MASTER THESIS

FINAL REPORT

NOISE ANALYSIS FOR BIOMOLECULAR SIGNAL DIFFERENTIATORS

Shuxin Chen

Master Embedded Systems

Guided by Luca Laurenti

Delft Center for Systems and Control

Delft University of Technology, the Netherlands

August 23, 2023

Abstract

Noise's impact on biochemical systems has long been a focal point of investigation, given its potential to compromise signal accuracy and disrupt system functionality [31]. This paper conducts a comprehensive exploration into the noise characteristics within a set of signal differentiators recognized for their high precision. Noteworthy for their modularity, swift computation, and ease of implementation, these differentiators play a pivotal role in computing concentration changes and bear the potential to regulate the dynamics of biological systems.

This study establishes a comprehensive simulation framework to examine the noise characteristics of these differentiators across diverse input signal scenarios. Furthermore, we also apply noise suppression techniques such as noise filters to mitigate excessive noise and enhance noise performance.

Our findings reveal that these differentiators significantly amplify the system noise level, surpassing both the Poisson level and the original system noise level. Moreover, while noise filters demonstrate notable success in noise reduction, achieving Poisson-level noise without compromising signal integrity remains a challenge.

This investigation yields invaluable insights into the noise properties of biochemical differentiators, shedding light on their inherent limitations. Additionally, it presents a viable pathway to enhance noise behaviour, thereby extending the scope of applications for these differentiators.

Contents

Chapter 1

Introduction

Biological systems encompass both internal and external sources of noise, the former arising from the intrinsic stochastic nature of biochemical reactions, while the latter includes factors like temperature, humidity, and food availability [39]. When organisms encounter noise, they exhibit adaptive behaviours by making adjustments, such as altering their diet to regulate mammalian influx calcium or utilizing lactation to export calcium [3]. These behaviours can be mathematically abstracted into basic operations, including addition and subtraction. Moreover, organisms can also perform more complex calculations, such as derivations. For instance, in 2010, Shimizu et al. discovered that the chemotaxis-signalling pathway in *Escherichia coli* is capable of computing time derivatives [49].

In the field of physics, differentiators play a crucial role. They are employed to differentiate signals and obtain important parameters such as rates, and with simple modifications, they can be transformed into controllers, similar to the "D" (derivative) component in classical PID (Proportional-Integral-Derivative) controllers. Consequently, the discovery of differential mechanisms in biology has gained the interest of researchers, as it has the potential to facilitate derivative calculations and control processes in the field of biochemistry.

Due to their importance, many novel designs of biochemical differentiators have been proposed [2] [28], but there has been limited research on the noise properties of biochemical differentiators, which holds great importance. Noise introduces stochastic fluctuations into the system, making it impossible to predict the system's evolution with absolute certainty [46]. These fluctuations can also introduce errors into information or corrupt it [46].

Hence, before these architectures can be practically applied, it is crucial to study their noise properties. This paper aims to analyze the noise behaviour of one specific biomolecular differentiator, **Biomolecular Signal Differentiators (BioSD)** [2], which can calculate the time derivative around the steady states. The BioSD includes 3 typologies, namely BioSD-I, II and

III. BioSD is selected for its modularity, short delay, and high accuracy, distinguishing it from other differentiators.

The investigation commences with the meticulous modelling of BioSD topologies using the syntax of **Chemical Reaction Networks (CRNs)** [37]. Noise quantification is achieved by employing the **Fano factor** [47], a widely-adopted metric for measuring noise intensity. The analytical exploration is facilitated through the utilization of the simulation tool **Kaemika** [9], which employs the **Linear Noise Approximation (LNA)** method [37]. Rigorous experiments, encompassing a diverse array of input signals, reveal a consistent trend: the noise levels of BioSD outputs persistently transcend the **Poisson level** and frequently surpass the intrinsic noise present in the initial input signal. Consequently, the application of **Noise Filters** [38] is investigated to temper this excessive noise.

In summary, this paper makes significant contributions to our understanding of the noise characteristics and performance of BioSD. The key contributions are as follows:

- Noise Level Revelation: This study unveils that the noise levels in BioSD outputs are notably higher compared to their input signals. This highlights the critical importance of employing noise suppression techniques.

- Noise Filtering Implementation: The paper introduces the application of noise filters to BioSD outputs, effectively reducing the noise levels and subsequently enhancing accuracy and reliability.

- Accuracy Validation: The accuracy of BioSD calculations is rigorously validated through experiments involving intricate input signals. Despite the challenging nature of these inputs, BioSD showcases commendable accuracy, further substantiating its potential practical utility.

- Parameter-Noise Relationship: The paper uncovers an intricate relationship between the parameter b of BioSD and the corresponding noise levels. This insight sheds light on the role of parameter configuration in influencing noise behaviour, thereby providing valuable guidance for optimization.

1.1 THESIS OUTLINE

The thesis is organized as follows:

Chapter 2 introduces the foundational modelling language, Chemical Reaction Networks (CRNs), which serves as the basis for subsequent chapters. Both deterministic and stochastic semantics are discussed, and under the stochastic framework, various approaches to modelling the system are explored.

Chapter 3 provides a comprehensive review of biochemical differentiators, with a specific focus on Biomolecular Signal Differentiators (BioSD) and Transfer Function Differentiators (TFD). This chapter highlights the respective advantages and disadvantages of these differentiators and conducts a comparative analysis, ultimately revealing BioSD's superiority over TFD.

Moving on to Chapter 4, we devote ourselves to the design of noise analysis. We first conduct an in-depth review concerning noise, which covers its effects, measurement techniques, and methods for suppression. This lays the groundwork for understanding the significance of noise analysis and provides valuable insights for experimental design. We then discuss the selection of the simulation software tool and the choice of input signals for testing. Additionally, we present the reference derivative used for comparison to assess the accuracy of BioSD. Subsequently, we input the selected test signals to BioSD and verify the accuracy.

Chapter 5 unveils the outcomes of the noise analysis conducted using the methodologies outlined in Chapter 4. Here, we obtain the noise levels of BioSD outputs corresponding to each test signal, and compare them with the noise level of the respective test signal as well as the Poisson level. This examination enables us to delve into the correlation between a specific BioSD parameter and the resulting noise, achieved through controlled adjustments of this parameter and subsequent noise level testing. Furthermore, we introduce noise filtering techniques to effectively attenuate the noise levels.

Chapter 6 is dedicated to presenting the conclusion and discussing the findings. We summarize the key contributions of this study and provide insights into future directions for further research and improvements in the field of biochemistry.

Chapter 2

Chemical Reaction Networks

Biochemical systems are constructed using biochemical reactions. These reactions are effectively modelled using a programming language called **Chemical Reaction Networks (CRNs)** [52]. The versatility of CRNs enables their application in various research areas, including noise filters [38] and molecular programming [51].

