmin}} \left\| X^+ - AX \right\|_{\mathcal{F}}. \tag{3-6}$$

An analytical solution to find $\hat{A}$ which is very similar to least squares is:

$$\hat{A} = X^+ X^\dagger, \tag{3-7}$$

where † indicates the Moore-Penrose pseudoinverse ($X^\dagger = X^*(XX^*)^{-1}$). Computing the pseudoinverse can be computationally very expensive if n_x becomes large. A more computationally efficient way is to use a (reduced) singular value decomposition (SVD). The SVD is as

$$X = U\Sigma V^*, \tag{3-8}$$

where $X \in \mathbb{C}^{n\times m}$ is any real or complex matrix, $U \in \mathbb{C}^{n\times n}$ and $V \in \mathbb{C}^{m\times m}$ are unitary matrices (their inverse is their transpose $U^* = U^{-1}$) and $\Sigma \in \mathbb{R}^{n\times m}$ a diagonal matrix containing the singular values [23]. In the rest of this thesis, the matrices on which the SVD is performed are assumed to be real ($X \in \mathbb{R}^{n\times m}$), making the conjugate transpose (*) equivalent to the normal transpose ($^\mathsf{T}$). A parameter of the DMD algorithm is the rank r of the predictor $\hat{A}$. This can be added as a constraint to the optimization problem in (3-7)

$$\begin{aligned} \hat{A} = &\underset{A}{\operatorname{argmin}} \left\| X^+ - AX \right\|_{\mathcal{F}} \\ &\text{s.t. } \operatorname{rank}(A) \leq r. \end{aligned} \tag{3-9}$$

The *Eckart-Young-Mirsky theorem* states that the best low-rank approximation of a matrix is the truncated SVD. More precisely, the best approximation of a matrix A given that $\operatorname{rank}(A) \leq r$ is $A \approx U_r\Sigma_r V_r^\mathsf{T}$. Where $U_r = U_{(\bullet,1:r)}$, $V_r = V_{(\bullet,1:r)}$ and $\Sigma_r = U_{(1:r,1:r)}$. Inserting the truncated SVD of $X \approx U_r\Sigma_r V_r^\mathsf{T}$ into (3-7) gives

$$\hat{A} = X^+(U_r\Sigma_r V_r^\mathsf{T})^\dagger = X^+ V_r \Sigma_r^{-1} U_r^\mathsf{T}. \tag{3-10}$$

It is possible to compute a reduced-order model of $\hat{A}$: $\tilde{A} \in \mathbb{R}^{r\times r}$ with r the rank and size of $\tilde{A}$. $\tilde{A}$ is computed by projecting onto the DMD modes U_r:

$$\tilde{A} = U_r^\mathsf{T}\hat{A}U_r = U_r^\mathsf{T} X^+ V_r \Sigma_r^{-1}, \tag{3-11}$$

The coordinate system of the reduced-order model is $\tilde{x}_k = U_r x_k$. And the dynamics are simply described by $\tilde{x}_{k+1} = \tilde{A}\tilde{x}_k$. The full-state vector is retrieved from the reduced order model with $x_k = U_r^\mathsf{T}\tilde{x}_k$. The advantage of a reduced order model is that it can capture the essence of high dimensional data, is less sensitive to overfitting and is computationally less expensive because the full SVD never has to be computed or the full matrix $\hat{A}$ has never to be stored in memory. Also, analysis based on the eigenvalues is much cheaper since the computation is now done on a $r \times r$ instead of a $n_x \times n_x$ matrix with $r \ll n_x$.

3-3 Extensions to DMD

The standard DMD algorithm is only able to model unforced systems that can be approximated with linear dynamics. This limits the use of DMD, especially in the field of control engineering. To increase the usefulness of DMD, extensions to DMD are made to be able to model controlled systems, this is elaborated upon in subsection 3-3-1. To better approximate complex nonlinear dynamics, the use of lifting functions or observables is necessary, this is discussed in subsection 3-3-2. To further increase the abilities of DMD, it is possible to include physical laws as a constraint into the algorithm. This is explained in subsection 3-3-3.

3-3-1 Dynamic mode decomposition with control

With DMDc control can be incorporated into the dynamics: $\mathrm{x}_{k+1} = \mathrm{A}\mathrm{x}_k + \mathrm{B}\mathrm{u}_k$, with n_x states and n_u control inputs [21]. Similar to DMD data matrices are needed to learn the model. Besides the two data matrices for the states from (3-5) an additional data matrix with the control inputs is needed:

$$\Upsilon = \begin{bmatrix} | & | & & | \\ \mathrm{u}_0 & \mathrm{u}_1 & \cdots & \mathrm{u}_{n_\mathrm{k}-1} \\ | & | & & | \end{bmatrix}, \tag{3-12}$$

where $\Upsilon \in \mathbb{R}^{n_\mathrm{u} \times n_\mathrm{k}-1}$. Now the controlled dynamics can be written in terms of its data matrices:

$$\mathrm{X}^+ = \mathrm{A}\mathrm{X} + \mathrm{B}\Upsilon = \begin{bmatrix} \mathrm{A} & \mathrm{B} \end{bmatrix} \begin{bmatrix} \mathrm{X} \\ \Upsilon \end{bmatrix} = \mathrm{G}\Omega. \tag{3-13}$$

Now the same method as with DMD as in (3-7) can be used to solve for $\hat{\mathrm{G}}$ with the SVD of $\Omega = \mathrm{U}\Sigma\mathrm{V}^\mathsf{T}$ the solution for $\hat{\mathrm{G}}$ becomes:

$$\hat{\mathrm{G}} = \mathrm{X}^+\mathrm{V}\Sigma^{-1}\mathrm{U}^\mathsf{T}. \tag{3-14}$$

Extracting $\hat{\mathrm{A}}$ and $\hat{\mathrm{B}}$ from $\hat{\mathrm{G}}$ results in:

$$\begin{bmatrix} \hat{\mathrm{A}} & \hat{\mathrm{B}} \end{bmatrix} = \begin{bmatrix} \mathrm{X}^+\mathrm{V}\Sigma^{-1}\mathrm{U}_1^\mathsf{T} & \mathrm{X}^+\mathrm{V}\Sigma^{-1}\mathrm{U}_2^\mathsf{T} \end{bmatrix}, \tag{3-15}$$

where $\mathrm{U}_1 \in \mathbb{R}^{n_\mathrm{x} \times p}$ and $\mathrm{U}_2 \in \mathbb{R}^{n_\mathrm{u} \times p}$ and p denotes the truncation factor of the SVD of Ω. Again, similar to normal DMD, if n_x and n_u are large, this model is computationally expensive. Thus, a reduced order model of rank r is sought with projected coordinates (equivalent to the projection onto modes U with normal DMD) of the form $\mathrm{x}_k = \mathrm{P}\tilde{\mathrm{x}}_k$, with $\tilde{\mathrm{x}}_k \in \mathbb{R}^r$. This is a reduced order model of the *output* space. To find this reduced order model a second SVD is required, the first of Ω with truncation factor p, that we already used. The second SVD is on X^+. For clarity of the notation, an under-script indicates which matrix the SVD is computed on.

$$\begin{aligned} \Omega &\approx \mathrm{U}_\Omega\Sigma_\Omega\mathrm{V}_\Omega^\mathsf{T} = \begin{bmatrix} \mathrm{U}_{\Omega_1} \\ \mathrm{U}_{\Omega_2} \end{bmatrix} \Sigma_\Omega\mathrm{V}_\Omega^\mathsf{T} \\ \mathrm{X}^+ &\approx \mathrm{U}_{\mathrm{X}^+}\Sigma_{\mathrm{X}^+}\mathrm{V}_{\mathrm{X}^+}^\mathsf{T} \end{aligned} \tag{3-16}$$

The truncation factors of the SVDs must be different with $p > r$.

Using the transformation $\mathrm{x}_k = \mathrm{U}_{\mathrm{X}^+}\tilde{\mathrm{x}}_k$ the reduced approximation can be computed:

$$\begin{aligned} \tilde{\mathrm{A}} &= \mathrm{U}_{\mathrm{X}^+}^{\mathsf{T}}\hat{\mathrm{A}}\mathrm{U}_{\mathrm{X}^+} &&= \mathrm{U}_{\mathrm{X}^+}^{\mathsf{T}}\mathrm{X}^+\mathrm{V}_\Omega{\Sigma_\Omega}^{-1}\mathrm{U}_{\Omega_1}^{\mathsf{T}}\mathrm{U}_{\mathrm{X}^+} \\ \tilde{\mathrm{B}} &= \mathrm{U}_{\mathrm{X}^+}^{\mathsf{T}}\hat{\mathrm{B}} &&= \mathrm{U}_{\mathrm{X}^+}^{\mathsf{T}}\mathrm{X}^+\mathrm{V}_\Omega{\Sigma_\Omega}^{-1}\mathrm{U}_{\Omega_2}^{\mathsf{T}} \end{aligned} \tag{3-17}$$

where $\tilde{\mathrm{A}} \in \mathbb{R}^{r\times r}$ and $\tilde{\mathrm{B}} \in \mathbb{R}^{r\times n_\mathrm{u}}$.

The reduced order model now becomes:

$$\tilde{\mathrm{x}}_{k+1} = \tilde{\mathrm{A}}\tilde{\mathrm{x}}_k + \tilde{\mathrm{B}}\mathrm{u}_k. \tag{3-18}$$

3-3-2 Extended dynamic mode decomposition

The standard DMD algorithms are mostly useful in reducing the the dimensionality of a system or to learn linear system dynamics. However, DMD can also be used the other way around to identify a linear operator from nonlinear measurements, thus identifying the Koopman operator. Extended DMD (EDMD) attempts to find the Koopman operator by extending the state vector with observables (also called lifting or augmenting the states). Then the normal DMD algorithm is applied to this extended state $\mathrm{z}_k = \mathrm{g}(\mathrm{x}_k)$ as shown in (3-3). To retrieve the original state vector, an inverse of the lifting function $\mathrm{g}^{-1}(\cdot)$ must exist. A straightforward method to achieve this is by adding the lifting functions to the original state vector, making the first n_x elements of z_k equal to x_k. Thus $\mathrm{g}^{-1}(\mathrm{z}_k) = \mathrm{z}_{k,(1:n_\mathrm{x})} = \mathrm{x}_k$. The lifted data matrices from (3-5) become

$$\mathrm{Z} = \begin{bmatrix} | & | & & | \\ \mathrm{g}(\mathrm{x}_1) & \mathrm{g}(\mathrm{x}_2) & \cdots & \mathrm{g}(\mathrm{x}_{n_\mathrm{k}-1}) \\ | & | & & | \end{bmatrix}, \quad \text{and} \quad \mathrm{Z}^+ = \begin{bmatrix} | & | & & | \\ \mathrm{g}(\mathrm{x}_2) & \mathrm{g}(\mathrm{x}_3) & \cdots & \mathrm{g}(\mathrm{x}_{n_\mathrm{k}}) \\ | & | & & | \end{bmatrix}. \tag{3-19}$$

The choice of lifting function is a big factor in being able to correctly identify the dynamics of a system. However, there is no clear path or rule for choosing the correct lifting functions and it is very problem-dependent which lifting function will perform well.

A very common lifting function in the field of system identification is Hankel delay lifting. The Hankel matrix has a specific structure where each row contains a time-shifted version of the previous row with overlap. Here Hankel DMD (HDMD) is classified as a version of EDMD since the state is lifted with time delay versions of itself. The lifted state vector z_k contains n_d delayed snapshots stacked on top of each other. The structure of the data matrix for HDMD is as follows:

$$\mathrm{Z} = \begin{bmatrix} \mathrm{x}_0 & \mathrm{x}_1 & \cdots & \mathrm{x}_{n_\mathrm{k}-n_\mathrm{d}} \\ \mathrm{x}_1 & \mathrm{x}_2 & \cdots & \mathrm{x}_{n_\mathrm{k}-n_\mathrm{d}+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{x}_{n_\mathrm{d}} & \mathrm{x}_{n_\mathrm{d}+1} & \cdots & \mathrm{x}_{n_\mathrm{k}-1} \end{bmatrix}, \text{ and } \mathrm{Z}^+ = \begin{bmatrix} \mathrm{x}_1 & \mathrm{x}_2 & \cdots & \mathrm{x}_{n_\mathrm{k}-n_\mathrm{d}+1} \\ \mathrm{x}_2 & \mathrm{x}_3 & \cdots & \mathrm{x}_{n_\mathrm{k}-n_\mathrm{d}+2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{x}_{n_\mathrm{d}+1} & \mathrm{x}_{n_\mathrm{d}+2} & \cdots & \mathrm{x}_{n_\mathrm{k}} \end{bmatrix}, \tag{3-20}$$

where n_d is the number of time-delayed vectors in the extended state and n_k (as with normal DMD) the number of temporal measurements (snapshots) available. x_k indicates the full state vector at time-step k [24].

3-3-3 Physics-informed dynamic mode decomposition

DMD models often break the laws of physics. If these laws can be taken into account it is possible to make better and more noise-robust models with less data. By enforcing a structure on the matrix A, called a matrix manifold $\mathcal{M}$, the physics can be embedded in the solution [25].

$$\hat{\mathrm{A}} = \underset{\mathrm{A}\in\mathcal{M}}{\operatorname{argmin}} \left\| \mathrm{X}^{+} - \mathrm{AX} \right\|_{\mathcal{F}} \tag{3-21}$$

In [25], five possible matrix manifolds corresponding to physical laws are described. Here, only one is discussed since it relates most to the application of this thesis: ATES.

Local DMD

This physical constraint is locality. This means that states are only influenced by neighbouring states. Take, for example, heat transfer in a rod. The instantaneous change in temperature at any point along the rod only changes based on the temperature of the points right next to the point of interest. A point on the other end of the rod does not influence the state directly. This can be translated into a matrix manifold by enforcing a tri-diagonal structure. A tri-diagonal matrix is a matrix that is zero everywhere except on the diagonal and the one-off diagonals:

$$\mathrm{A} = \begin{bmatrix} \beta_1 & \gamma_1 & & & \\ \alpha_2 & \beta_2 & \gamma_2 & & \\ & \alpha_3 & \ddots & \ddots & \\ & & \ddots & \ddots & \gamma_{n-1} \\ & & & \alpha_n & \beta_n \end{bmatrix}. \tag{3-22}$$

The rows of A are now decoupled, meaning that the coefficients of that row only depend on the data from that row and one row up and down. This makes it possible to divide the optimization problem into n_{x} smaller problems. The optimization problem in (3-21) with the tri-diagonal matrix manifold reduces to: [1]

$$\underset{\alpha_i,\beta_i,\gamma_i}{\operatorname{argmin}} \left\| \alpha_i \mathrm{X}_{(i-1,\bullet)} + \beta_i \mathrm{X}_{(i,\bullet)} + \gamma_i \mathrm{X}_{(i+1,\bullet)} - \mathrm{X}^{+}_{(i,\bullet)} \right\|_2 \quad \text{for } 2 \leq i \leq n_{\mathrm{x}} - 1. \tag{3-23}$$

For $i = 1$ and $i = n_{\mathrm{x}}$, the optimization problem changes slightly, the out-of-bounds indices ($\mathrm{X}_{(0,\bullet)}$ and $\mathrm{X}_{(n_{\mathrm{x}}+1,\bullet)}$) and corresponding coefficients (α_1 and $\gamma_{n_{\mathrm{x}}}$) need to be removed. The analytical solution to this optimization problem is

$$\begin{bmatrix} \alpha_i & \beta_i & \gamma_i \end{bmatrix} = \mathrm{X}^{+}_{(i,\bullet)} \begin{bmatrix} \mathrm{X}_{(i-1,\bullet)} \\ \mathrm{X}_{(i,\bullet)} \\ \mathrm{X}_{(i+1,\bullet)} \end{bmatrix}^{\dagger} = \mathrm{X}^{+}_{(i,\bullet)} (\mathrm{X}_{(i-1:i+1,\bullet)})^{\dagger} \quad \text{for } 2 \leq i \leq n_{\mathrm{x}} - 1. \tag{3-24}$$

This solution is similar to the DMD regression problem in (3-7). However, in contrast to the normal DMD algorithm it is not possible to compute a low rank approximation with this method.