CRNs are based on a premise called the **Well-Mixed Assumption**, which assumes that the system is well-stirred, and all molecules are uniformly distributed [37]. This assumption ensures that the spatial location of molecules does not influence the likelihood of reactions, thereby excluding unrelated disturbances.

At any given moment, the reactions of molecules are randomized with some probability. When the copy numbers of molecules are sufficiently large, the fluctuations resulting from this inherent randomness become negligible, allowing for deterministic modelling of the system [26]. This scenario is relatively straightforward and is referred to as **Deterministic Semantics**. However, when the copy numbers of molecules are low, stochastic effects become significant and cannot be disregarded. In order to accurately account for the uncertainty in such situations, the **Stochastic Semantics** is employed.

In this section, we begin by introducing the syntax of CRNs in Section 2.1. We then delve into the deterministic modelling approach in Section 2.2, providing an in-depth exploration of its principles and calculation process. Subsequently, in Section 2.3, we shift our focus towards stochastic models and their associated methodologies, accompanied by relevant information and considerations.

2.1 Syntax

A Chemical Reaction Network (CRN) $C = (\Lambda, R)$ consists of two sets: Λ, which represents all the chemical species involved, with $|\Lambda|$ denoting the size of the species set, and R, which represents the set of reactions, with n denoting the number of reactions [37]. The species in Λ interact with each other based on the reactions in R. Each element $\tau_i = (r_{\tau_i}, p_{\tau_i}, k_{\tau_i})$ ($i \in \mathbb{N}^+$, $i \leq n$) in R represents the i-th reaction. Here, $r_{\tau_i} \in \mathbb{N}^{|\Lambda|}$ and $p_{\tau_i} \in \mathbb{N}^{|\Lambda|}$ are two sets that represent the stoichiometry of the reactants and products of the reaction, respectively. And $k_{\tau_i} \in \mathbb{R}{>}\ 0$ denotes the **reaction rate coefficient**. The net change of species in this reaction is given by $v_{\tau_i} = p_{\tau_i} - r_{\tau_i}$. If the initial condition x_0, which sets the initial concentration of each species, is also included, then the network $C = (\Lambda, R, x_0)$ is referred to as a chemical reaction system (CRS) [37].

Example 2.1 *Consider the following reactions:*

$$\begin{aligned} \emptyset &\xrightarrow{100} b \\ a + b &\xrightarrow{10} c \\ b + c &\xrightarrow{1} a \\ c &\xrightarrow{0.1} \emptyset \end{aligned}$$

The corresponding CRN is as follows:

$$\begin{aligned} &C = (\Lambda, R) && \\ &\Lambda = \{a, b, c\} && |\Lambda| = 3 \\ &R = \{\tau_1, \tau_2, \tau_3, \tau_4\} && n = 4 \\ &\tau_1 = ([0,0,0],[0,1,0],100) && v_{\tau_1} = [0,1,0] \\ &\tau_2 = ([1,1,0],[0,0,1],10) && v_{\tau_2} = [-1,-1,1] \\ &\tau_3 = ([0,1,1],[1,0,0],1) && v_{\tau_3} = [1,-1,-1] \\ &\tau_4 = ([0,0,1],[0,0,0],0.1) && v_{\tau_4} = [0,0,-1] \end{aligned}$$

2.2 DETERMINISTIC SEMANTICS

In deterministic semantics, CRNs are described using a set of deterministic Ordinary Differential Equations (ODEs) known as **rate equations** [37]. These rate equations represent the concentration change of each species over time:

$$F\big(\Phi(t)\big) = \frac{d\Phi(t)}{dt} \tag{2.1}$$

where $F\big(\Phi(t)\big)$ is known as **drift term**, and $\Phi(t) : \mathbb{R}_{\geq 0} \to \mathbb{R}^{|\Lambda|}$ describes the concentration of each species against time.

To obtain the rate equations, the **Law of Mass Action**, also known as **Mass Action Kinetics**, is used. This principle states that the reaction rate is proportional to the product of reactant concentrations to the power of its stoichiometry [42] [34], and it was first introduced by Cato Maximilian Guldberg and Peter Waage in 1864 [25] to study the chemical affinity between reactants. The Law of Mass Action has found applications not only in chemistry [54] and biology but also in pharmacology [35], epidemiology [30], and even economics [48].

Using Mass Action Kinetics, the rate equation can be obtained through the following procedure:

1. Calculate the **propensity rate** $\alpha_{\tau_i}\big(\Phi(t)\big)$ for each reaction τ_i, and arrange them in a vector $\alpha_\tau\big(\Phi(t)\big)$ in the order of reactions, from top to bottom. These rates represent the likelihood of the reaction occurring at time t. The propensity rate can be calculated using the Law of Mass Action, which multiplies the reaction rate coefficient with the concentration of each species raised to the power of its reactant stoichiometry (see the equation below).

$$\alpha_{\tau_i}\big(\Phi(t)\big) = k_{\tau_i} \prod_{j\in\Lambda} \Phi_j(t)^{r_{\tau_i}(j)} \tag{2.2}$$

$$\alpha_\tau\big(\Phi(t)\big) = \begin{bmatrix} \alpha_{\tau_1}\big(\Phi(t)\big) \\ \alpha_{\tau_2}\big(\Phi(t)\big) \\ \vdots \\ \alpha_{\tau_n}\big(\Phi(t)\big) \end{bmatrix}$$

2. Calculate the net change v_{τ_i} for each reaction τ_i, and generate the row vector v_τ in the order of reactions, but from left to right.

$$v_\tau = \begin{bmatrix} v_{\tau_1} & v_{\tau_2} & \cdots & v_{\tau_n} \end{bmatrix}$$

3. The rate equation is obtained by the dot product of the net change row vector and propensity rate vector $F(\Phi(t)) = v_\tau \cdot \alpha_\tau$. Each element in the resulting vector is actually the expect value of concentration change of the corresponding species at a given time, which is also the concentration change rate.

This process is further demonstrated by the following example:

Example 2.2 *Considering the same chemical reaction network (CRN) system as illustrated in Example 2.1, we proceed to calculate the drift term by first determining the propensity rate for each reaction:*

$$
\begin{array}{rll}
(reactions) & (propensity\ rates) & (net\ changes) \\
\emptyset \xrightarrow{100} b & 100 & [0\ 1\ 0] \\
a+b \xrightarrow{10} c & 10\Phi_a(t)\Phi_b(t) & [-1\ -1\ 1] \\
b+c \xrightarrow{1} a & \Phi_b(t)\Phi_c(t) & [1\ -1\ -1] \\
c \xrightarrow{0.1} \emptyset & 0.1\Phi_c(t) & [0\ 0\ -1]
\end{array}
$$

And then, we dot product the net change row vector and propensity rate vector:

$$
\begin{aligned}
F(\Phi(t)) &= \begin{bmatrix}[0\ 1\ 0] & [-1\ -1\ 1] & [1\ -1\ -1] & [0\ 0\ -1]\end{bmatrix} \cdot \begin{bmatrix} 100 \\ 10\Phi_a(t)\Phi_b(t) \\ \Phi_b(t)\Phi_c(t) \\ 0.1\Phi_c(t) \end{bmatrix} \\
&= \begin{bmatrix} 0 & 100 & 0 \end{bmatrix} + \begin{bmatrix} -10\Phi_a(t)\Phi_b(t) & -10\Phi_a(t)\Phi_b(t) & 10\Phi_a(t)\Phi_b(t) \end{bmatrix} + \\
&\quad \begin{bmatrix} \Phi_b(t)\Phi_c(t) & -\Phi_b(t)\Phi_c(t) & -\Phi_b(t)\Phi_c(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0.1\Phi_c(t) \end{bmatrix} \\
&= \begin{bmatrix} -10\Phi_a(t)\Phi_b(t) + \Phi_b(t)\Phi_c(t) \\ 100 - 10\Phi_a(t)\Phi_b(t) - \Phi_b(t)\Phi_c(t) \\ 10\Phi_a(t)\Phi_b(t) - \Phi_b(t)\Phi_c(t) - 0.1\Phi_c(t) \end{bmatrix}^T
\end{aligned}
$$

From the drift term, we can get the concentration change rate for each species:

$$
\begin{aligned}
\dot\Phi_a(t) &= -10\Phi_a(t)\Phi_b(t) + \Phi_b(t)\Phi_c(t) \\
\dot\Phi_b(t) &= 100 - 10\Phi_a(t)\Phi_b(t) - \Phi_b(t)\Phi_c(t) \\
\dot\Phi_c(t) &= 10\Phi_a(t)\Phi_b(t) - \Phi_b(t)\Phi_c(t) - 0.1\Phi_c(t)
\end{aligned}
$$

2.3 Stochastic Semantics

As mentioned previously, deterministic semantics utilize the expected changing rate of species to represent CRNs. However, to investigate the noise or the randomness in species concentrations, deterministic semantics is not sufficient. In such cases, we turn to stochastic semantics to address our requirements and capture the stochastic nature of the system.

In this section, we delve into the stochastic semantics of CRNs. We begin by introducing the fundamental concept of a **Markov process**, which serves as the cornerstone of stochastic modelling and is regarded as one of the most essential types of random processes [50]. Subsequently, we explore **Chemical Master Equation (CME)**, **Gillespie Algorithm** and **Linear Noise Approximation (LNA)**. Each of these three methods has its own pros and cons, and is applicable to different types of applications.

2.3.1 Markov Process

A Markov process $\{X(t), t \in T\}$ is a stochastic process with the **Markov property**, which implies that the conditional probability density has no memory. In other words, the probabilities of future events only depend on the current state [20]. A **Markov chain** is a special case of the Markov process where the state space S is discrete. Depending on whether the time domain T is continuous or not, the Markov Chain can be classified into **Continuous-Time Markov Chains (CTMCs)** and **Discrete-Time Markov Chains (DTMCs)** [29].

For a discrete process $X(t), t \in T$ to be a Markov chain, it must satisfy the condition that for any $n > 0$, any $t_1 < t_2 < ... < t_n < t_{n+1}$ in the time domain T, any states $i_1, i_2, ..., i_n$, and any state j in the state space S, the following equality holds:

$$\begin{aligned} &Pr\{X(t_{n+1}) = j|X(t_1) = i_1, ..., X(t_n) = i_n\} \\ =&Pr\{X(t_{n+1}) = j|X(t_n) = i_n\} \qquad (n \subseteq \mathbb{N}, \quad n \geq 1) \end{aligned} \tag{2.3}$$

where $Pr\{B|A\}$ denotes the conditional probability density of event B given event A, also known as the **transition probability** from event A to event B [29].

The transition probability on the right-hand side of the equal sign in *Equation 2.3* can be viewed as a one-step square transition matrix $P_{ij} = [p_{ij}]$, where one step corresponds to one time interval, i represents all the states that can reach state j in one step, j is the destination state, and the each element p represents the conditional probabilities of transitioning from state i to state j [29].

By extending the number of steps to n, the transition matrix becomes:

$$P_{ij}^{(n)} = Pr\{X(t_{m+n}) = j|X(t_m) = i\} \tag{2.4}$$

And the n-step transition probabilities can be calculated by using the **Chapman-Kolmogorov equation** [21] for all $n, m \geq 0$:

$$P_{ij}^{(n+m)} = \sum_{k \in S} p_{ik}^{(n)} p_{kj}^{(m)} \tag{2.5}$$

When $m = 0$, for $n = 2, 3, ...$

$$\begin{aligned} P_{ij}^{(n)} &= \sum_{k \in S} p_{ik}^{(n-1)} p_{kj} \\ &= \sum_{k \in S} p_{ik} p_{kj}^{(n-1)} \end{aligned} \tag{2.6}$$

Moreover, since $P_{ij}^{(n)} = P_{ij}^{(n-k)} P_{ij}^{(k)} = P_{ij}^{(n-1)} P_{ij}$, $P_{ij}^{(n)}$ can be seen as the n-th power of the P_{ij}. With the initial probability distribution $Pr\{X(0) = i\} = p_i$, the unconditional distribution can be determined as:

$$\begin{aligned} Pr\{X(t_n) = j\} &= \sum_i Pr\{X(t) = j|X(0) = i\} \\ &= \sum_i p_i p_{ij}^{(n)} \end{aligned} \tag{2.7}$$

In the area of biochemistry, the states contain the copy number of each species. Since the copy numbers are all discrete nature numbers, the states are also discrete. And because the reactions fire continuously and randomly, the time is continuous. Therefore, for this thesis, CTMCs are suitable for representing stochastic dynamics.

The Markov process is commonly described in 2 ways. One is the Chemical Master Equation (CME) (Section 2.3.2), which models the probability distribution function against time. The other is the Gillespie algorithm, which simulates the trajectory of each step [22] (Section 2.3.3). In the following sections, we introduce these two methods for expressing CTMCs.

2.3.2 Chemical Master Equation

The Chemical Master Equation (CME) is a method used to accurately model the stochastic dynamics of biochemical systems by taking the uncertainty of reactions firing into account through the use of transition probabilities, rather than deterministic equations [32].