[1] $\mathrm{X}_{(i,\bullet)}$ denotes a row vector containing the i-th row of matrix X

This tri-diagonal matrix manifold is especially useful in continuous time. In discrete-time systems, this constraint might be too stringent. When working in discrete-time the locality also depends on the sampling time. Take the heat transfer in the rod again, the instantaneous change is only influenced by the neighbouring states. In discrete-time, however, heat can spread to further away states if the sampling time is large. Therefore the manifold in (3-22) can be extended to include more than 1 off-diagonal to incorporate this weaker spatial locality. The solution in (3-23) can be generalized to take more off-diagonal terms into account, taking n_r states to the right and n_l states to the left off the diagonal

$$\begin{bmatrix} \mathrm{A}_{(i-n_\mathrm{l},i)} & \cdots & \mathrm{A}_{(i+n_\mathrm{r},i)} \end{bmatrix} = \mathrm{X}^{+}_{(i,\bullet)} \begin{bmatrix} \mathrm{X}_{(i-n_\mathrm{l},\bullet)} \\ \vdots \\ \mathrm{X}_{(i+n_\mathrm{r},\bullet)} \end{bmatrix}^{\dagger} \quad \text{for } n_\mathrm{l} \leq i \leq n_\mathrm{x} - n_\mathrm{r}. \tag{3-25}$$

Extending this result to include control is also trivial. Then the decoupled optimization problem in (3-23) for multiple off-diagonals, again taking n_r states to the right and n_l states to the left of the diagonal and with n_u control inputs, the analytical solution generalizes to:

$$\begin{bmatrix} \mathrm{A}_{(i-n_\mathrm{l},i)} & \cdots & \mathrm{A}_{(i+n_\mathrm{r},i)} & \mathrm{B}_{(1,i)} & \cdots & \mathrm{B}_{(n_\mathrm{u},i)} \end{bmatrix} = \mathrm{X}^{+}_{(i,\bullet)} \begin{bmatrix} \mathrm{X}_{(i-n_\mathrm{l},\bullet)} \\ \vdots \\ \mathrm{x}_i \\ \vdots \\ \mathrm{X}_{(i+n_\mathrm{r},\bullet)} \\ \Upsilon \end{bmatrix}^{\dagger} \quad \text{for } n_\mathrm{l} \leq i \leq n_\mathrm{x} - n_\mathrm{r}, \tag{3-26}$$

where Υ is the control input data matrix as defined in (3-12). If the control also has local properties and thus does not work on all states, this can easily be taken into account by switching per row between the controlled solution (3-26) or the uncontrolled solution (3-25) since every row of the matrix is solved separately. In this thesis, this algorithm will be referred to as the piDMD algorithm.

Stable DMD

Maybe the most important property of a (linear) dynamical system is whether it is stable or not. If this could be embedded in the learning algorithm, it would result in better, more noise-robust models. However, learning a stable system (in discrete or continuous time) often relies on complicated constraint or non-convex optimization [26, 27]. To simplify this process [28] proposes a different solution, adding a convex regularization term to the optimization problem that indirectly punishes instability. Here, discrete-time stability is considered, more precisely, the magnitude of the largest eigenvalue (the spectral radius) is smaller than one $\rho(A) < 1$.

The Frobenius norm ($\|\cdot\|_\mathcal{F}$) is related to stability but not directly. If $\|\mathrm{A}\|_\mathcal{F} < 1$, then A is stable, but not the other way around, it is a sufficient but not a necessary condition. There exist matrices which have eigenvalues smaller than one but do have a Frobenius norm larger than one.

$$\|\mathrm{A}\|_\mathcal{F} < 1 \implies \rho(\mathrm{A}) < 1 \tag{3-27}$$

Although it is not possible to say more about the stability from the Frobenius norm with certainty, it does show a clear (almost linear) trend that if the Frobenius norm of a matrix is lower, then the spectral radius is lower as well. It is very matrix and size-dependent on how clear this trend is, although for larger matrices it appears to be clearer, see Appendix A. Using this relation, the Frobenius norm can be leveraged to penalize the stability of a matrix. The DMD optimization problem can be augmented to [28]

$$\begin{aligned} \hat{\mathrm{A}} = \underset{\mathrm{A}}{\operatorname{argmin}} \left\| \mathrm{X}^{+} - \mathrm{AX} \right\|_{\mathcal{F}}^{2} + \lambda \left\| \mathrm{A} \right\|_{\mathcal{F}}^{2} \\ \text{s.t. } \operatorname{rank}(\mathrm{A}) \leq r, \end{aligned} \tag{3-28}$$

with $\lambda \in \mathbb{R}_{+}$ This optimization problem can be rewritten to be in standard DMD form

$$\begin{aligned} \hat{\mathrm{A}} = \underset{\mathrm{A}}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathrm{X}^{+} & \mathbf{0}_{n_x} \end{bmatrix} - \mathrm{A} \begin{bmatrix} \mathrm{X}^{+} & \sqrt{\lambda}\mathbf{I}_{n_x} \end{bmatrix} \right\|_{\mathcal{F}}^{2} \\ \text{s.t. } \operatorname{rank}(\mathrm{A}) \leq r \end{aligned} \tag{3-29}$$

Which has an analytical solution equivalent to DMD, using the (truncated) SVD :

$$\hat{\mathrm{A}} = \mathrm{X}^{+}\mathrm{V}_r\Sigma_r^{-1}\mathrm{U}_r^{\mathsf{T}} \text{ with } \mathrm{U}_r\Sigma_r\mathrm{V}_r^{\mathsf{T}} \approx \begin{bmatrix} \mathrm{X}^{+} & \sqrt{\lambda}\mathbf{I}_{n_x} \end{bmatrix}. \tag{3-30}$$

The regularization parameter λ that results in a stable system can be found using the bi-section algorithm. Since we know that when the Frobenius norm becomes lower, the spectral radius decreases as well. Thus finding the root of $\rho(\mathrm{A}) - 1$ (or minimizing $|\rho(\mathrm{A}) - 1|$) will result in a critically stable system. We now have to solve:

$$\min_{\lambda} \left| \rho\left(\mathrm{X}^{+}\mathrm{V}_r\Sigma_r^{-1}\mathrm{U}_r^{\mathsf{T}}\right) - 1 \right| \text{ with } \mathrm{U}_r\Sigma_r\mathrm{V}_r^{\mathsf{T}} \approx \begin{bmatrix} \mathrm{X}^{+} & \sqrt{\lambda}\mathbf{I}_{n_x} \end{bmatrix}. \tag{3-31}$$

This solution is extended to include control. Updating equation (3-29) and (3-30) to include control results in:

$$\begin{aligned} \begin{bmatrix} \hat{\mathrm{A}} & \hat{\mathrm{B}} \end{bmatrix} = \underset{\mathrm{A,B}}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathrm{X}^{+} & \mathbf{0}_{n_x} \end{bmatrix} - \begin{bmatrix} \mathrm{A} & \mathrm{B} \end{bmatrix} \begin{bmatrix} \mathrm{X}^{+} & \sqrt{\lambda}\mathbf{I}_{n_x} \\ \Upsilon & \mathbf{0}_{n_u \times n_x} \end{bmatrix} \right\|_{\mathcal{F}}^{2} \\ \text{s.t. } \operatorname{rank}(\mathrm{A}) \leq r \end{aligned} \tag{3-32}$$

$$\begin{bmatrix} \hat{\mathrm{A}} & \hat{\mathrm{B}} \end{bmatrix} = \begin{bmatrix} \mathrm{X}^{+} & \mathbf{0}_{n_x} \end{bmatrix} \mathrm{V}_r\Sigma_r^{-1}\mathrm{U}_r^{\mathsf{T}} \text{ with } \mathrm{U}_r\Sigma_r\mathrm{V}_r^{\mathsf{T}} \approx \begin{bmatrix} \mathrm{X}^{+} & \sqrt{\lambda}\mathbf{I}_{n_x} \\ \Upsilon & \mathbf{0}_{n_u \times n_x} \end{bmatrix} \tag{3-33}$$

Combining stability and locality

Is it now possible to combine the stability and tri-diagonal constraints? For this to work, a stability criterion must exist that can indicate stability per row. Otherwise, the fast solution of computing decoupled rows from the piDMD algorithm is lost. This criterion does not exist directly, however, similar to the Frobenius norm, there is a sufficient condition based on the independent rows of a matrix: the *Gershgorin circle theorem* states that the eigenvalues of a matrix lie inside the outer edge of the union of the disks with the centre of the disk defined as the diagonal element of the matrix ($\mathrm{A}_{(i,i)}$) and the radius the sum of the absolute value of the off-diagonal elements of that row.

Theorem 1. Gershgorin circle theorem. *Every eigenvalue* $\lambda \in \mathbb{C}$ *of a square matrix* $\mathrm{A} \in \mathbb{C}^{n \times n}$ *lies in at least one of the Gershgorin disks* D_i

$$D_i = \left\{ z \in \mathbb{C} : \left| z - \mathrm{A}_{(i,i)} \right| \leq r_i \right\}, \quad \textit{with } r_i = \sum_{\substack{j=1, \\ j \neq i}}^{n} \left| \mathrm{A}_{(i,j)} \right| \quad \textit{for } 1 \leq i \leq n \tag{3-34}$$

This means that if the sum of the absolute values for all elements of the row of a square matrix is smaller than one, then the matrix is discrete-time stable. Let us define the matrix Gershgorin norm $\|\cdot\|_{\mathcal{G}}$ as

$$\|\mathrm{A}\|_{\mathcal{G}} = \max_{1 \leq i \leq n} \sum_{j=1}^{n} \left| \mathrm{A}_{(i,j)} \right|, \tag{3-35}$$

then

$$\|\mathrm{A}\|_{\mathcal{G}} < 1 \implies \rho(\mathrm{A}) < 1. \tag{3-36}$$

Similar to the Frobenius norm stability condition, this is a sufficient but not a necessary condition. If we also assume that a lower maximum Gerhsgorin disk outer edge corresponds to a lower maximum eigenvalue (see Appendix A), we can relax the stability constraint by saying that the maximum disc radius is some value λ instead of 1. This means that (3-36) becomes:

$$\|\mathrm{A}\|_{\mathcal{G}} < \lambda \implies \rho(\mathrm{A}) < 1. \tag{3-37}$$

We now have a linear constraint for stability that is row-independent. Formulating this as a linear constraint optimization problem, for a tri-diagonal matrix with one control input results in:

$$\begin{aligned} \underset{\alpha_i, \beta_i, \gamma_i, \delta_i}{\operatorname{argmin}} & \left\| \alpha_i \mathrm{X}_{(i-1,\bullet)} + \beta_i \mathrm{X}_{(i,\bullet)} + \gamma_i \mathrm{X}_{(i+1,\bullet)} - \mathrm{X}^{+}_{(i,\bullet)} + \delta_i u_{(i)} \right\|_2 \\ \text{s.t.} & \ |\alpha_i| + |\beta_i| + |\gamma_i| < \lambda. \end{aligned} \tag{3-38}$$

Note that the constraints containing absolute values are not actually linear. However, it is trivial to linearize them as is shown in Appendix B. We call this algorithm the Gershgorin DMD (GeDMD) algorithm. Due to the constraints, no analytical solution exists. The parameter λ can now be tuned similarly to the Frobenius norm stability in (3-33), using the bi-section algorithm that finds a critically stable system. For a new λ in an iteration of the bisection algorithm, only the rows that violate the constraint have to be recomputed.

It is possible to extend these results to also handle a diagonal block matrix structure. This is especially helpful when it is combined with EDMD, where lifted states can again only be local with each other.

Chapter 4

DMD for data-driven modelling of ATES

First, the required data for training is described in section 4-1. Then, the methods for lifting the data are discussed on section 4-2. The expected model structure from the lifting methods is discussed in section 4-3. A summary of the learning process is provided in section 4-4. Then, some model properties are discussed in section 4-5. The chapter concludes by explaining the hyperparameter optimization strategy used to tune the model parameters.

Given the similarity in the dynamics of hot and cold wells, this chapter focuses on a single well, the hot well. Thus, the hot injection mode will be referred to as *injection*, and the cold injection mode as *extraction*.

4-1 Training data excitation

Training data that covers the full frequency spectrum is required to identify a linear system. Although a linear system is learned, the underlying dynamics are non-linear. Therefore, the frequency spectrum and the normal working range need to be covered.

The four possible inputs are the same as of the actual ATES system:

- V_{in}^{H} the flow rate (or volume) into the hot well,
- $\text{T}_{\text{inj}}^{\text{H}}$ the temperature of the water entering the well,
- V_{in}^{C} the flow rate (or volume) into the hot well, and
- $\text{T}_{\text{inj}}^{\text{C}}$ the temperature of the water entering the hot well.

As described in section 2-1, the flow into and out of the two wells is coupled. So, only one flow is considered ($V^{\text{in}} = V_{\text{in}}^{\text{H}} = -V_{\text{in}}^{\text{C}}$).

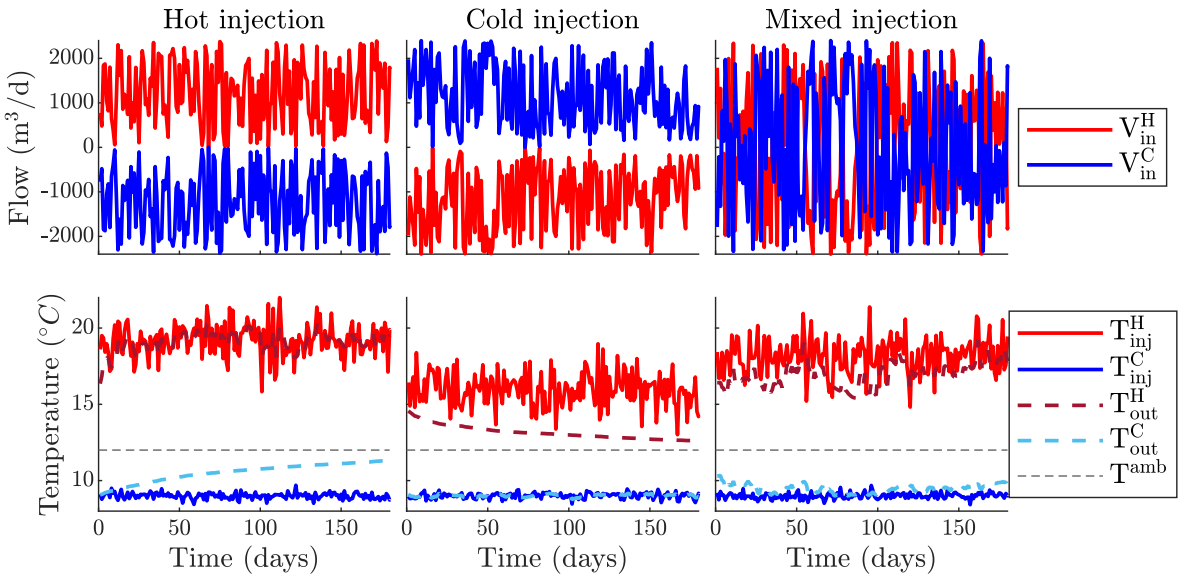

Figure 4-1: Input and output of training data for each mode separate. Only one experiment per node is shown, for training multiple experiments are used.

Uniformly distributed noise is used as an input for the flow rate to cover a broad range of operating conditions. For the flow rate, three operating modes are considered:

- only hot injection ($V^{\text{in}} > 0$),
- only cold injection ($V^{\text{in}} < 0$) and
- mixed injection, where both modes are randomly used.

For the injection temperatures, noise is added to their expected value. For the cold injection temperature this is $T^C_{inj} = 9\,^\circ\text{C}$ and for hot injection there is a wider range T^H_{inj} =15 to 20 °C. In Figure 4-1, the flow rates and temperatures are shown for the three operating modes.

Multiple experiments with different initial conditions are done. First, a simulation is made with normal operating conditions for 10 years, then the system is in an equilibrium operating mode. Every month of this final year has been used as the initial condition for the simulations for all three operating modes.

4-2 Methods for lifting the data

For now, we are focusing on one well: the hot well. This is done to keep the dimensions slightly smaller and have a clearer view of what is happening. The dynamics are similar for the cold well.