To calculate the **master equation** $\frac{\mathrm{d}}{\mathrm{d}t}(Pr\{X(t)=j\})$, we can take a very small time interval $\mathrm{d}t$ in which only one reaction can occur. We then take the limit:

$$\frac{\mathrm{d}}{\mathrm{d}t}(Pr\{X(t)=j\}) = \lim_{x\to 0}\frac{Pr\{X(t+dt)=j\}-Pr\{X(t)=j\}}{\mathrm{d}t} \tag{2.8}$$

The result for $Pr\{X(t+dt)=j\}-Pr\{X(t)=j\}$ is the difference between the probability of entering the state j and leaving the state j during the time interval $\mathrm{d}t$.
Given that i is the set of all the states that can reach state j in $\mathrm{d}t$, k is the set of all the states that state j can reach in $\mathrm{d}t$, we get:

probability of entering the state j:

$$\sum_i\{\underbrace{\alpha_\tau(i)}_{propensity\ rate} Pr\{X(t)=i\}\}$$

or

$$Pr\{X(t+\mathrm{d}t)=j|X(t)=i\}Pr\{X(t)=i\}$$

probability of leaving the state j:

$$\sum_{\tau\in R}\{\underbrace{\alpha_\tau(j)}_{propensity\ rate} Pr\{X(t)=j\}\}$$

or

$$Pr\{X(t+\mathrm{d}t)=k|X(t)=j\}Pr\{X(t)=j\}$$

Therefore, we obtain the Chemical Master Equation:

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}(Pr\{X(t)=j\})\\ =&Pr\{X(t+\mathrm{d}t)=j|X(t)=i\}Pr\{X(t)=i\}-Pr\{X(t+\mathrm{d}t)=k|X(t)=j\}Pr\{X(t)=j\} && (2.9)\\ =&P_{ij}(t)Pr\{X(t)=i\}-P_{jk}(t)Pr\{X(t)=j\} && (2.10)\end{aligned}$$

From the equation above, we can see that CME generates one ODE to describe one state. The CME generates one ODE to describe one state. To describe the entire system, the CME needs to generate one ODE for each state. However, when the copy numbers or species count are high, the number of states and corresponding equations grows exponentially. Consequently, solving the CME becomes computationally infeasible in many cases [37].

2.3.3 Gillespie Algorithm

In cases where CME is difficult or impossible to solve either numerically or analytically [37], alternative methods are needed. The Gillespie Algorithm, introduced by Dan Gillespie in 1976 [23], provides a solution for simulating chemical reaction networks (CRNs) and is also known as the **Stochastic Simulation Algorithm (SSA)** [22].

The Gillespie algorithm employs **Monte Carlo Simulation** to describe the system [43], also referred to as **Monte Carlo Method**, a name generally believed to be derived from the casinos of Monte Carlo. It is a statistical technique that uses random samples to approximate the process [33]. The Monte Carlo method operates differently from the usual approach: intuitively, one often estimates random quantities in a deterministic way, whereas this method does exactly the opposite, using random quantities to provide an estimate of a deterministic quantity [44].
A simple example can help illustrate the approach:

Example 2.3 *Given a square of known area A_{square}, an irregular closed shape is in the square and the area of the shape needs to be found.*
The area can be derived by the Monte Carlo Simulation:

1. *Scatter small balls (much smaller than the square and the irregular shape) of the same size randomly over the square*

2. *Count the number of all balls on the square and the number of balls on the irregular shape*

3. *The area of the irregular shape A_{irr} can be calculated by $A_{irr} = \frac{\#balls\ on\ shape}{\#balls\ on\ square} \times A_{square}$*

4. *If the result is not precise enough, add more balls or/and repeat the above steps*

With the knowledge of Monte Carlo Simulation, the Gillespie algorithm works like this [24]:

1. Initialize the chemical concentrations for the CRN

2. Evaluate the propensity rate of each reaction occurring in this state

3. Use the Monte Carlo method to randomly select the time interval (infinitesimal) until the next event and then randomly select the reaction that will occur.

4. Record the new state and update the concentrations of species. Then return to step 2 and repeat the process or end the simulation.

The Gillespie algorithm simulates the stochastic dynamics of the system by sampling reaction events based on their propensity rates. It generates a trajectory of the system by iteratively updating the chemical concentrations according to the selected reactions. The more simulations are performed, the more accurate the Gillespie algorithm becomes. In fact, if all possible trajectories are simulated, the Gillespie algorithm is an exact method for representing CRNs [53] [23].

In *Figure 2.1*, the results of two independent runs of the Gillespie Algorithm simulating a simple reaction $S \xrightarrow{c} \emptyset$ are compared with the solution obtained by solving the corresponding ordinary differential equation (ODE), with reaction rate coefficient equal to 1 and initial copy number equal to 100. The blue and red stair-step curves are the 2 independent simulated results, while the smooth black curve is obtained by solving deterministically.

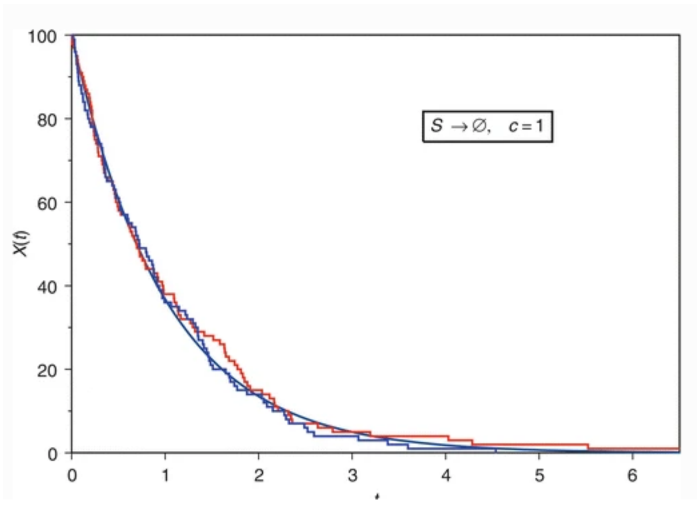


Figure 2.1: Comparison of SSA and ODE Solution for a Simple Reaction

It is evident that the accuracy of the Gillespie Algorithm increases with the number of simulations performed. However, achieving perfect accuracy requires a high computational cost, particularly when the system involves many species or high copy numbers. Fortunately, the advancement of computer power has alleviated this problem [8].

2.3.4 Linear Noise Approximation

While high-performance computers have mitigated the computational challenges of the Gillespie Algorithm, they may not always be available. In situations where high accuracy is not necessary, an alternative method for stochastic modelling is the Linear Noise Approximation (LNA), also known as the **Central Limit Approximation (CLA)** [37].

The LNA approximates the stochastic behaviour of a chemical reaction network (CRN) by linearizing (using a first-order expansion) the CME. It assumes that the mean and variance of species against time can be modelled as a Gaussian process. The LNA can be represented as follows:

$$Y = \underbrace{\Omega \cdot \Phi}_{mean} + \underbrace{\sqrt{\Omega} \cdot Z}_{covariance} \tag{2.11}$$

where Ω is the volume of a CRS $C = (\Lambda, R, x_0)$, Φ is the solution of the *Equation 2.1* with initial condition $\Phi(0) = \frac{x_0}{\Omega}$, and Z is a zero-mean Gaussian process whose variance C[Z(t)] is described by the following differential equation:

$$\frac{\mathrm{d}C[Z(t)]}{\mathrm{d}t} = J_F\big(\Phi(t)\big) \cdot C[Z(t)] + C[Z(t)] \cdot J_F\big(\Phi(t)\big)^T + W\big(\Phi(t)\big) \tag{2.12}$$

where $J_F\big(\Phi(t)\big)$ is the Jacobian of $F\big(\Phi(t)\big)$, $J_F\big(\Phi(t)\big)^T$ is the transpose version, $C[Z(t)]$ is the correlation matrix containing variance and covariances of all species against themselves and each other, and $W\big(\Phi(t)\big)$ is the sum of the square of net change times the propensity rate [11]:

$$W\big(\Phi(t)\big) = \sum_{\tau \in R} v_\tau^T v_\tau \alpha_\tau\big(\Phi(t)\big) \tag{2.13}$$

The first part of *Equation 2.11* determines the expectation value of LNA: $E[Y(t)] = \Omega \cdot \Phi(t)$, the second part determines the covariance matrix of LNA: $C[Y(t)] = \Omega \cdot C[Z(t)]$ [11]. By calculating the mean and variance values, the LNA provides approximations for the mean and variance values obtained from the CME. Research has shown that the LNA can yield accurate approximations even when fluctuations are large [14].