4-2-1 Spatial lifting

For now, we do not consider the full 3D space, but only a line of measurements. The line is chosen to run through both wells in the middle of the screen. On this line, the temperature

head and flow data are available at each grid point. So, at each grid point p, we have a snapshot

$$\mathrm{y}_k^p = \begin{bmatrix} \mathrm{T}_k^p \\ h_k^p \\ {q_x}_k^p \\ {q_y}_k^p \\ {q_z}_k^p \end{bmatrix}. \tag{4-1}$$

The number of grid points is chosen, moving away from the well. So $p = 0$ indicates the grid cell where the well is placed, $p = 1$ and $p = -1$ indicate the grid cell 10 m to the right and left of the well, respectively. n_{sc} indicates the maximum value of p thus how many locations to one side are considered. Then, the total number of spatial coordinates considered is $2n_{\mathrm{sc}} + 1$. In (4-1), all available MODFLOW data is shown. However, this is not necessary for the algorithm. The amount of data types considered is n_{data}. If for example, only temperature and the head are chosen then $n_{\mathrm{data}} = 2$, if only temperature is considered $n_{\mathrm{data}} = 1$. Non-required data can be left out of this vector. In chapter 6, the data chosen is shown. In most cases, it is only the temperature reducing y_k^p to $\mathrm{y}_k^p = \mathrm{T}_k^p$. Now, the full snapshot vector, when considering n_{sc} grid points, becomes

$$\mathrm{x}_k = \begin{bmatrix} \mathrm{y}_k^{-n_{\mathrm{sc}}} \\ \mathrm{y}_k^{-n_{\mathrm{sc}}+1} \\ \vdots \\ \mathrm{y}_k^{n_{\mathrm{sc}}-1} \\ \mathrm{y}_k^{n_{\mathrm{sc}}} \end{bmatrix}, \tag{4-2}$$

where $\mathrm{x}_k \in \mathbb{R}^{n_{\mathrm{x}}}$. The dimension of the snapshot vector is

$$n_{\mathrm{x}} = (2n_{\mathrm{sc}} + 1)n_{\mathrm{data}}. \tag{4-3}$$

4-2-2 Hankel lifting

The process of Hankel DMD is already explained in subsection 3-3-2. For completeness, the full lifted state vector, including spatial and Hankel lifting, is shown here

$$\mathrm{z}_k = \begin{bmatrix} \mathrm{x}_k \\ \mathrm{x}_{k-1} \\ \vdots \\ \mathrm{x}_{k-n_{\mathrm{d}}} \end{bmatrix}, \tag{4-4}$$

where n_{d} is the number of delay coordinates considered, $\mathrm{z}_k \in \mathbb{R}^{n_{\mathrm{z}}}$ and x is the spatial lifted state vector from (4-4). The dimension of the lifted vector and thus of the system can be computed as

$$n_{\mathrm{z}} = n_{\mathrm{x}}(n_{\mathrm{d}} + 1) = (2n_{\mathrm{sc}} + 1)n_{\mathrm{data}}(n_{\mathrm{d}} + 1). \tag{4-5}$$

4-2-3 Hybrid system

Since the dynamics are very dependent on the flow, they change drastically if the flow is reversed. Therefore, a hybrid model could provide better results. Making the model hybrid does move away from the ideal of having a linear model for ATES. However, with ATES the modes are very predictable. Hot injection only occurs in summer, and cold injection only in winter, with a storage period in between. The moment of switching between these modes will differ yearly depending on the weather.

The state vector x is the same for the models. For the injection mode, we have the dynamics $\mathrm{A}^{\mathrm{inj}}$ and $\mathrm{B}^{\mathrm{inj}}$ and input $\mathrm{u}_k^{\mathrm{inj}}$ and similar for extraction $\mathrm{A}^{\mathrm{ext}}$ and $\mathrm{B}^{\mathrm{ext}}$ and input $\mathrm{u}_k^{\mathrm{ext}}$. The input vector is different for each operating mode. For hot injection

$$\mathrm{u}_k^{\mathrm{inj}} = \begin{bmatrix} V^{\mathrm{in}} \\ \mathrm{T}^{\mathrm{in}} \\ V^{\mathrm{in}}\mathrm{T}^{\mathrm{in}} \end{bmatrix}, \tag{4-6}$$

and for extraction

$$\mathrm{u}_k^{\mathrm{ext}} = V^{\mathrm{in}}. \tag{4-7}$$

The hybrid dynamics can now be written as

$$\mathrm{z}_{k+1} = \begin{cases} \mathrm{A}^{\mathrm{inj}}\mathrm{z}_k + \mathrm{B}^{\mathrm{inj}}\mathrm{u}_k^{\mathrm{inj}} & \text{for } V_k^{\mathrm{in}} > 0 \\ \mathrm{z}_k & \text{for } V_k^{\mathrm{in}} = 0 \\ \mathrm{A}^{\mathrm{ext}}\mathrm{z}_k + \mathrm{B}^{\mathrm{ext}}\mathrm{u}_k^{\mathrm{ext}} & \text{for } V_k^{\mathrm{in}} < 0 \end{cases} \tag{4-8}$$

4-2-4 Subsampling

The DMD algorithm optimizes the one-step-ahead prediction. For longer timescale predictions, this is not always ideal. By raising the A matrix to a power the model's errors accumulate. Therefore a subsampled model is learned, denoted with ${}^{(n_{\mathrm{sub}})}\mathrm{A}$, ${}^{(n_{\mathrm{sub}})}\mathrm{B}$ where n_{sub} is the number of subsampled time steps. If the system is linear and the dynamics are learned perfectly $\mathrm{A}^k = {}^{(k)}\mathrm{A}$. Now, instead of computing the future states with only the one-step-ahead prediction model as

$$\mathrm{z}_k = \mathrm{A}^k\mathrm{z}_0 + \sum_{i=0}^{k-1} \mathrm{A}^{k-i-1}\mathrm{B}\mathrm{u}_i, \tag{4-9}$$

the future state is predicted with a subsampled model specific to that prediction horizon

$$\mathrm{z}_k = {}^{(k)}\mathrm{A}\mathrm{z}_0 + {}^{(k)}\mathrm{B}\mathrm{h}(\mathrm{u}_0, \mathrm{u}_1, \ldots, \mathrm{u}_{k-1}) \tag{4-10}$$

where h is some subsample function of the input. Here, this function adds the subsampled inputs as a control input. They are added to the existing control vector. More precisely $\mathrm{h} : \mathbb{R}^{n_{\mathrm{u}} \times n_{\mathrm{sub}}} \to \mathbb{R}^{n_{\mathrm{u}} n_{\mathrm{sub}}}$:

$$\mathrm{h}(\mathrm{u}_k, \mathrm{u}_{k+1}, \ldots, \mathrm{u}_{k+n_{\mathrm{sub}}}) = \begin{bmatrix} \mathrm{u}_k \\ \mathrm{u}_{k+1} \\ \vdots \\ \mathrm{u}_{k+n_{\mathrm{sub}}} \end{bmatrix} \tag{4-11}$$

The subsampled data matrices are

$$\begin{aligned} \mathrm{Z} &= \begin{bmatrix} | & | & & | \\ \mathrm{z}_0 & \mathrm{z}_1 & \cdots & \mathrm{z}_{n_\mathrm{k}-n_\mathrm{sub}-1} \\ | & | & & | \end{bmatrix}, \quad \mathrm{Z}^+ = \begin{bmatrix} | & | & & | \\ \mathrm{z}_{1+n_\mathrm{sub}} & \mathrm{z}_{2+n_\mathrm{sub}} & \cdots & \mathrm{z}_{n_\mathrm{k}} \\ | & | & & | \end{bmatrix} \\ \text{and } \Upsilon &= \begin{bmatrix} | & | & & | \\ \mathrm{h}(\mathrm{u}_0, \cdots, \mathrm{u}_{n_\mathrm{sub}}) & \mathrm{h}(\mathrm{u}_1, \cdots, \mathrm{u}_{n_\mathrm{sub}+1}) & \cdots & \mathrm{h}(\mathrm{u}_{n_\mathrm{k}-n_\mathrm{sub}-1}, \cdots, \mathrm{u}_{n_\mathrm{k}-1}) \\ | & | & & | \end{bmatrix}. \end{aligned} \tag{4-12}$$

4-3 Expected structure of the model

A tri-diagonal (or possibly more diagonals, depending on the sampling time) matrix structure for the A matrices is expected. This is due to the spatial locality of the dynamics. See subsection 3-3-3 for a detailed explanation of the local dynamics and their structure. The forcing term B is also assumed to have local properties. The dynamics and forcing for the spatial lifted vector (4-2) are expected to be of the following shape

$$\mathrm{x}_{k+1} = \begin{bmatrix} \beta_1 & \gamma_1 & & & \\ \alpha_2 & \beta_2 & \gamma_2 & & \\ & \alpha_3 & \ddots & \ddots & \\ & & \ddots & \ddots & \gamma_{n-1} \\ & & & \alpha_n & \beta_n \end{bmatrix} \mathrm{x}_k + \begin{bmatrix} \\ \\ \delta_1 \\ \delta_2 \\ \delta_3 \\ \\ \\ \end{bmatrix} \mathrm{u}_k \tag{4-13}$$

When time delay coordinates are considered, a specific structure to the lifted dynamics is expected as well.

$$\begin{bmatrix} \mathrm{x}_{k+1} \\ \mathrm{x}_k \\ \mathrm{x}_{k-1} \\ \vdots \\ \mathrm{x}_{k-n_\mathrm{d}+1} \end{bmatrix} = \begin{bmatrix} \mathrm{A}_{11} & \mathrm{A}_{12} & \mathrm{A}_{13} & \dots & \mathrm{A}_{1n_\mathrm{d}} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} \\ \vdots & & \ddots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathrm{x}_k \\ \mathrm{x}_{k-1} \\ \mathrm{x}_{k-2} \\ \vdots \\ \mathrm{x}_{k-n_\mathrm{d}} \end{bmatrix} + \begin{bmatrix} \mathrm{B}_1 \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \mathrm{u}_k, \tag{4-14}$$

where A_{1i} and B_1 are sub matrices structured as shown in (4-13) and $\mathbf{I}$ is the identity matrix and $\mathbf{0}$ a zero matrix.

4-4 Summary of the learning process

Several variables and choices must be made before a model can be learned in the learning process. This is done in four steps.

1. The first step is to choose what data to include. From the simulation data, the available choices for the full grid are:
   - Temperature,
   - Head and

- Flow, in three directions.

This choice also determines the value of n_{data}. For the input and output

- Hot injection temperature,
- Cold injection temperature,
- Flow into/out of the hot well,
- Flow into/out of the cold well,
- Temperature inside the hot well, and
- Temperature inside the cold well.

Also, a combination of these inputs can be chosen (i.e. temperature times flow).

2. The second step is to choose between a hybrid or normal model. For now, a hybrid system of two models exists, one for injection and one for extraction. The data has to be split between these modes. The unforced storage model is determined analytically and not learned from data.
3. The third step is the lifting of the data and forming of the data matrices. Here, the data matrices are formed depending on three parameters:
   - The number of spatial coordinates of the grid that are considered (n_{sc}),
   - The number of time-delayed coordinates (n_{d}), and
   - The number of subsamples (n_{sub}).
4. The fourth and final step is learning the model. Here, the choice for the algorithm and its parameters is made. The available algorithms and their parameters are:
   - Dynamic mode decomposition with control (DMDc)[1]. This algorithm has one parameter, the rank r of the A-matrix.
   - Physics-informed dynamic mode decomposition (piDMD)[2]. Here, the parameter to the model is the number of off-diagonal terms (n_{diag}) considered. Also, the structure caused by the Hankel lifting is enforced, see section 4-3.
   - Gershgorin DMD (GeDMD). The parameters are the same as for piDMD. Now, not only the model structure but also the stability is enforced. If the system is stable without constraints, the solution is the same as piDMD.

After completing these four steps, a linear (hybrid) DMD model for simulating ATES is obtained.

4-5 Properties of the models

For now, we have only discussed the state vector of the models. The desired output is only the temperature in the well at the current time. Thus, the output matrix has a single row containing one non-zero number, more precisely $\mathrm{C} \in \mathbb{R}^{1\times n_{\text{z}}}$. The non-zero number is determined by the scaling factor that is used to normalise the data. The location of the non-zero number is determined by the chosen number of spatial and time delay coordinates

$$\mathrm{C} = \begin{bmatrix} \mathbf{0}_{1\times n_{\text{sc}}} & c_{1,n_{\text{sc}}+1} & \mathbf{0}_{1\times n_{\text{sc}}} & \mathbf{0}_{1\times n_{\text{x}} n_{\text{d}}} \end{bmatrix}. \tag{4-15}$$

[1] The data is lifted with spatial and time delay coordinates. Therefore, the technically correct name would be extended (Hankel) dynamic mode decomposition with control (EHDMDc). For clarity, we choose the shorter notation DMDc.

[2] Again extended (Hankel) physics-informed dynamic mode decomposition with control (EpiDMDc) would be the technically correct name. For clarity, we choose the shorter notation piDMD.

Next to the model parameters, three model properties are relevant. First, the stability is computed with the spectral radius of the A-matrix. For the hybrid systems, the maximum spectral radius is considered[3].

Second, the number of unobservable modes is computed. For the hybrid models, the maximum number of unobservable modes is considered. The unobservable modes are of interest since, in the training data, we supply information about the states located underground, which cannot be measured in reality. Therefore, an initial state estimation is required. This is possible to do for unobservable modes. However, this state estimation might not be the real state of the system. If we want to retrieve the true value of the temperature underground (for simulating interactions between neighbouring ATES wells), the model has to be observable. More precisely, the observability matrix should be full rank. The number of unobservable modes is computed by subtracting the rank of the observability matrix from the dimension of A:

$$\dim(\mathrm{A}) - \operatorname{rank}\left(\begin{bmatrix} \mathrm{C}^{\mathrm{obs}} \\ \mathrm{C}^{\mathrm{obs}}\mathrm{A} \\ \vdots \\ \mathrm{C}^{\mathrm{obs}}\mathrm{A}^{n_z-1} \end{bmatrix}\right), \tag{4-16}$$

where n is the dimension of A and $\mathrm{C}^{\mathrm{obs}}$, is the observability output matrix, here the time delay coordinates of the well temperature are also considered as an output since we can measure them in reality. The observability output matrix is computed as

$$\mathrm{C}^{\mathrm{obs}} = \Big[\ \mathbf{0}_{1\times n_{\mathrm{sc}}} \quad c_{1,n_{\mathrm{sc}}+1} \quad \mathbf{0}_{1\times n_{\mathrm{sc}}} \quad \cdots \underbrace{\quad \mathbf{0}_{1\times n_{\mathrm{sc}}} \quad c_{1,n_{\mathrm{sc}}+1} \quad \mathbf{0}_{1\times n_{\mathrm{sc}}} \quad}_{\text{repeated } n_{\mathrm{d}} \text{ times}} \Big]. \tag{4-17}$$

Third, the model's degrees of freedom (DoF). The DoF of the model are how many parameters there are available for fitting. This differs greatly between the DMDc and piDMD algorithms. For DMDc, the DoF of A is the dimension of A times the rank. The dimension n_{z} is computed according to (4-5), making the DoF $n_{\mathrm{z}}r$ [4]. For the piDMD algorithm, the computation is different since no truncated SVD is used, and most of the A-matrix is fixed. The DoF of piDMD are computed by multiplying the length of the spatial lifted vector n_{z} from (4-2) times the number of off diagonals and delay coordinates: $n_{\mathrm{x}}n_{\mathrm{diag}}(n_{\mathrm{d}}+1)$. For a hybrid model, the total degrees of freedom are computed for a single matrix.

4-6 Hyperparameter optimization

A hyperparameter optimization strategy is implemented to choose the best possible hyperparameters for the learning algorithm. This is a non-convex integer optimization problem. To make the optimization problem more manageable, the problem is split between hybrid and non-hybrid models per DMD algorithm. So, a separate optimization is implemented for each combination of the following algorithms.

[3] The stability of a hybrid system cannot be concluded only from the individual stability of both systems. However, it does provide a good indication.

[4] This is only the vector v_i, the vector u_i is directly related to v_i and thus not an independent parameter. $\mathrm{u}_i = \frac{1}{\sigma_i}\mathrm{X}\mathrm{v}_i$, with $\mathrm{X} = \mathrm{U}\Sigma\mathrm{V}^{\mathsf{T}}$.

- Hybrid model
    1. Yes
    2. No
- Algorithm
    1. DMDc
    2. piDMD
    3. GeDMDc

The Hyperparameter optimization problem now has four decision variables:

- Number of spatial coordinates considered n_{sc}
- Number of time delay coordinates on the output n_{d}
- Prediction horizon (subsampling) n_{sub}
- Rank of the model r (DMDc) or the number of off-diagonals n_{diag} (piDMD and GeDMD).

The model performance is evaluated over a multiyear prediction horizon. This is done to promote memory in the system. DMD already optimizes the one-step-ahead prediction, the goal of the hyperparameter optimization is to find models that work well on longer prediction horizons.

The optimization problem is implemented in MATLAB using the genetic algorithm `ga()` and Bayesian optimization `bayesopt()` of the global optimization toolbox. The choice for these algorithms is made since they can handle integer constraints. Due to the non-convexity, the computation time is very long.

Chapter 5

ATES data collection

This study aims to provide a proof of concept for learning a simplified model of ATES for control. Therefore, a representative ATES system is required. Not only are the ATES system parameters relevant, but the local ground properties also greatly influence the system dynamics. These parameters are presented in section 5-1. Since ATES systems are situated deep underground, direct measurement of the entire system grid is impractical. Additionally, due to the extended timescales over which ATES systems operate, acquiring comprehensive input and output data over several decades is required for accurate analysis. Consequently, simulation data is employed to overcome these challenges. This simulation data is generated using MODFLOW, a high-fidelity solver that addresses the PDEs governing ATES dynamics. This is explained in section 5-2.

5-1 ATES system parameters

A case study about the thermal efficiency of over 400 ATES systems in the Netherlands is used to determine the parameters of a representative ATES system. The most prevalent system Storage capacity (V_{sto}) in the study is 100 000 to 150 000 $\text{m}^3\,\text{yr}^{-1}$. The system is chosen to be at the high end of this range since the average permit capacity is higher [4].

The distance between the wells is an important characteristic of the system. The closer the hot and cold wells are together, the more they interfere with each other. This is determined by the thermal radius. As a rule of thumb, 3 thermal radii are considered enough to limit interference and have a realistic footprint. The thermal radius is computed by [4]:

$$R_{th} = \sqrt{\frac{c_w V_{\text{sto}}}{c_{aq} \pi L}} \tag{5-1}$$

where c_w and c_{aq} are the volumetric heat capacity of water and the aquifer, respectively, and L is the screen length. The thermal radius for the chosen system and location is approximately 50 m. A summary of the ATES system parameters can be found in Table 5-1.

Table 5-1: Parameters of the modelled ATES system, based on [4, 8, 9, 11, 29, 30]

Parameter	Value	Unit
Screen length (L)	30	m
Screen location (below NAP)	20 to 50	m
Distance between wells	150 ($3R_{th}$)	m
Storage capacity (V_{sto})	150 000	$\text{m}^3\,\text{yr}^{-1}$
Maximum pumping rate	200	$\text{m}^3\,\text{h}^{-1}$
Hot injection temperature (T^{H}_{inj})	15	°C
Cold injection temperature (T^{C}_{inj})	9	°C

Since ATES parameters are very location-dependent, choosing a common location for ATES systems is necessary. From the literature, the location for the ground condition is chosen to be similar to that of Utrecht since many studies have modelled this area or done field testing [9–11, 29, 31, 32]. Even in a specific location, it is very difficult to determine the exact properties. Therefore, a lot of sources offering different ranges are combined to provide an estimate of the local ground conditions. See Table 5-2 for all parameters regarding the groundwater, aquifer and aquitard.

Table 5-2: Parameters of the ground conditions used in the simulations based on [4, 5, 8, 9, 11, 29, 33, 34]

Ground parameters	symbol	Value	Unit
Ambiend ground temperature	T_{amb}	12	°C
Ambiend groundwater flow		0	$\mathrm{m\,d^{-1}}$
Thermal conductivity sandy solids	$k\mathrm{T}_s$	2	$\mathrm{W\,m^{-1}\,K^{-1}}$
Thermal conductivity sandy clay	$k\mathrm{T}_{\text{clay}}$	1.7	$\mathrm{W\,m^{-1}\,K^{-1}}$
Thermal conductivity water	$k\mathrm{T}_f$	0.58	$\mathrm{W\,m^{-1}\,K^{-1}}$
Specific heat capcity solid	$c\mathrm{p}_s$	710	$\mathrm{J\,kg^{-1}\,K^{-1}}$
Specific heat capcity fluid (water)	$c\mathrm{p}_f$	4183	$\mathrm{J\,kg^{-1}\,K^{-1}}$
Density solids	ρ_s	2640	$\mathrm{kg\,m^{-3}}$
Density fluids	ρ_f	1000	$\mathrm{kg\,m^{-3}}$
Density (wet) bulk	ρ_b	$\rho_s(1-\theta)+\rho_f\cdot\theta$	$\mathrm{kg\,m^{-3}}$
Volumetric heat capacity fluid (water)	c_w	$c\mathrm{p}_f\cdot\rho_f$	$\mathrm{J\,m^{-3}\,K^{-1}}$
Volumetric heat capacity (solid)	c_s	$c\mathrm{p}_s\cdot\rho_s$	$\mathrm{J\,m^{-3}\,K^{-1}}$
Thermal distribution coefficient	K_{dist}	$c\mathrm{p}_s/(\rho_f\cdot c\mathrm{p}_f)$	$\mathrm{m^3\,kg^{-1}}$
Aquifer			
Depth (below NAP)		20 to 50	m
Horizontal hydraulic conductivity	K_h	25	$\mathrm{m\,d^{-1}}$
Vertical hydraulic conductivity	K_v	5	$\mathrm{m\,d^{-1}}$
Porosity	θ	0.35	–
Thermal conductivity aquifer bulk	$k\mathrm{T}_{\text{aq}}$	$k\mathrm{T}_s(1-\theta)+kT_f\cdot\theta$	$\mathrm{m\,d^{-1}}$
Thermal diffusivity	$T\mathrm{diff}_{\text{aq}}$	$k\mathrm{T}_{\text{aq}}/(\theta\cdot\rho_f\cdot c\mathrm{p}_f)\cdot 3600\cdot 24$	$\mathrm{m^2\,d^{-1}}$
Aquitard			
Depth (below NAP)		0 to 20 and 50+	m
Horizontal hydraulic conductivity	K_h	0.05	$\mathrm{m\,d^{-1}}$
Vertical hydraulic conductivity	K_v	0.01	$\mathrm{m\,d^{-1}}$
Porosity	θ	0.35	–
Thermal conductivity aquifer bulk	$k\mathrm{T}_{\text{aqt}}$	$k\mathrm{T}_{\text{clay}}(1-\theta)+kT_f\cdot\theta$	$\mathrm{m\,d^{-1}}$
Thermal diffusivity	$T\mathrm{diff}_{\text{aqt}}$	$k\mathrm{T}_{\text{aqt}}/(\theta\cdot\rho_f\cdot c\mathrm{p}_f)\cdot 3600\cdot 24$	$\mathrm{m^2\,d^{-1}}$

5-2 Simulations using MODFLOW

A detailed model based on PDEs is used to simulate ATES systems and gather learning data. These PDEs are discussed in section 2-3. To solve the PDEs MODFLOW-2005 [35] is used. MODFLOW is a finite-difference groundwater model with a modular structure. This modular structure allows the addition of packages to take more physical phenomena into account. For simulating ATES, two additional packages are required MT3DMS and SEAWAT [8, 16, 36, 37]

The code to make the simulations is implemented in Python. A Python package called FloPy [38] is used as a more user-friendly interface between the user and the MODFLOW solvers

written in Fortran. Specifically for simulating ATES, a library called PySeawATES is used [34].

For making simulations in MODFLOW, a number of settings are relevant. The discretization settings and boundary conditions can especially influence the results. In Table 5-3, the MODFLOW settings are provided. The sampling time of the model varies, and simulations have been made with hourly, daily, and monthly sampling times. For the remainder of this thesis, we will use daily data. Internally, MODFLOW adapts the time discretization steps of the solver automatically to achieve good numerical solutions to the PDEs.

Table 5-3: MODFLOW simulation settings [5, 29, 34].

Parameter	Value	Unit
xy-discretization	10	m
z-discretization	5	m
min linear grid distance from well	100	m
max grid distance from well	500	m
Top boundary condition	const. temperature, no flow	
Bottom boundary conmdition	const. temperature, no flow	

5-2-1 MODFLOW output

The output of the numerical simulations is the temperature T (°C), head h (m) and the flow or specific discharge $\mathrm{q} = [\ q_x \quad q_y \quad q_z\ ]^\mathsf{T}$ ($\mathrm{m\,d^{-1}}$) in the full x, y and z grid where the system is placed, see Figure C-1 for the grid produced by MODFLOW. An estimation has to be made of the well output temperature ($\mathrm{T^{out}}$). The MODFLOW simulations are not detailed enough to simulate a well with a diameter of a realistic ATES well (<1 m). The output temperature is approximated by the average temperature of the grid cells where the well is located. The grid cells around the well have a size of $10 \times 10 \times 5$m for $x \times y \times z$ respectively. The volume that approximates the well is $10 \cdot 10 \cdot 5 \cdot 6 = 3000\,\mathrm{m^3}$. This means that the well temperature will not immediately be equal to the well output temperature during injection.

5-3 Preparing the data

To minimize the amount of data for model training, we use a single line of coordinates from the 3D grid. First, we consider a 2D slice of the system in the xz-plane (at y=0), which intersects both wells. This is illustrated in Figure 5-1 for the temperature data. On this 2D slice, the line is chosen to run at a depth of 35 m. This is halfway along the screen length, thus in the middle of the storage volume. This line and the MODFLOW grid and axis directions are shown in Figure C-1. This will still provide enough informative data for the model since the data is symmetric around the plane cutting through the two wells. An assumption of ATES is that the stored heat and cold are in a cylindrical shape. This means that the temperatures are approximately the same at any height along the well screen (for homogenous ground conditions), this can be seen in Figure 5-1. Therefore, a line of coordinates is considered to

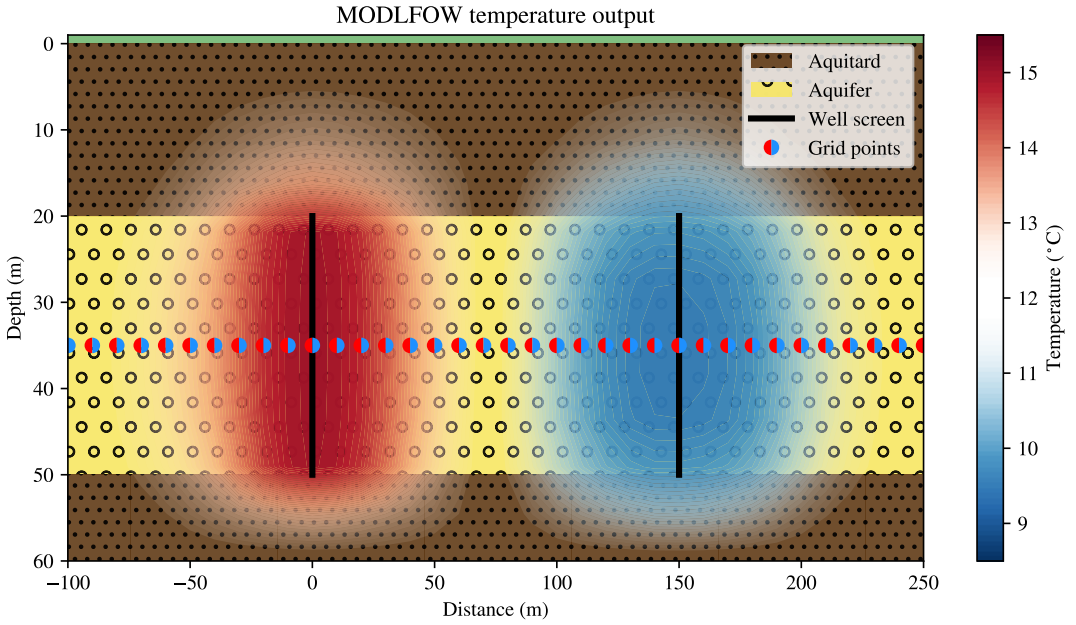

Figure 5-1: MODFLOW temperature output slice through the two wells (the zx-plane). The well-screen locations are indicated with the fat black lines. The aquitard and aquifer are shown with a brown and yellow background, respectively.

capture enough information to approximate the dynamics. This reduces the dimension of the state vector from 39 000 to 65 , if only temperature is considered ($n_{\text{data}} = 1$).

To make the data behave more linear the data is normalized and centered around zero. The flow and head data are already centred and only require normalization. For temperature data, we subtract the ambient temperature to establish an equilibrium at zero.

Chapter 6

Model performance: numerical simulation results

In this chapter, the results of the different learning strategies are shown. First, the parameters of seven different learned models are discussed in section 6-1. Next, the matrix structure of the models is compared to the expected results. After which, the models are compared to each other and the analytical ATES-well temperature model (AATM) as a baseline. This is done by computing the root mean squared error (RMSE) on the one-step-ahead prediction (section 6-2) and by comparing a prediction trajectory over a multi-year horizon (section 6-3).

6-1 Parameters and properties of the models

Multiple models are learned with different parameters. Models are learned for the different algorithms, hybrid and non-hybrid and with different numbers of spatial and time delay coordinates. The models presented here are only based on the temperature information for the full grid, and the input data is fixed as described in subsection 4-2-3. This means that step 1 from the summary of the learning process (section 4-4) is the same for all models. Including other data (e.g. the head or flow) always decreased model performance. The model's parameters and properties are summarised in Table 6-1. The models with an asterisk (*) indicate models where the parameters were found using the hyperparameter optimization strategy. The bold highlighted model is the best performing on the multi-year horizon prediction, which is shown in the subsequent plots.

Table 6-1: Models and their parameters.

	Data used	Hybrid	Spatial coordinates (n_{sc})	Time delayed coordinates (n_d)	Sub-samples (n_{sub})	Algorithm	Rank (r), or Off diagonals (n_{diag})	$\rho(A)$	Unobservable modes	DoF
hyb piDMD*	T	✓	0	34	1	piDMD	4	0.997	0	140
hyb DMDc no n_{sc}	T	✓	0	10	1	DMDc	3	0.997	0	33
non-hyb DMDc*	T	✗	15	1	1	DMDc	6	0.999	54	372
non-hyb piDMD*	T	✗	3	12	2	piDMD	3	1.001	0	273
hyb DMDc	T	✓	7	3	1	DMDc	9	1.000	47	540
hyb piDMD with n_{sc}	T	✓	7	3	1	piDMD	2	9.112	59	120
hyb GeDMD with n_{sc}	T	✓	7	3	1	GeDMD	2	1.000	21	120

*indicates models that are found using hyperparameter optimization, the other models are tuned manual.

Figure 6-1 illustrates the structure of the models. The extraction dynamic matrix (A^{ext}) for the last three models of Table 6-1 are displayed. The hyb DMDc full rank model has the same properties as the hyb DMDc model except for its full rank. The figure clearly shows that the expected model structure appears in the model learned with the DMDc algorithm with full rank. There is some noise, but the overall model structure is clear. For the model learned with piDMD, this structure is enforced, and Figure 6-1 shows that this is done correctly. If the rank of the DMDc model is not full rank, this matrix structure disappears. It can also be seen that the GeDMD algorithm produces the same structure as piDMD. However, the values inside the matrix are a factor 10 lower in this case. In section D-1, the matrices of the models are shown, including the injection and extraction matrices and the control matrices.

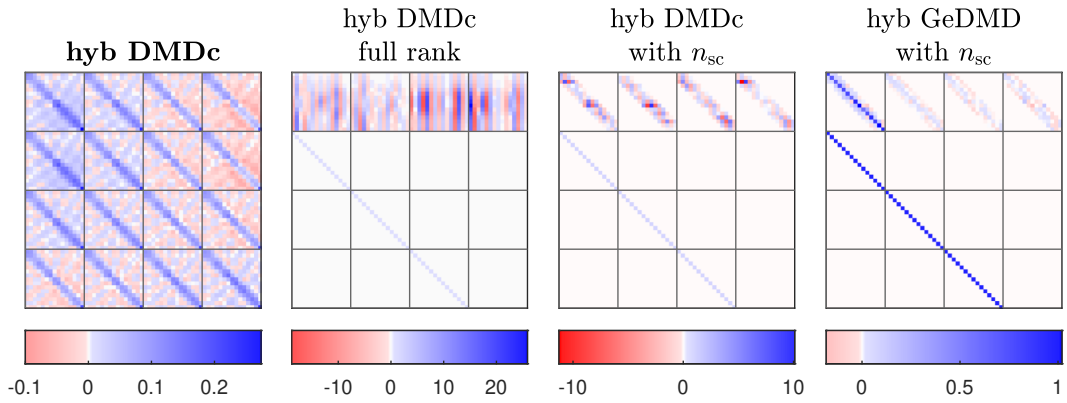

Figure 6-1: Matrix structure of models with different solving algorithms and rank. Note that each model has their own colour scale.