Consider the case of steady state, we can let *Equation 2.12* equal to 0, and then the matrix $C[Z(t)]$ can be solved and thus all the variance values are revealed. The mean value can be solved by making *Equation 2.1* equal to 0. Inserting these values back to *Equation 2.11*, the corresponding LNA of the CME can be obtained.

Below we give a simple example of how it works.

Example 2.4 *Given a simple system below, we want to get the corresponding mean value and the covariance of LNA at a steady state.*

$(reactions)$	$(propensity\ rates)$
$\emptyset \xrightarrow{1} a$	1
$a \xrightarrow{1} \emptyset$	$\Phi_a(t)$
$a \xrightarrow{1} a+b$	$\Phi_a(t)$
$b \xrightarrow{1} \emptyset$	$\Phi_b(t)$

The rate equation is:

$$\dot{\Phi}_a(t) = 1 - \Phi_a(t) \qquad \dot{\Phi}_b(t) = \Phi_a(t) - \Phi_b(t)$$

Let the rate equation equal to 0, we can get the mean value $E[\Phi_a] = 1 \quad E[\Phi_b] = 1$.
The Jacobian of the rate equation is:

$$J_F(\Phi) = \begin{bmatrix} -1 & 0 \\ 1 & -1 \end{bmatrix}$$

And the covariance matrix is:

$$C[Z] = \begin{bmatrix} V_a(t) & C[a,b] \\ C[a,b] & V_b(t) \end{bmatrix}$$

With

$$v_1^T = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ v_2^T = \begin{bmatrix} -1 \\ 0 \end{bmatrix},\ v_3^T = \begin{bmatrix} 0 \\ 1 \end{bmatrix},\ v_4^T = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

we can get

$$W(\Phi) = \begin{bmatrix} 1+\Phi_a & 0 \\ 0 & \Phi_a + \Phi_b \end{bmatrix}$$

Finally, solve

$$J_F(\Phi) \cdot C[Z] + C[Z] \cdot J_F(\Phi)^T + W(\Phi) = 0$$

we can get

$$C[Z] = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$

From the matrix, we can extract that $V_a(t) = 1$ and $V_b(t) = 1.5$.

And we simulate the system in software, where μ is the mean value and σ^2 represents the variance (see in Figure 2.2). From the figure, we can see that $\sigma_a^2 = 1$ and $\sigma_b^2 = 1.5$, verifying the calculated variance value above.

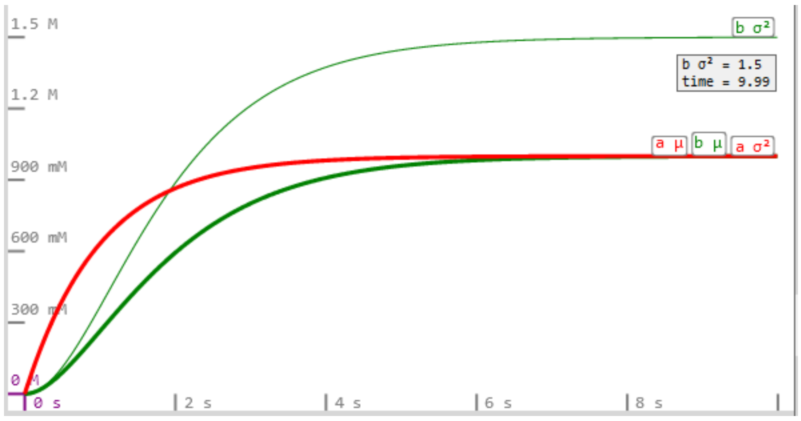


Figure 2.2: LNA Plot in Kaemika

Chapter 3

Biochemical Differentiators

In recent years, numerous biological differentiators have been studied and designed. In this chapter, we introduce and compare two recent differentiators, namely **Biomolecular Signal Differentiators (BioSD)** and **Transfer Function Differentiators (TFD)**. Our goal is to identify an accurate differentiator that is easy to implement, tune, and test. These distinct approaches offer diverse strategies for calculating derivatives, each with its own advantages and applications. After conducting a thorough comparison, we selected BioSD as the study object of our analysis.

3.1 BIOMOLECULAR SIGNAL DIFFERENTIATORS

In 2021, Alexis et al. introduced a novel set of architectures called Biomolecular Signal Differentiators (BioSD) for calculating the time derivative [2]. The BioSD consists of three differentiators: **BioSD-I**, **BioSD-II**, and **BioSD-III**, as illustrated in *Figure 3.1*. The complexity of the BioSD, including the number of auxiliary species, reactions, and parameters, increases with the ordinal numbers in their names (see *Figure 3.1* below).

3.1.1 System Structures

This section describes the system structures of BioSD. In these architectures, the input signal is represented by species U, and the output signal is represented by X. Additional species are introduced as auxiliary species to assist in the differential calculations.

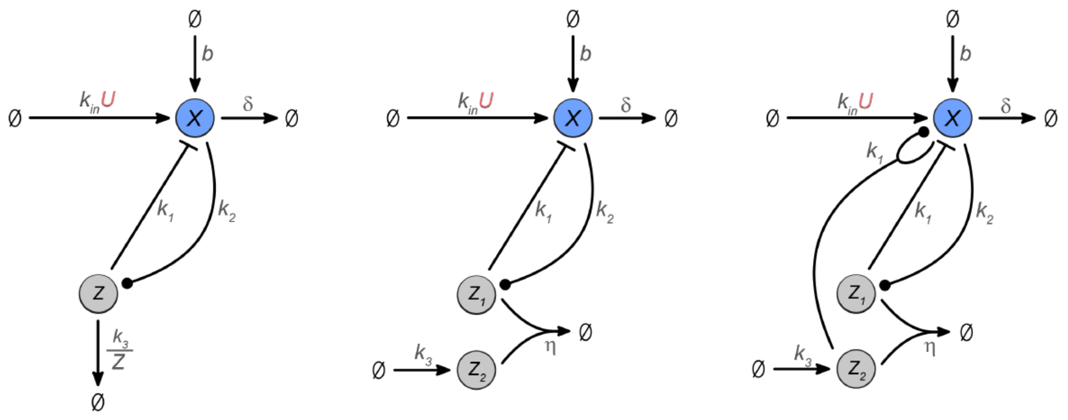


Figure 3.1: Architecture of BioSD
From left to right: BioSD-I, BioSD-II, and BioSD-III. The graphical representation denotes various types of reactions: (∅) represents the empty set, (→) indicates the transformation of reactants into products following the direction of the arrow, (–•) describes catalytic production where products are generated without consuming the reactants (denoted by •). In some cases, there may be two reactants in this reaction, with one of the reactants also being a product, as seen in BioSD-III. (—⊣) represents the catalytic inhibition reaction, where ⊣ indicates the reactants being diminished.