6-2 One-step-ahead prediction

To validate and compare the models, we first look at the one-step-ahead prediction. The RMSE is computed on the output to quantify the model performance. The one-step-ahead prediction is computed in terms of the data matrices as

$$\hat{\mathrm{Z}^{+}} = \hat{\mathrm{A}}\mathrm{Z} + \hat{\mathrm{B}}\Upsilon. \tag{6-1}$$

The prediction for the extraction and injection modes are made separately. The predicted output is computed by multiplying with C. Now the RMSE for a single operating mode can be computed using

$$\mathrm{RMSE} = \sqrt{\frac{1}{n_{\mathrm{k}}}} \left\| \mathrm{C}\left(\mathrm{Z}^{+} - (\hat{\mathrm{A}}\mathrm{Z} + \hat{\mathrm{B}}\Upsilon)\right) \right\|_2, \tag{6-2}$$

where Υ is the control input data matrix with only injection or only extraction control inputs. The values for the RMSE for the training, validation and normal operating data are shown in Table 6-2. For injection, the training and validation RMSE are very close to each other, so the model does not over-fit. This is the case for extraction, so the model might be overfitting. The RMSE is much lower for the normal operating conditions, which is to be expected since the forcing is less extreme and follows a much more predictable (sinusoidal) pattern.

Table 6-2: RMSE of different models of the one-step-ahead prediction temperature (°C) for injection and extraction. The RMSE is computed based on the training, validation, and normal operating data.

	Injection			Extraction		
	Training	Validation	Normal	Training	Validation	Normal
hyb piDMD	0.144	0.147	0.062	0.002	0.15	0.004
hyb DMDc no n_{sc}	0.357	0.359	0.21	0.004	0.205	0.004
non-hyb DMDc	0.546	0.469	0.353	0.344	0.398	0.554
non-hyb piDMD	0.419	0.421	0.087	0.256	0.244	0.082
hyb DMDc	0.174	0.176	0.062	0.003	0.146	0.027
hyb piDMD with n_{sc}	0.144	0.149	0.072	0.002	0.856	0.003
hyb GeDMD with n_{sc}	0.144	0.149	0.072	0.002	0.123	0.003

The AATM is not included in the RMSE table since it is difficult to supply it with the right initial conditions. These would have to be estimated for every step which involves solving a non-convex optimization problem.

Another method of validating a model is examining the autocorrelation of the residuals. If the model fully captures the system's dynamics, the residuals should be random. In Figure 6-2, the autocorrelation shows non-random residuals for the normal operating data, indicating that the model does not fully capture the dynamics. However, the residuals of the training and validation data do appear random. This is caused by the forcing of the training and validation data. Since the forcing is random, any unmodeled dynamics also lack a discernible pattern.

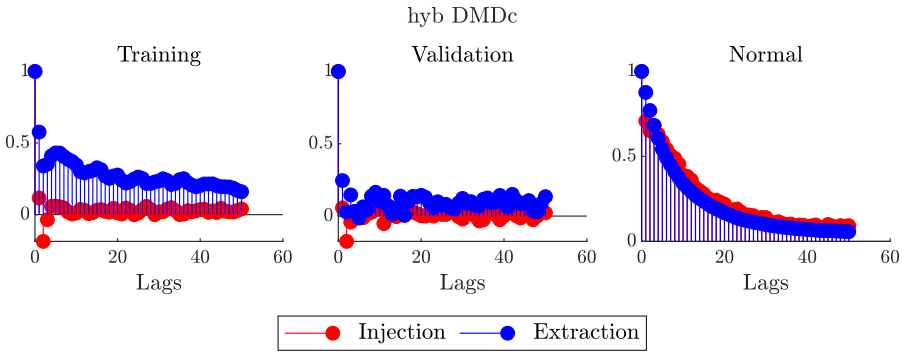

Figure 6-2: Autocorrelation of the one-step-ahead residual based on the training, validation, and normal operating data.

6-3 Multi-step-ahead prediction

To get a more interpretable sense of the model performance, a multi-step prediction is computed. In Figure 6-3, the trajectory of the best-performing hybrid DMDc model is plotted over a two-year horizon with normal operating conditions, meaning a constant injection temperature and sinusoidal flow rate with a yearly frequency. Next to the hybrid DMDc model, the output of the MODFLOW simulations is shown, this is considered as ground truth. As a benchmark, the prediction of the AATM is also plotted. Note that the AATM initial conditions and parameter α are fitted on the validation data.

During extraction, the model follows the trend very closely. During injection, the model fit is first very good but then deviates slightly. However, the output of the injection model is not the most relevant. During injection, the temperature in the well is very closely related to the input temperature. Far more important is that the internal states are at the right temperature. So that when the dynamics switch, the extraction model has the proper initial conditions. This appears to be working well since the extraction dynamics follow the truth quite closely. The model fit is far better than that off AATM.

In section D-2, the results of other DMD models with different parameters are shown. Interestingly, models that score well on the one-step-ahead prediction (see Table 6-2) do not necessarily have a good multi-step fit. The hyb DMDc model has a low score on the one-step-ahead prediction but performs best on the multi-step prediction. This could be caused by the choice of hyperparameters. DMD optimizes the one-step-ahead prediction, but the hyperparameter optimization looks for the best multi-step prediction on the normal operating data.

Figure 6-4 shows a more realistic operating condition where the injection temperature also varies, and noise is present on the inputs. The injection temperature rises with the temperature in summer. Again, the hybrid DMDc model performs well. It can predict the rise in extraction temperature caused by the varying injection temperature, where the AATM cannot. On the extraction side, the prediction diverges from the actual temperature near the end of the extraction period. This divergence may result from the model learning a steady rate of temperature drop that is not influenced by the flow rate. This issue is more evident in the other DMD models, as shown in Appendix D. Since the system exhibits bi-linear behavior with respect to the flow rate, a linear model cannot effectively utilize the flow rate

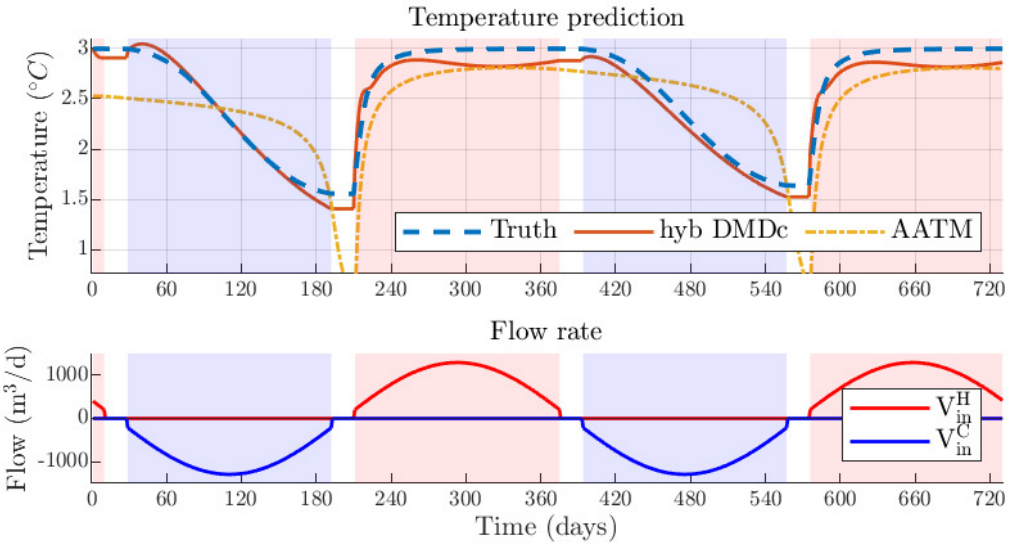

Figure 6-3: Two-year prediction trajectory of the hot well temperature. The coloured background indicates hot injection and extraction for red and blue, respectively. The injection temperature is constant at 3 °C. The temperatures are the deviation from the ambient ground temperature $T^{amb} = 12\,°C$.

as an input. Consequently, the model learns an average rate of temperature transport and loss independent of the amount of water being extracted.

In section D-3, the two-year prediction is shown for the cold well. The models perform worse for the cold well than for the hot well. The models seem to respond stronger to the flow rate than the hot well models. This could be caused by the lower variation in injection temperatures for the cold well. All models were tuned for hot well performance, if the models are tuned for the cold well similar results are expected.

The full grid predictions are shown in section D-4. The states are predicted quite accurately, although they slightly diverge over time. This can be attributed to the unobservability. The prediction is worse for the states closer to the cold well that reaches temperatures below the ambient temperature. This is to be expected since the cold well injection temperature is not provided as an input.

6-4 Computation time and complexity

The computation time of the (hybrid) linear models is very fast since they only depend on matrix multiplications. The exact speed depends on the dimension of the model, which is determined by the model's hyperparameters. The computation time of AATM is of the same order since it is also computed with only elementary operations in a for loop. Due to the hybrid nature of the model, they would both result in non-convex optimization, although for a hybrid linear model, more fast solvers are available than for a completely non-linear hybrid model. As discussed before, the switching between the modes is very predictable, which is a property that could be exploited in a solver.

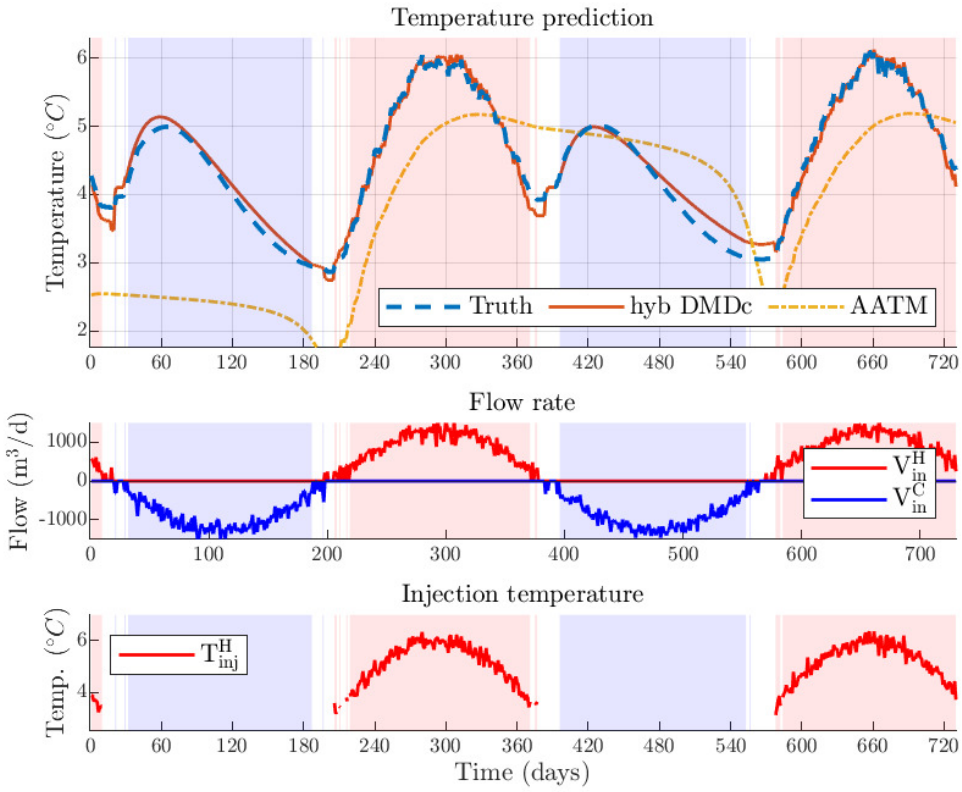

Figure 6-4: Two-year prediction trajectory of the hot well temperature. The coloured background indicates hot injection and extraction for red and blue, respectively. The injection temperature varies with the hottest point in the model of summer and follows a sinusoidal trend.

The computation times for 10 years of prediction are

- MODFLOW: 76.3 min = 4578 s (average over 4 experiments),
- Hybrid DMDc model: 0.011 s (average over 10 experiments), and
- AATM: 0.029 s (average over 10 experiments).

It is of course very logical that MODFLOW takes much longer, it also generates much more data than the other two models (1.8 GB for 10 years). Although the hybrid model has more internal states, it is still twice as fast as the AATM. This is probably due to the fact that matrix multiplications are extremely well optimized in MATLAB. And the implementation of the AATM in MATLAB might not be optimal.

Chapter 7

Conclusion and Discussion

7-1 Conclusion

The main goal of this thesis was to find an output temperature prediction model suitable for control of ATES. Linear hybrid models provide a good prediction estimation over a suitable horizon for ATES. Over multi-year prediction horizons, they do not diverge and follow the global trend well. The models have to be tested more rigorously on more complex operating conditions, but using dynamic mode decomposition for learning a control-oriented model for ATES shows promising results. The temperature at multiple points can be predicted accurately if the initial condition is known. However, the models are unobservable. This means that retrieving the temperature from grid points outside of the well is not possible.

The best-performing algorithm was dynamic mode decomposition with control (DMDc). The states were lifted with spatial and time delay coordinates. Physics-informed DMD (piDMD) did not improve model performance and often resulted in unstable models. The new algorithm Gershgorin DMD (GeDMD) can enforce stability in those cases but the performance is still less than that of DMDc on long prediction horizons.

7-2 Discussion

Spatial lifting The hyperparameter of the number of spatial coordinates considered could be formulated better. Now, the number of states away from the well is the hyperparameter. Increasing the number of points considered scales weirdly. Taking into account the grid cell next to the well does not add much information, and adding the next does not do much either. Only when increasing the number of points to an amount which is approximately larger than the thermal radius the full grid is covered, and the dynamics across the points are informative. A better approach would have been to choose a maximum distance to where the dynamics are of interest (this could be a separate hyperparameter) and then choose the number of discretization steps taken in this interval. Similar to the RC-model from subsection 2-4-3. This would involve interpolating the MODFLOW simulation results.

Subsampling Subsampling the model and learning models for different prediction horizons was expected to work better for longer prediction horizons since errors would accumulate less than by exponentiating A. However this expectation was not met. This is probably caused by the subsample function on the input. The choice is made to keep this function as simple as possible by stacking the subsampled control inputs into one bigger input vector. Attempts to have some weighted average where the energy flow (temperature times flow rate) into the system was kept the same did not provide better results. More investigation into good input subsampling functions is required.

Use of flow and head data Incorporating additional information, such as the flow or head, did not enhance model performance. Given the significant impact of flow on underground temperature distribution, providing flow data at every underground point should theoretically improve the model. Similarly, supplying flow times temperature, which appears in the PDEs, should boost performance. However, this was not observed. Learning the flow with DMD alone was effective, as was learning flow times temperature, but integrating this information into a single state vector to better predict temperature did not work. This outcome can be explained by the system's bilinear nature. When modelling using a linear structure, adding this information does not help. Providing this input in a single state vector also increased the complexity of the model since now the flow and head needed to be learned as well. A better implementation might have been to make a separate model for the flow and head and use the output of that model as an input to the temperature model. This implies that there is no influence of temperature on the flow and head, which is not entirely true, but a good approximation for the low temperatures ATES works on.

Hyperparameter optimization The hyperparameter optimization method did not perform well. It did improve the models without spatial coordinates and the non-hybrid models. However, it was not able to identify that a large number of spatial coordinates significantly improved the model performance on more complex conditions. In the objective function for the hyperparameter optimization, the normal operating validation data is used to make sure the model performs well under normal operation conditions and not just on the one-step-ahead prediction. A better strategy should be found where the memory and, thus, prediction on longer horizons is promoted.

Memory in data For learning, data with random inputs is used. This is good practice in system identification and avoids learning patterns that are (accidentally) present in the inputs. The lack of patterns in the training data could explain why the system's memory does not perform as well as expected when only time-delayed coordinates are used as observables. In the training data set of the hyperparameter optimization, there are some patterns, but storage was still not required for learning a good model, and thus, the model cannot deal with varying injection temperatures where this is required. Only models where the memory is forced into the model by adding spatial coordinates can handle this more difficult case. Therefore, training data of the hyperparameter optimization where memory is essential for good model performance should be used. When doing this, it is very important to avoid that the training data and validation data are the same.

Storage dynamics For now, the very simple dynamics $z_{k+1} = z_k$ represent the dynamics when the system is in storage mode. This is quite a reasonable approximation since the dynamics of ATES are very slow when not actuated. However, improving the performance using DMD on the unforced data should be very straightforward. This was not done due to time constraints.

DMDc vs piDMD The piDMD algorithm performs worse than the DMDc algorithm, with every tested model based on a two-year prediction. On the other hand, the piDMD models perform better on the one-step-ahead prediction. This is the exact opposite of what was expected. The main idea behind piDMD is that by enforcing physical laws known to be present in the dynamics, a model is learned to capture the essence of the dynamics better. For the same matrix dimension, the piDMD algorithm has fewer degrees of freedom than the DMDc algorithm. This would suggest that the DMDc algorithm would score better on the training data. The amount of data might have had an influence on this. From Figure 6-1, it can be seen that the DMDc algorithm clearly recognizes the structure expected from the Hankel lifting. So, enforcing those constraints with the piDMD algorithm does not add much information. If less data was available or data with more noise and DMDc did not have enough information to recognize this structure, piDMD could have an improved performance. Further investigation should be conducted into why piDMD did not perform better than DMDc, and what constraints are best to enforce.