BioSD-I consists of the following reactions:

$$\begin{gathered}\emptyset \xrightarrow{k_{in}U} X, \quad \emptyset \xrightarrow{b} X, \quad X \xrightarrow{k_2} X+Z \\ X+Z \xrightarrow{k_1} Z, \quad X \xrightarrow{\delta} \emptyset, \quad Z \xrightarrow{k_3/Z} \emptyset\end{gathered} \tag{3.1}$$

In BioSD-I, the input species U acts as a catalyst for the reaction rather than being directly involved. The generation of auxiliary species Z is catalyzed by X, and Z catalyzes the diminish of X in turn. Additionally, species X and Z directly diminish, simulating species ageing in real-world scenarios.
Using the Law of Mass Action, the corresponding rate equations for BioSD-I are as follows:

$$\dot{X} = k_{in}U + b - k_1XZ - \delta X \tag{3.2a}$$
$$\dot{Z} = k_2X - k_3 \tag{3.2b}$$

BioSD-II is defined by the following reactions:

$$\begin{gathered}\emptyset \xrightarrow{k_{in}U} X, \quad \emptyset \xrightarrow{b} X, \quad X \xrightarrow{k_2} X+Z_1 \\ X+Z_1 \xrightarrow{k_1} Z_1, \quad \emptyset \xrightarrow{k_3} Z_2, \quad Z_1+Z_2 \xrightarrow{\eta} \emptyset, \quad X \xrightarrow{\delta} \emptyset\end{gathered} \tag{3.3}$$

From *Figure 3.1*, we can see that the structures of BioSD-I and BioSD-II are similar. In BioSD-II, Z is replaced with Z_1, and an additional auxiliary species Z_2 is introduced. The reaction between Z_1 and Z_2 leads to their annihilation.
BioSD-II can be described by the following ODEs:

$$\dot{X} = k_{in}U + b - k_1 X Z_1 - \delta X \tag{3.4a}$$

$$\dot{Z}_1 = k_2 X - \eta Z_1 Z_2 \tag{3.4b}$$

$$\dot{Z}_2 = k_3 - \eta Z_1 Z_2 \tag{3.4c}$$

BioSD-II is essentially a variant of BioSD-I, where Z_2 is generated at a rate of k_3 and subsequently combines with Z_1, leading to the disappearance of Z_1. When the value of η is set to be large enough, we can consider that $\dot{Z}_2 = 0$ and k_3 is equal to $\eta Z_1 Z_2$. Thus, $\dot{Z}_1$ becomes the same as the $\dot{Z}$ in BioSD-I, decreasing at a rate of k_3.

The CRN of **BioSD-III** is described as follows:

$$\begin{gathered}\emptyset \xrightarrow{k_{in}U} X, \quad \emptyset \xrightarrow{b} X, \quad X \xrightarrow{k_2} X + Z_1 \\ X + Z_1 \xrightarrow{k_1} Z_1, \quad \emptyset \xrightarrow{k_3} Z_2, \quad X + Z_2 \xrightarrow{k_1} X + X + Z_2 \\ Z_1 + Z_2 \xrightarrow{\eta} \emptyset, \quad X \xrightarrow{\delta} \emptyset \end{gathered} \tag{3.5}$$

BioSD-III is an extension of BioSD-II, without introducing new auxiliary species but including a double catalytic production reaction where X and Z_2 catalyze the production of X. However, the number of reactions is considerably increased.
The following ODE model can be generated from BioSD-III:

$$\begin{aligned}\dot{X} &= k_{in}U + b - k_1 X Z_1 + k_1 X Z_2 - \delta X \\ &= k_{in}U + b - k_1 X (Z_1 - Z_2) - \delta X \end{aligned} \tag{3.6a}$$

$$\dot{Z}_1 = k_2 X - \eta Z_1 Z_2 \tag{3.6b}$$

$$\dot{Z}_2 = k_3 - \eta Z_1 Z_2 \tag{3.6c}$$

BioSD-III is a modification of BioSD-II. However, the underlying principles differ. In BioSD-II, the equation of X looks like this:

$$\dot{X} = k_{in}U + b - k_1 X \int (k_2 X - k_3) - \delta X$$

while in BioSD-III, *Equation 3.6a* actually looks like this:

$$\begin{aligned}\dot{X} &= k_{in}U + b - k_1X(Z_1 - Z_2) - \delta X\\ &= k_{in}U + b - k_1X\int\big((k_2X - \eta Z_1Z_2) - (k3 - \eta Z_1Z_2)\big) - \delta X\\ &= k_{in}U + b - k_1X\int(k_2X - k_3) - \delta X\end{aligned}$$

Therefore, while the reactions in BioSD-III and BioSD-II are very similar, a large value of η is not necessary for BioSD-III.

3.1.2 System Design Principle

Since both U and X represent the concentration of species, BioSD can only accept non-negative input signals and produce non-negative output signals. However, it is important to note that the time derivative of a non-negative signal can still be negative in certain cases. To address this issue, Alexis et al. introduced a bias term to the calculated result, which takes on the steady-state value of X. By incorporating this bias, BioSD ensures that the resulting outputs remain non-negative.

The steady-state value of X can be determined by setting *Equation 3.2b* of BioSD-I, *Equation 3.4b* and *3.4c* of BioSD-II, *Equation 3.6b* and *3.6c* of BioSD-III to 0:

$$0 = k_2X - k_3$$

$$\begin{aligned}0 &= k_2X - \eta Z_1Z_2\\ 0 &= k_3 - \eta Z_1Z_2\end{aligned}$$

$$\begin{aligned}0 &= k_2X - \eta Z_1Z_2\\ 0 &= k_3 - \eta Z_1Z_2\end{aligned}$$

The steady-state value of X in all typologies can be easily calculated as k_3/k_2, which serves as the bias term.

When employing BioSD for derivative calculations, our focus is primarily on scenarios that are in close proximity to their respective steady states. To obtain the derivatives near these steady states, a linearization process is performed on the rate equations. In this study, we utilize BioSD-I as our illustrative example (the processes for the other two differentiators are the same). The linearized representation of the system dynamics is expressed as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -\frac{k_2(k_{in}U^*+b)}{k3} & -\frac{k_1 k_3}{k_2} \\ k_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} k_{in} \\ 0 \end{bmatrix} u$$

where the variables $u = U - U^*$, $X = X - X^*$,$z = Z - Z^*$ represents the small deviations around the equilibrium points (U^*,X^*,Z^*).