Gershgorin DMD is able to enforce locality and stability. However, the algorithm is very slow compared to piDMD. There is quite a lot of room for speed up by optimizing the code, but at the end of the day, constrained convex optimization will always be much slower than unconstrained optimization for which an analytical solution exists. A possible solution to this would be to implement a penalty term on the Gersgorin norm instead of enforcing it with a constraint. This is similar to the process of stable DMD in section 3-3-3. This will return a slightly different solution but has the chance of being much faster. Another problem encountered with constraint optimization was the tolerances. Since we are solving an optimization problem inside a bisection algorithm, the tolerance of the inner optimization problem should be lower than that of the bisection algorithm.

7-3 Future work

A sound basis for using dynamic mode decomposition on ATES has been established. To improve the model further, some recommendations are made for further work. They are divided into three categories. First, how can the current model be improved, given the hybrid linear structure? Second, if better model performance is required and added model complexity is allowed, what can we do? Finally, we will discuss how to test the model more rigorously and under more realistic circumstances.

7-3-1 Improvements with the current model structure

To improve the current linear hybrid model, some steps can be taken:

- Learn a separate C and D matrix. Now, the output matrix C has been determined analytically, but this could be improved by fitting separate matrices. This is trivial to do using linear least squares, the expected increase in model performance is limited.
- Increasing the number of spatial coordinates to two or even three dimensions. As we have seen, more spatial coordinates improve model performance. This could be extended to not only include the coordinates to a line but also to a plane or even the full three dimensional space. This would greatly increase the required data and severely increase the computation time of the DMD algorithm. Still, by computing a low-rank approximation and projecting it onto the modes, a model of reasonable dimension should be obtained.
- Next to the delay coordinates on the state vector, input delay coordinates could be considered. The expected improvement is little, but it is straightforward to implement.
- As described in the discussion, the subsampling strategy did not give satisfactory results. However, there could still be an added benefit in finding a subsampled model.
- To improve long horizon predictions, a prediction error method could be used with the DMD model as a start for the optimization.
- A combined model for the hot and cold well would improve predictions for the underground states. If only one well is considered, the underground states between the two wells will have difficulty making accurate predictions since they do not have the input of the lower temperature from the other well.

7-3-2 Improvements for more complex model structures

DMD provides a reasonable approximation of ATES. To take a big step in model performance the model complexity has to be increased. Here, some possible options for more complex but more accurate models are discussed:

- Bi-linear DMD. The main culprit for low DMD model performance seems to be the combination of temperature and flow. Learning models of the flow works well. Models for the temperature work well. However, the combination does not. This is due to the bi-linearity between the temperature and the flow. This shows up in the PDEs and two of the models discussed in section 2-4. Bi-linear DMD allows for learning dynamics in the form of [39, 40]
$$\mathrm{x}_{k+1} = \mathrm{A}\mathrm{x}_k + \mathrm{N}\mathrm{x}_k\mathrm{u}_k + \mathrm{B}\mathrm{u}_k. \tag{7-1}$$
- More hybrid steps. Since we are already working with a hybrid model, one for injection, extraction and storage, why not increase the number off discretization steps? Thus learn separate models for not only different signs of the flow but also different magnitudes.
- The perfect Koopman operator has not been identified. For now, only interpretable and mostly linear lifting functions have been used. To get a better approximation of the Koopman operator, more complex and non-linear lifting functions should be evaluated. This can be done using Deep DMD/Koopman. Here, a neural network is used not only to learn the Koopman operator but also to find the lifting functions and their inverse [41]. This would still yield a linear model, but the lifting functions could be highly non-linear, and the observables could have little physical interpretation.

7-3-3 More realistic ATES conditions

In this work, the models have been trained and evaluated under idealized ATES conditions. Now that a proof of concept for DMD for ATES has been made, the models should be tested and updated with more complex and realistic data:

- Include ambient groundwater flow in the data.
- Include heterogeneity in the aquifer, meaning that the ground properties are not constant over the whole simulation domain.
- Extend to a 2D or even 3D prediction. This is desirable since a model capable of doing this can be used to model interactions between neighboring ATES systems.
- More than 1 ATES system should be included in the simulation. This is in line with the previous point of being able to predict and thus optimally control interactions between neighbouring wells. By running simulations, a combined model for multiple ATES systems could be developed, or separate models can be learned where the boundary conditions are set by simulations from the neighbouring model, allowing distributed control approaches.

Bibliography

[1] Andreu Angel, Magdalena Berberich, Wolfgang Birk, Christoph Brunner, Marco Calderoni, Maria Joao Carvalho, Guglielmo Cioni, Luis Coelho, Alice Denarie, Christian Doczekal, et al. *Strategic Research and Innovation Agenda for Climate-Neutral Heating and Cooling in Europe*. 2020. URL: https://www.rhc-platform.org/content/uploads/2020/10/EUREC-Brochure-RHC-SRI-06-2022-WEB.pdf.

[2] IEA. *Renewables 2019 - Market analysis and forecast from 2019 to 2024*. Oct. 2019. URL: https://www.iea.org/reports/renewables-2019.

[3] Martin Bloemendal, Martin Van der Schans, Stijn Beernink, Niels Hartog, and Philip J. Vardon. "Drivers to allow widespread adoption of ATES systems: a reflection on 40 years experience in The Netherlands". In: *Symposium on Energy Geotechnics 2023* (Oct. 2023). DOI: 10.59490/seg.2023.557.

[4] Martin Bloemendal and Niels Hartog. "Analysis of the impact of storage conditions on the thermal recovery efficiency of low-temperature ATES systems". In: *Geothermics* 71 (Jan. 2018), pp. 306–319. DOI: 10.1016/J.GEOTHERMICS.2017.10.009.

[5] Martin Bloemendal. *Lecture slides - CEGM2006 Subsurface Storage for Energy, Water and Climate Applications*. Delft University of Technology 2023.

[6] Martin Bloemendal, Jan Hoogendoorn, and Sjaak Rijk. *Bodemenergie & Drinkwaterwinningen*. Nov. 2016. URL: https://www.h2owaternetwerk.nl/index.php/vakartikelen/bodemenergie-drinkwaterwinningen.

[7] Stijn Beernink, Niels Hartog, Philip J. Vardon, and Martin Bloemendal. "Heat losses in ATES systems: The impact of processes, storage geometry and temperature". In: *Geothermics* 117 (Feb. 2024), p. 102889. DOI: 10.1016/J.GEOTHERMICS.2023.102889.

[8] Ruben Johannes Caljé. "Future use of Aquifer Thermal Energy Storage below the historic centre of Amsterdam". In: *TU Delft, Department of Water Management, Delft* (2010).

[9] W. T. Sommer, P. J. Doornenbal, B. C. Drijver, P. F.M. van Gaans, I. Leusbrock, J. T.C. Grotenhuis, and H. H.M. Rijnaarts. "Thermal performance and heat transport in aquifer thermal energy storage". In: *Hydrogeology Journal* 22 (1 Nov. 2014), pp. 263–279. DOI: 10.1007/S10040-013-1066-0/TABLES/5. URL: https://link.springer.com/article/10.1007/s10040-013-1066-0.

[10] Martin Bloemendal, Theo Olsthoorn, and Frank Boons. "How to achieve optimal and sustainable use of the subsurface for Aquifer Thermal Energy Storage". In: *Energy Policy* 66 (Mar. 2014), pp. 104–114. DOI: 10.1016/J.ENPOL.2013.11.034.

[11] Vahab Rostampour, Marc Jaxa-Rozen, Martin Bloemendal, Jan Kwakkel, and Tamás Keviczky. "Aquifer Thermal Energy Storage (ATES) smart grids: Large-scale seasonal energy storage as a distributed energy management solution". In: *Applied Energy* 242 (May 2019), pp. 624–639. DOI: 10.1016/J.APENERGY.2019.03.110.

[12] Renato Macciotta. *Encyclopedia of Engineering Geology*. Ed. by Peter T. Bobrowsky and Brian Marker. Springer Cham, 2018. DOI: https://doi.org/10.1007/978-3-319-73568-9. URL: https://link.springer.com/referencework/10.1007/978-3-319-73568-9.

[13] Sape A. Miedema. "The Delft Sand, Clay & Rock Cutting Model: 3rd edition". In: *TU Delft OPEN Textbooks* (Oct. 2019). DOI: 10.5074/T.2019.001.

[14] Gualbert Oude Essink. *Modelling of groundwater flow and solute transport*. Aug. 2002. URL: https://www.slideserve.com/josie/modelling-of-groundwater-flow-and-solute-transport.

[15] Lhoussaine El Mezouary and Bouabid El Mansouri. "Groundwater flow equation, overview, derivation, and solution". In: *E3S Web of Conferences* 314 (Oct. 2021). DOI: 10.1051/E3SCONF/202131404007.

[16] Christian D Langevin, Daniel T Thorne Jr., Alyssa M Dausman, Michael C Sukop, Weixing Guo, and Geological Survey (U.S.) *SEAWAT Version 4: A Computer Program for Simulation of Multi-Species Solute and Heat Transport*. 2008. DOI: 10.3133/tm6A22. URL: https://pubs.usgs.gov/publication/tm6A22.

[17] Vahab Rostampour, Marc Jaxa-Rozen, Martin Bloemendal, and Tamás Keviczky. "Building Climate Energy Management in Smart Thermal Grids via Aquifer Thermal Energy Storage Systems". In: *Energy Procedia* 97 (Nov. 2016), pp. 59–66. DOI: 10.1016/J.EGYPRO.2016.10.019.

[18] Vahab Rostampour, Martin Bloemendal, and Tamas Keviczky. "A model predictive framework of Ground Source Heat Pump coupled with Aquifer Thermal Energy Storage System in heating and cooling equipment of a building". In: *Proceedings 12th IEA Heat Pump Conference* (2017), pp. 0–000. URL: https://repository.tudelft.nl/islandora/object/uuid%3A95255ef7-7267-421a-b3ea-7b701c6064d9.

[19] Vahab Rostampour, Martin Bloemendal, M. Jaxa-Rozen, and T. Keviczky. "A control-oriented model for combined building climate comfort and aquifer thermal energy storage system". In: *Proceedings European Geothermal Congress 2016* (2016). URL: https://repository.tudelft.nl/islandora/object/uuid%3A47a49a2f-32a9-42e7-b81c-15ddf796a54c.

[20] Elisa Scalco, Angelo Zarrella, Alessandro Maccarini, and Alireza Alireza. "An aquifer thermal energy storage model for efficient simulations of district systems". In: *CLIMA 2022 conference* (May 2022). DOI: 10.34641/CLIMA.2022.346. URL: https://proceedings.open.tudelft.nl/clima2022/article/view/346.

[21] Joshua L. Proctor, Steven L. Brunton, and J. Nathan Kutz. "Dynamic mode decomposition with control". In: *SIAM Journal on Applied Dynamical Systems* 15 (1 Sept. 2014), pp. 142–161. DOI: 10.1137/15M1013857. URL: https://arxiv.org/abs/1409.6358v1.

[22] Peter J Schmid. "Dynamic mode decomposition of numerical and experimental data". In: *Journal of Fluid Mechanics* 656 (2010), pp. 5–28. DOI: DOI:10.1017/S0022112010001217.

[23] Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz. "On Dynamic Mode Decomposition: Theory and Applications". In: *Journal of Computational Dynamics* 1 (2 Nov. 2013), pp. 391–421. DOI: 10.3934/jcd.2014.1.391.

[24] Christopher W. Curtis, D. Jay Alford-Lago, Erik Bollt, and Andrew Tuma. "Machine Learning Enhanced Hankel Dynamic-Mode Decomposition". In: (Mar. 2023). URL: https://arxiv.org/abs/2303.06289v3.

[25] Peter J. Baddoo, Benjamin Herrmann, Beverley J. McKeon, J. Nathan Kutz, and Steven L. Brunton. "Physics-informed dynamic mode decomposition (piDMD)". In: (Dec. 2021). URL: https://arxiv.org/abs/2112.04307v1.

[26] Giorgos Mamakoukas, Orest Xherija, and Todd Murphey. "Memory-Efficient Learning of Stable Linear Dynamical Systems for Prediction and Control". In: *Advances in Neural Information Processing Systems* 2020-December (June 2020). URL: https://arxiv.org/abs/2006.03937v3.

[27] Fletcher Fan, Bowen Yi, David Rye, Guodong Shi, and Ian R. Manchester. "Learning Stable Koopman Embeddings". In: *Proceedings of the American Control Conference* 2022-June (Oct. 2021), pp. 2742–2747. DOI: 10.23919/ACC53348.2022.9867865. URL: https://arxiv.org/abs/2110.06509v1.

[28] Ashin Mukherjee and Ji Zhu. "Reduced rank ridge regression and its kernel extensions". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 4 (6 Dec. 2011), pp. 612–622. DOI: 10.1002/SAM.10138. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/sam.10138.

[29] Wijbrand Teunis Sommer. "Modelling and monitoring of Aquifer Thermal Energy Storage : impacts of soil heterogeneity, thermal interference and bioremediation". In: (June 2015). URL: https://research.wur.nl/en/publications/modelling-and-monitoring-of-aquifer-thermal-energy-storage-impact.

[30] *NVOE Werkwijzen en richtlijnen ondergrondse energieopslag | Bodemenergie Branchevereniging*. May 2019. URL: https://branchevereniging.bodemenergie.nl/nvoe-werkwijzen-en-richtlijnen-ondergrondse-energieopslag/nvoe-werkwijzen-en-richtlijnen-ondergrondse-energieopslag/.

[31] Stijn Beernink, Martin Bloemendal, Rob Kleinlugtenbelt, and Niels Hartog. "Maximizing the use of aquifer thermal energy storage systems in urban areas: effects on individual system primary energy use and overall GHG emissions". In: *Applied Energy* 311 (Apr. 2022), p. 118587. DOI: 10.1016/J.APENERGY.2022.118587.

[32] Martin Bloemendal, Marc Jaxa-Rozen, and Theo Olsthoorn. "Methods for planning of ATES systems". In: *Applied Energy* 216 (Apr. 2018), pp. 534–557. DOI: 10.1016/J.APENERGY.2018.02.068.

[33] TNO Geologische Dienst Nederland. *DINOloket Data en Informatie van de Nederlandse Ondergrond*. URL: https://www.dinoloket.nl/.

[34] Martin Bloemendal, Marc Jaxa-Rozen, Theo Olsthoorn, and Stijn Beernink. *PySeawATES*. 2020. URL: https://github.com/martinbloemendal/PySeawATES.

[35] Arlen W. Harbaugh. "MODFLOW-2005, The U.S. Geological Survey Modular Ground-Water Model—the Ground-Water Flow Process". In: U.S. Geological Survey, 2005. URL: https://pubs.usgs.gov/tm/2005/tm6A16/PDF.htm.

[36] Chunmiao Zheng and P. Patrick Wang. *MT3DMS: A Modular Three-Dimensional Multispecies Transport Model for Simulation of Advection, Dispersion, and Chemical Reactions of Contaminants in Groundwater Systems; Documentation and User's Guide*. Dec. 1999. URL: https://apps.dtic.mil/sti/citations/ADA373474.

[37] Vivek Bedekar, Eric D. Morway, Christian D. Langevin, and Matthew J. Tonkin. "MT3D-USGS version 1: A U.S. Geological Survey release of MT3DMS updated with new and expanded transport capabilities for use with MODFLOW". In: *Techniques and Methods* (2016). DOI: 10.3133/TM6A53.

[38] U.S. Geological Survey. *FloPy: Python Package for Creating, Running, and Post-Processing MODFLOW-Based Models*. 2021. URL: https://www.usgs.gov/software/flopy-python-package-creating-running-and-post-processing-modflow-based-models.

[39] Andy Goldschmidt, E. Kaiser, J. L. Dubois, S. L. Brunton, and J. N. Kutz. "Bilinear dynamic mode decomposition for quantum control". In: *New Journal of Physics* 23 (3 Mar. 2021), p. 033035. DOI: 10.1088/1367-2630/ABE972. URL: https://iopscience.iop.org/article/10.1088/1367-2630/abe972.

[40] Ion Victor Gosea and Igor Pontes Duff. "Toward fitting structured nonlinear systems by means of dynamic mode decomposition". In: *International Series of Numerical Mathematics* 171 (Mar. 2020), pp. 53–74. DOI: 10.1007/978-3-030-72983-7_3. URL: https://arxiv.org/abs/2003.06484v2.

[41] Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. "Deep learning for universal linear embeddings of nonlinear dynamics". In: *Nature Communications 2018 9:1* 9 (1 Nov. 2018), pp. 1–10. DOI: 10.1038/s41467-018-07210-0. URL: https://www.nature.com/articles/s41467-018-07210-0.

Appendix A

Relation between the Frobenius and Gershgorin norm and the spectral radius

The figures below show the relation between the stability and the norm of a matrix for two different norms. The matrices are created randomly with `A = 5*randn(nx)`. Next, using an eigen decomposition, the matrix's eigenvalues are scaled to ensure that we have a nice distribution of spectral radii to plot. The plots show a linear relation between the norm of a matrix and the spectral radius. The slope differs for every matrix size and type, but the relationship remains linear. The larger the matrix size, the better the relation.

Listing A.1: MATLAB code used to generate random matrices with self-chosen spectral radius.

```
for i=1:m
    A = 5*randn(nx);

    [V, D, W] = eig(A);
    max_eig = max(abs(diag(D)));
    sf = eig_val(i);
    scale = sf/(max_eig+eps);
    A_tilde = scale*V*D/(W'*V)*W';
    A_tilde = real(A_tilde);

    max_eig(i) = abs(eigs(A_tilde,1));
    norm_fro(i) = norm(A_tilde, 'fro');
    gershgor(i) = Gershgorin(A_tilde);
end
```

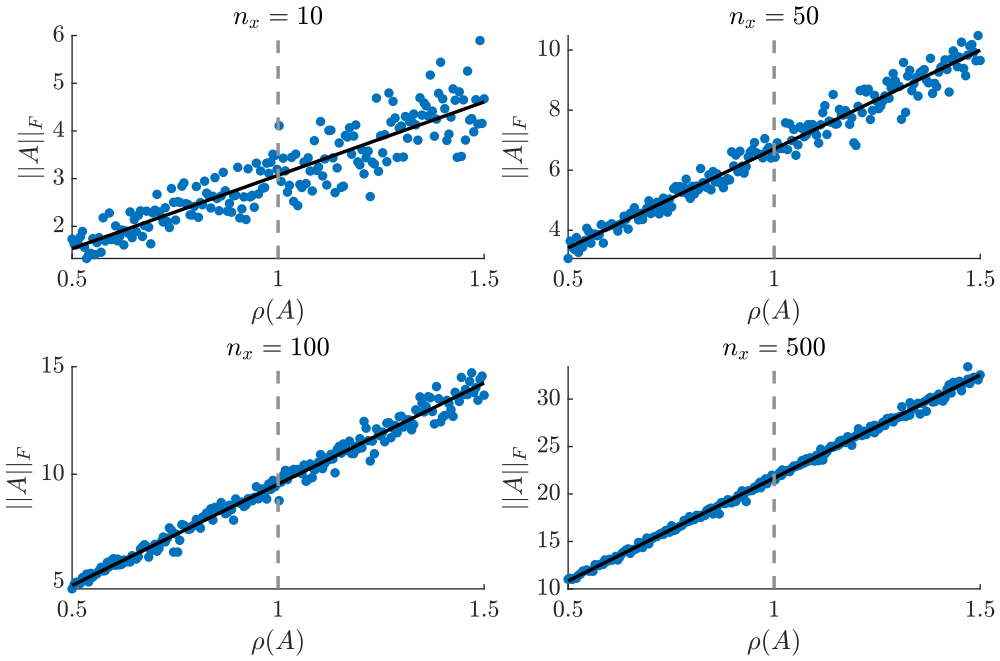

Figure A-1: The Frobenius norm of a square matrix plotted against the spectral radius for different matrix sizes. The line shows the least squares best fit through the points.

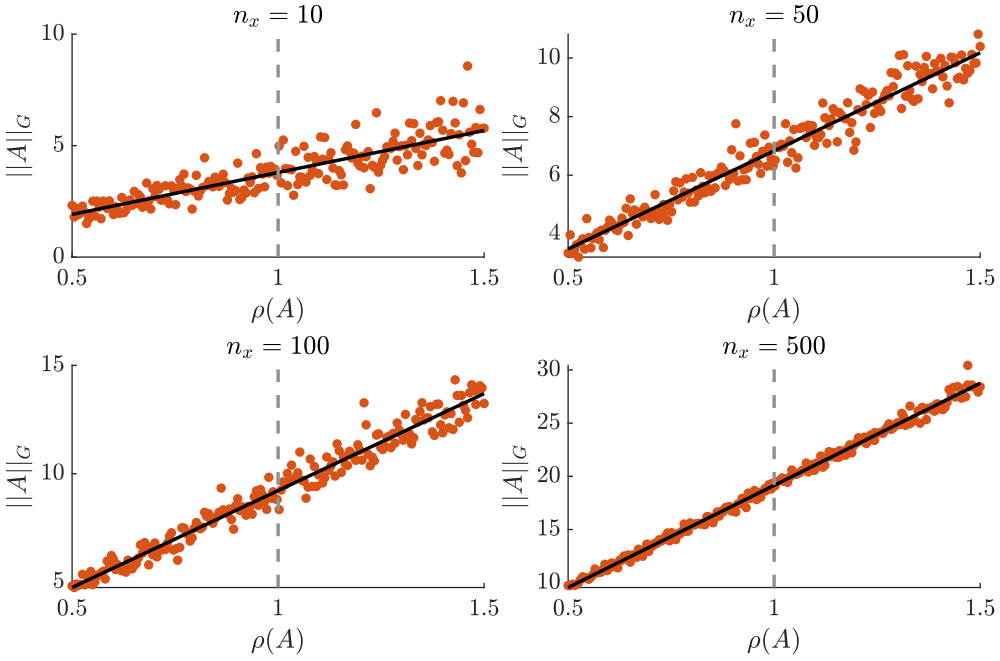

Figure A-2: The Gershgorin norm of a square matrix plotted against the spectral radius for different matrix sizes. The line shows the least squares best fit through the points.

Appendix B

Gerhsgorin DMD optimization with linear constraints

The original formulation

$$\begin{aligned} \underset{\alpha_i,\beta_i,\gamma_i,\delta_i}{\operatorname{argmin}} \; & \left\| \alpha_i \mathrm{x}^{\mathsf{T}}_{(i-1,*)} + \beta_i \mathrm{x}^{\mathsf{T}}_{(i,*)} + \gamma_i \mathrm{x}^{\mathsf{T}}_{(i+1,*)} - \mathrm{x}'^{\mathsf{T}}_{(i,*)} + \delta_i \mathrm{u}^{\mathsf{T}}_{(i,*)} \right\|_2^2 \\ \text{s.t.} \; & |\alpha_i| + |\beta_i| + |\gamma| < \lambda \end{aligned} \tag{B-1}$$

can be rewritten to include only linear constraints by defining additional optimization variables for each absolute value (t_{α_i} for α_i)

$$\begin{aligned} \underset{\substack{\alpha_i,\beta_i,\gamma_i,\delta_i,\\ t_{\alpha_i},t_{\beta_i},t_{\gamma_i}}}{\operatorname{argmin}} \; & \left\| \alpha_i \mathrm{x}^{\mathsf{T}}_{(i-1,*)} + \beta_i \mathrm{x}^{\mathsf{T}}_{(i,*)} + \gamma_i \mathrm{x}^{\mathsf{T}}_{(i+1,*)} - \mathrm{x}'^{\mathsf{T}}_{(i,*)} + \delta_i \mathrm{u}^{\mathsf{T}}_{(i,*)} \right\|_2^2 \\ \text{s.t.} \; & \begin{aligned} t_{\alpha_i} + t_{\beta_i} + t_{\gamma_i} &< \lambda \\ \alpha_i - t_{\alpha_i} &< 0 \\ -\alpha_i - t_{\alpha_i} &< 0 \\ \beta_i - t_{\beta_i} &< 0 \\ -\beta_i - t_{\beta_i} &< 0 \\ \gamma_i - t_{\gamma_i} &< 0 \\ -\gamma_i - t_{\gamma_i} &< 0 \end{aligned} \end{aligned} \tag{B-2}$$

Appendix C

MODFLOW grid

The number of grid cells is $n_x = 65, n_y = 50, n_z = 12$, making the total number of grid cells 39 000. If all the available data is used, this will produce a state vector of dimension 195 000 (per time step). In Figure C-1 the grid of MODFLOW is shown in all three dimensions. Note that not the full grid is shown. Toward the edges, the grid cell size increases. The final cells are located approximately 1000 m from the wells.

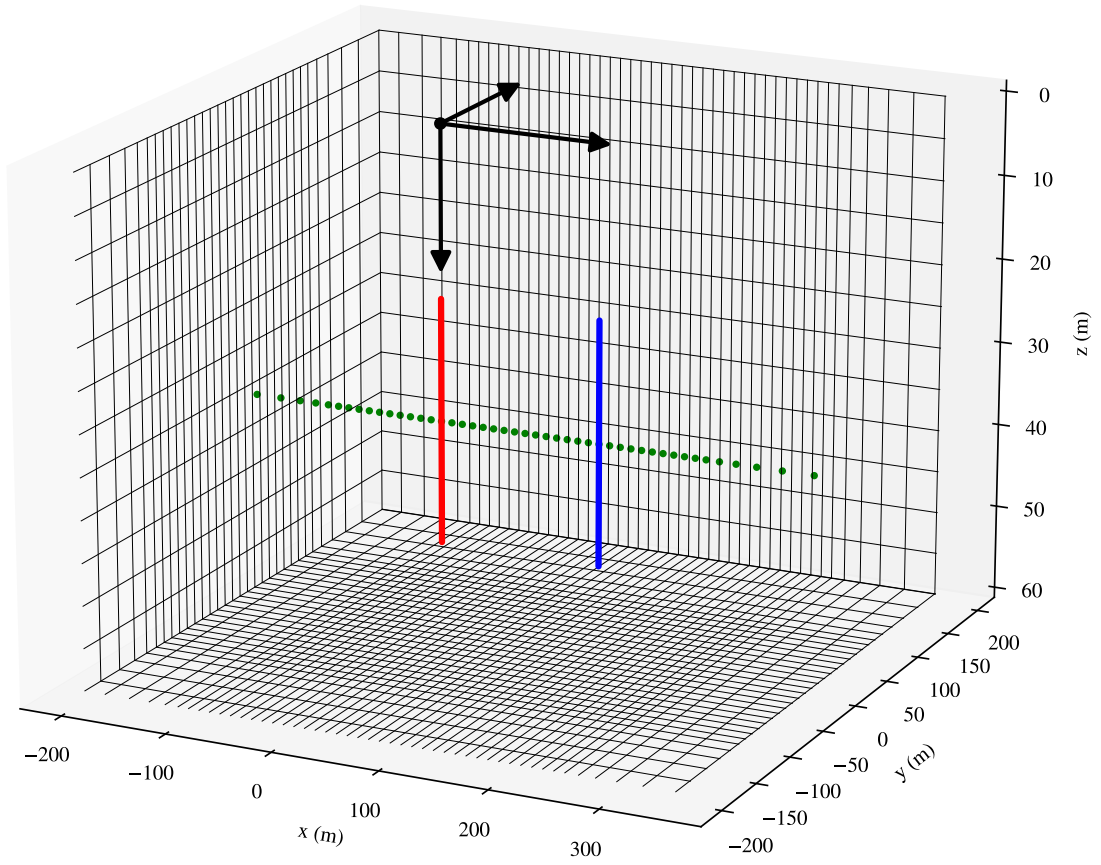

Figure C-1: The grid produced by MODFLOW. Around the wells, the grid cells are evenly spaced with a size of $10 \times 10 \times 5\mathrm{m}$ for $x \times y \times z$ respectively. Further away from the well, the grid cells increase in size. The origin is located at ground level above the hot well. The green dots indicate the line of data used for learning.

Appendix D

Additional results

D-1 Matrix structure

In Figure D-1, the full matrix structure for the hyb DMDc model is shown compared with the same model but with full rank.

In Figure D-2, the full matrix structure of the piDMD and GeDMD are shown.

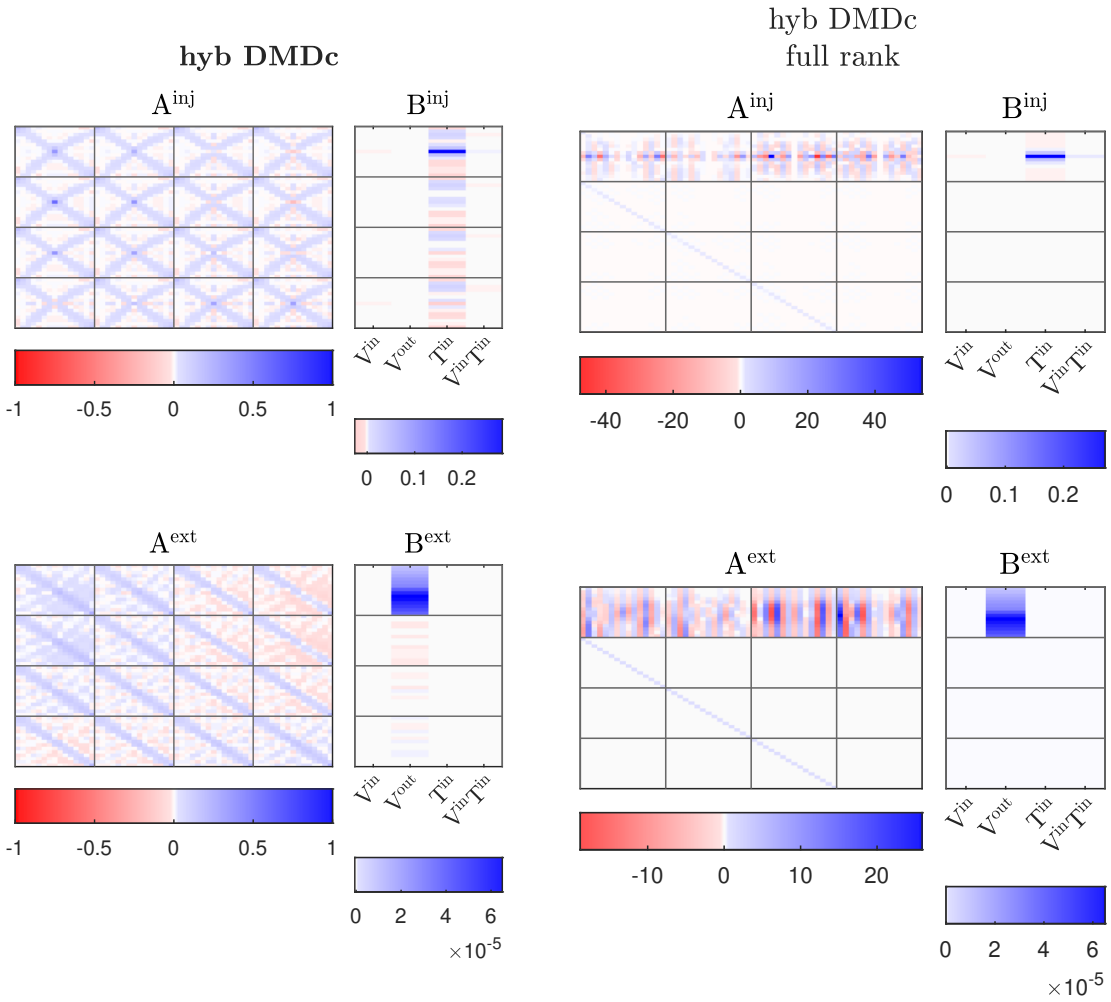

Figure D-1: Matrix structure of models with different rank. The number of spatial and time delay coordinates are equal for both models. **Left:** rank deficient DMDc model. **Right:** full rank hybrid DMDc model.