Then we apply variable transformations according to the section *Behaviour Analysis of Biomolecular Signal Differentiators* in [2], transforming the equation into the second-order differential equation and finally getting the following expression:

$$x = \frac{k_{in}}{k_1 k_3} \dot{u} \tag{3.7}$$

Following these calculations, the researchers made an observation that the time derivatives can be approximated as:

$$X = \underbrace{\frac{k_{in}}{k_1 k_3}}_{amplificatio} \dot{U} + \underbrace{\frac{k_3}{k_2}}_{bias} \tag{3.8}$$

If the parameters are appropriately set, specifically $k_{in} = k_1 k_3$, the equation simplifies to:

$$X = \dot{U} + bias \tag{3.9}$$

Suppose that the input signal U can be divided into two parts:

$$U = \underbrace{U^*}_{constant} + \underbrace{U^{TV}}_{time-varying} \tag{3.10}$$

A sufficient but not necessary condition to ensure accurate results from the BioSD can be expressed as:

$$\varepsilon = \frac{k_2^2}{k_1 k_3^3}(k_{in}U^* + b)^2 \ll 1 \tag{3.11}$$

3.1.3 System Performance

In the research [2], the authors utilized a simple linear signal and a basic design of **Antithetic Integral Feedback (AIF)** [7] (which will be discussed in subsection 4.5.2) t**o** evaluate the performance of BioSD.

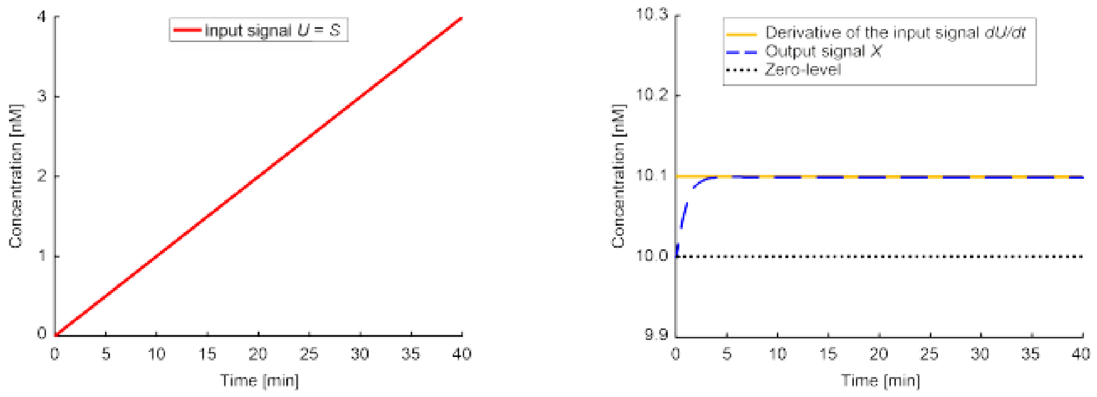


Figure 3.2: Input and Output of BioSD-I on Linear Signal

The CRN for the linear signal is given by:

$$\emptyset \xrightarrow{v_b} U, \quad U \xrightarrow{v_d} \emptyset$$

Here, the reaction rates are specified as $v_b = 0.1$ and $v_d = 0.001$.
The BioSD parameters are set as follows: $k_{in} = 10$, $b = 100$, $k_1 = 1$, $k_2 = 1$, $k_3 = 10$, and $\delta = 0.5$.
Consequently, the following values are obtained:

$$amplification = \frac{k_{in}}{k_1 k_3} = \frac{10}{1 \times 10} = 1$$
$$bias = \frac{k_3}{k_2} = \frac{10}{1} = 10$$
$$U^* = 0$$
$$\varepsilon = \frac{k_2^2}{k_1 k_3^3}(k_{in}U^* + b)^2 = 10 \gg 1$$

Here we only present the results for BioSD-I, as the other two differentiators yield identical outcomes. The results are depicted in *Figure 3.2*, which shows the input and output of BioSD-I for the linear signal.
From the figure, it is evident that despite the large value of ε exceeding 1, the BioSD still accurately matches the calculated derivative, except during the initial period where the BioSD undergoes a "warm-up" phase.

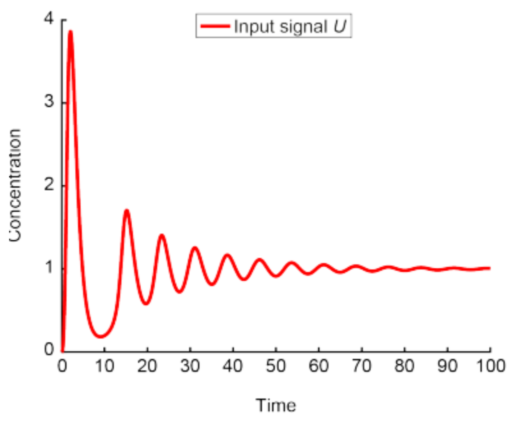
The CRN for the Antithetic Integral Feedback is described by the following reactions:

$$\emptyset \xrightarrow{v_1} C_1, \quad C_1 \xrightarrow{v_2} C_1 + Y_1, \quad Y_1 \xrightarrow{v_3} Y_1 + Y_2$$
$$Y_2 \xrightarrow{v_4} Y_2 + C_2, \quad C_1 + C_2 \xrightarrow{v_5} \emptyset, \quad Y_1 \xrightarrow{v_6} \emptyset, \quad Y_2 \xrightarrow{v_7} \emptyset$$

Here, Y_2 represents the input signal U to the BioSD, and the reaction rates are specified as $v_1 = v_2 = v_4 = 2$, $v_3 = 4$, $v_5 = 12$, $v_6 = v_7 = 1$.
The parameters of BioSD-I are set as follows: $k_{in} = k_3 = b = 100$, $k_1 = k_2 = 1$, $\delta = 0.5$, $\eta = 1000$. Consequently, the following values are obtained:

$$amplification = \frac{k_{in}}{k_1 k_3} = \frac{100}{1 \times 100} = 1$$
$$bias = \frac{k_3}{k_2} = \frac{100}{1} = 100$$
$$U^* = 0$$
$$\varepsilon = \frac{k_2^2}{k_1 k_3^3}(k_{in}U^* + b)^2 = 0.01 \ll 1$$

Same as the previous experiment, only the results for BioSD-I are presented below in *Figure 3.3*. From the left graph, we can observe that the input signal undergoes rapid and intense oscillations, which poses a greater challenge for the BioSD. However, in the right graph, the yellow curve representing the reference derivative is almost indistinguishable, as the output signal X closely matches the $\mathrm{d}U/\mathrm{d}t$ curve. This indicates that BioSD produces highly accurate results for this example.