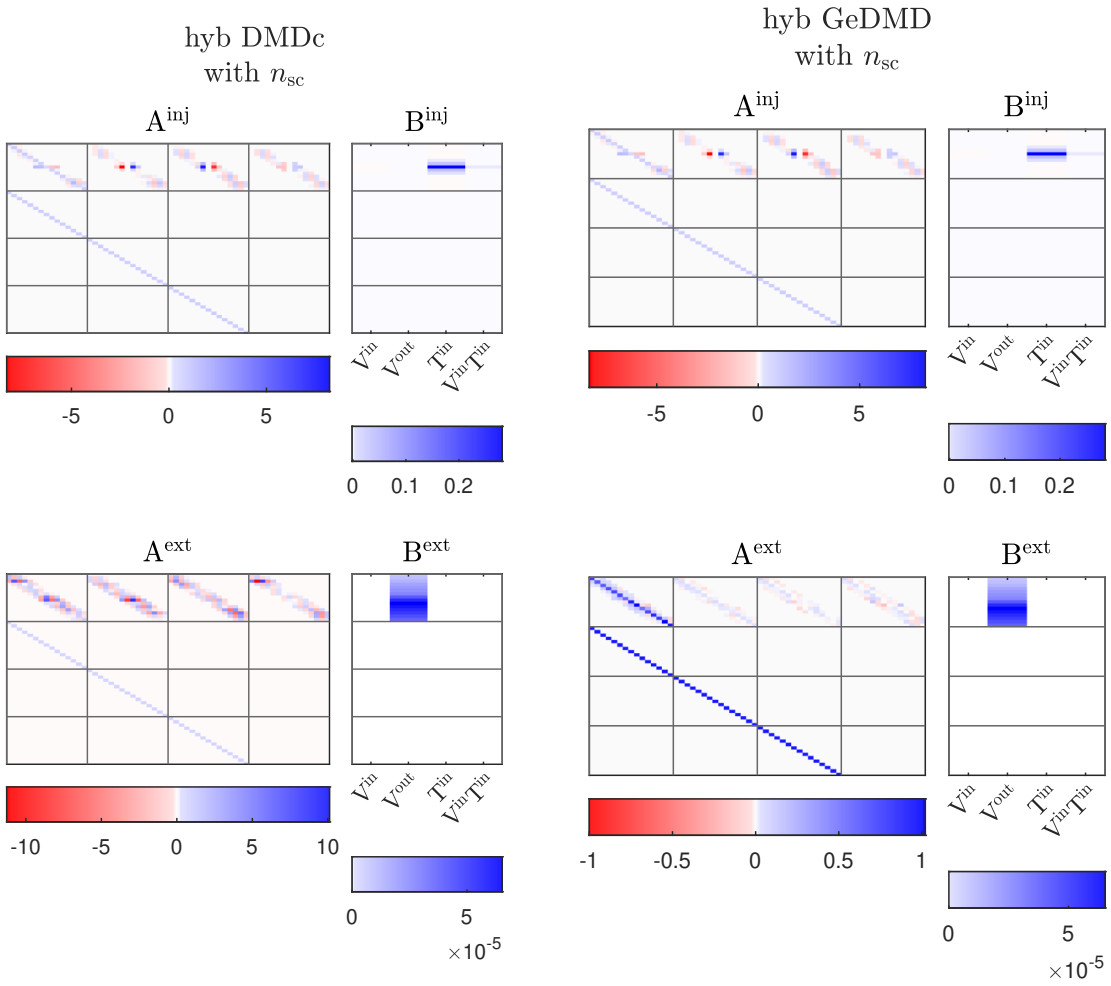

Figure D-2: Matrix structure of models trained with different algorithms. The number of spatial and time delay coordinates are equal for both models. **Left:** piDMD model. **Right:** GeDMD.

D-2 All models plotted

In Figure D-3, all models are plotted. What is interesting to note is that models that score well on the one-step-ahead prediction (see Table 6-2 do not necessarily have a good multi-step fit. Especially the hyb DMDc model has a low score on the one-step-ahead prediction but performs best on the multi-step prediction. This could be caused by the hyperparameter optimization. DMD optimizes the one-step-ahead prediction, but the hyperparameter optimization looks for the best multi-step prediction on the normal operating data.

Figure D-4 shows a more realistic operating condition where the injection temperature also varies. The injection temperature rises with the temperature in summer.

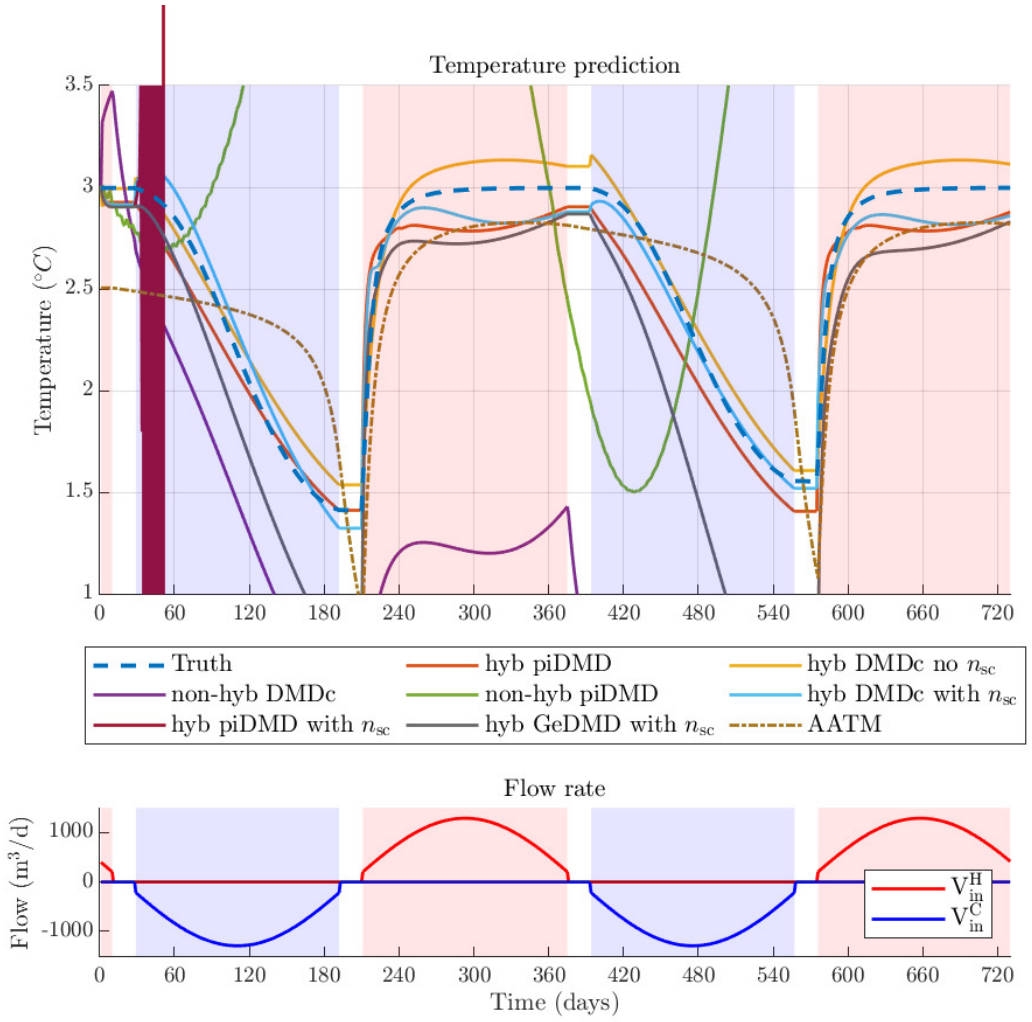

Figure D-3: Two-year prediction trajectory of the hot well temperature with the corresponding input. The coloured background indicates hot injection and extraction (cold injection) for red and blue, respectively. The injection temperature is constant.

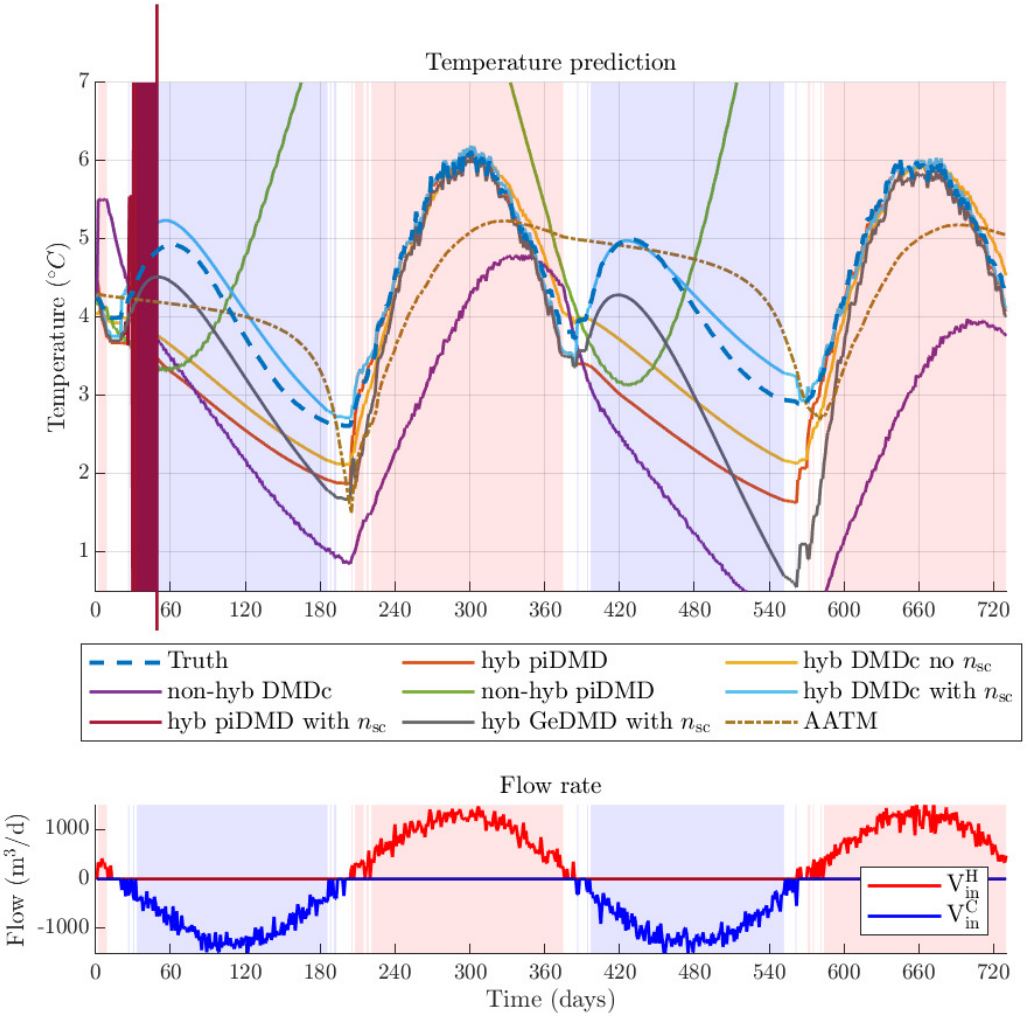

Figure D-4: Two-year prediction trajectory of the hot well temperature with the corresponding input. The coloured background indicates hot injection and extraction (cold injection) for red and blue, respectively. The injection temperature varies.

D-3 Cold well prediction

In Figure D-5, the prediction for the cold well is made. This figure is similar to Figure D-4 except now we look at the cold well.

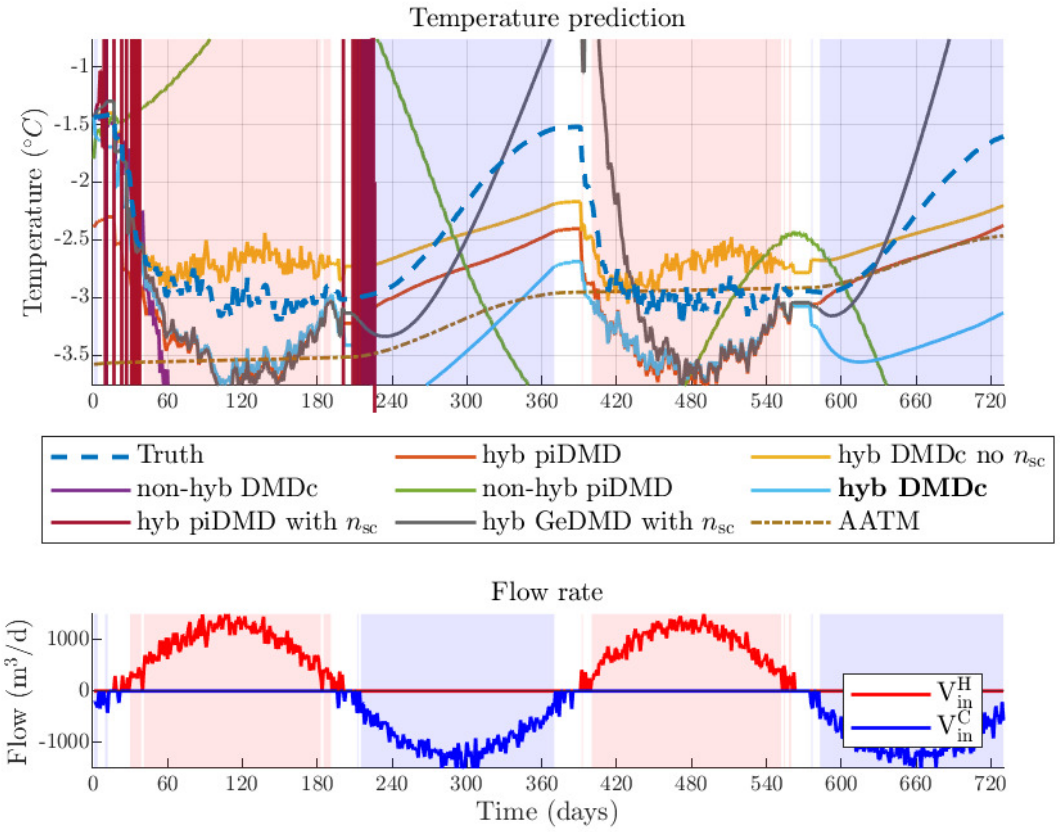

Figure D-5: Temperature prediction of the cold well

D-4 Full grid prediction

In Figure D-6, all the underground states of the model are plotted. The inputs and model are the same as from Figure 6-3. To make the figure less crowded, the states are split into two parts. One part going to the left of the well, this is towards the outside of the system. The other states are going to the other well. Here, it is clear that there is interference from the cold well since the temperatures are below the ambient temperature.

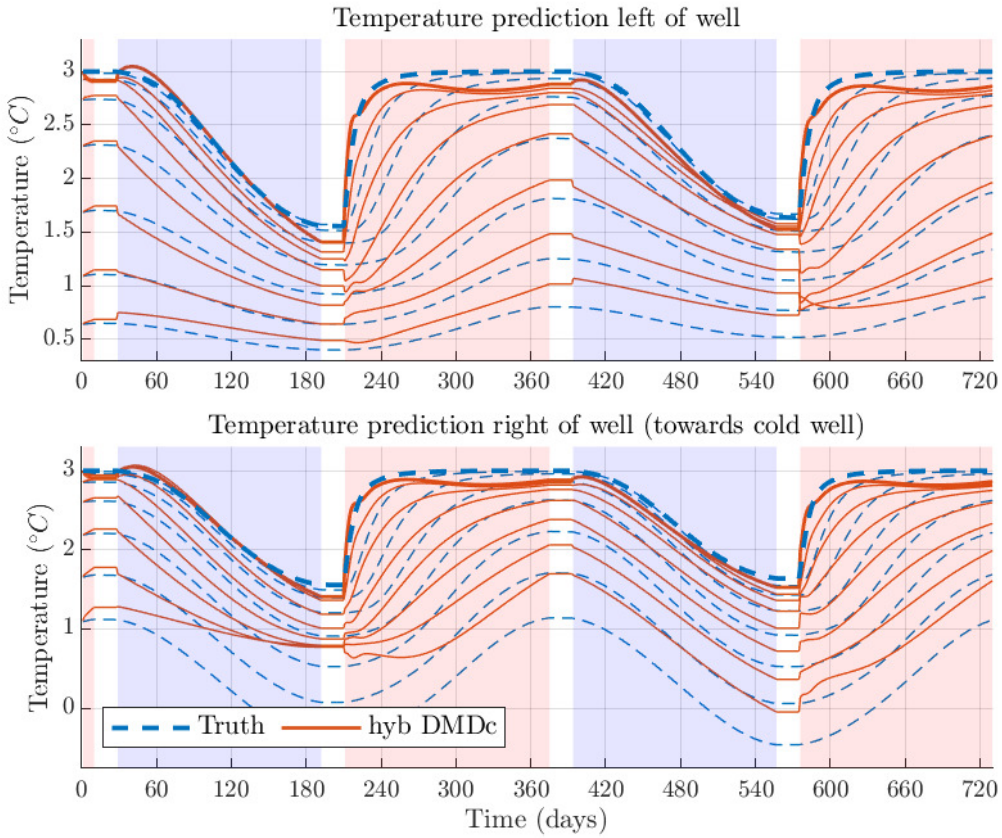

Figure D-6: Full state temperature prediction. The thinner lines indicate states further away from the well. The top plot shows states moving away from the well towards the outside of the system. The lower plot shows the states moving towards the cold well.