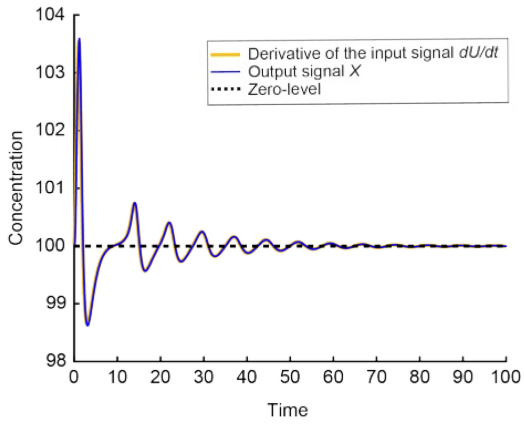


Figure 3.3: Input and Outputs of BioSD-I on AIF
Time scale is in minutes, concentration unit is nM ($1nM = 1 \cdot 10^{-6}M$)

However, if the *premise 3.11* is not satisfied for this case, the accuracy of BioSD may be compromised, as illustrated in *Figure 3.4*. It can be observed from the figure that BioSD-I exhibits some delays in tracking the time derivative. Although the output signal X cannot closely follow the curve during the initial surge and plunge of the time derivative, the changing trends are still preserved.

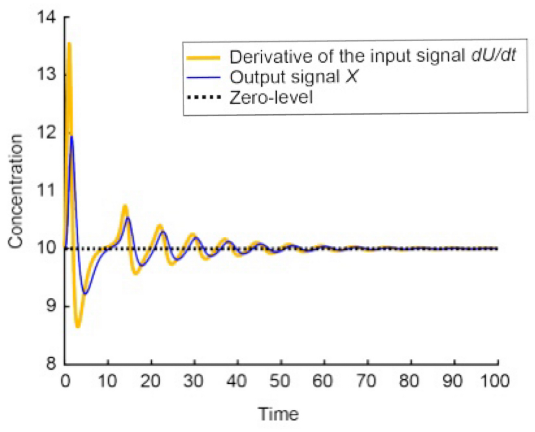


Figure 3.4: Poor Performance of BioSD-I on AIF
Time scale is in minutes, and the concentration unit is nM ($1\text{nM} = 1 \times 10^{-6}\text{M}$). k_{in} and k_3 have been changed to 10, resulting in bias = 10 and $\varepsilon = 10 \gg 1$.

3.2 Transfer Function Differentiator

In 2017, Halter et al. proposed a novel design of differentiator based on the concept of Laplace transformation, which enables the transformation between the time domain and the frequency domain [28]. We call it Transfer Function Differentiator (TFD).

In the frequency domain, the ideal differentiator is represented by a transfer function of $G(s) = s$, where s denotes the Laplace variable. However, this ideal differentiator is non-causal, meaning that its output depends on future inputs, making it physically unrealizable. To overcome this limitation, Halter et al. approximated the ideal differentiator by adding a low-pass filter. The transfer function of the approximated differentiator is given by $G(s) = \frac{Ks}{s+K}$, where K represents the bandwidth of the filter. This transfer function can be realized through the circuits illustrated below.

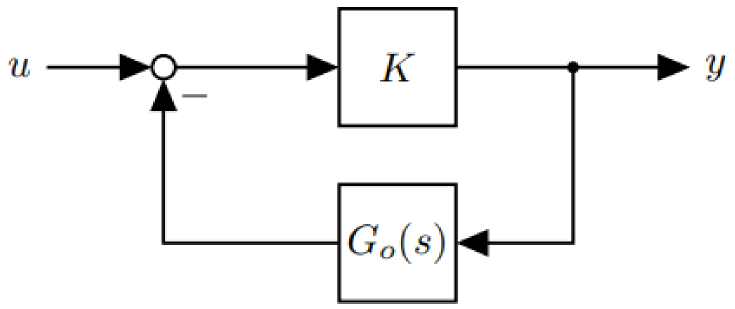


Figure 3.5: Approximated Differentiator Circuit [28]

From the circuit diagram, the transfer function can be derived as this:

$$\begin{aligned}
&Y = K(U - G_o(s)Y) \\
\Longrightarrow &Y + KG_o(s)Y = KU \\
\Longrightarrow &(1 + KG_o(s))Y = KU \\
\Longrightarrow &\frac{Y}{U} = G(s) = \frac{K}{1 + G_o(s)K}
\end{aligned}$$

By assuming a large K and choosing the integrator $\frac{1}{s}$ for $G_o(s)$, the transfer function becomes:

$$\frac{Y}{U} = G(s) = \frac{K}{1 + \frac{K}{s}} = \frac{Ks}{s + K} \approx s$$

Therefore, the design of the transfer function differentiator can be implemented using a signal difference, a gain, and an integrator.

3.2.1 System Design

The process of turning the design into a genetic differentiator is guided by the Oishi and Klavins' approach [41]. By using the molecular quantities of DNA, mRNA, and proteins to represent the signal levels, the implementation of this differentiator design can be achieved. *Figure 3.6* illustrates the system architecture, where the concentration of mRNA denotes the differential result y.

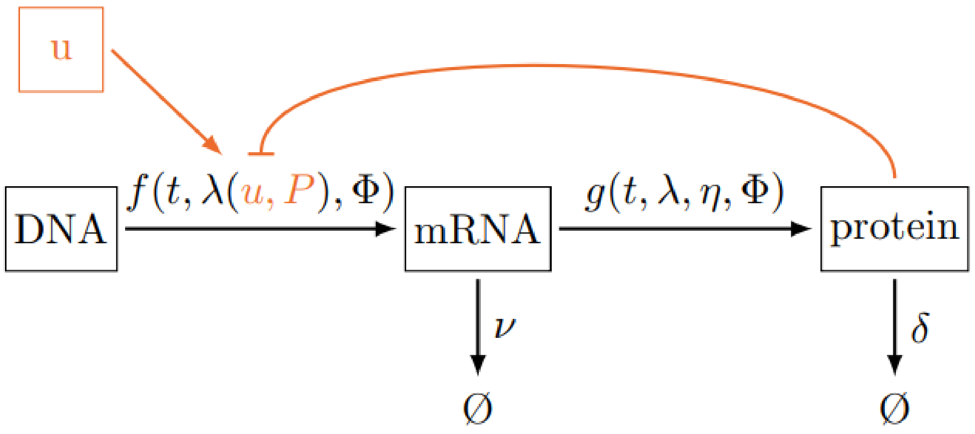


Figure 3.6: System Design of TFD
$f(t,\lambda(u,P),\Phi)$ is the DNA transcription rate, where t is time, λ is the transcription initiation rate, u is the input, P is the amount of Protein and Φ is the environmental parameters; ν is the degradation rate of mRNA; $g(t,\lambda,\eta,\Phi)$ is the mRNA translation rate, η is the translation initiation rate; δ is the degradation rate of protein.

The CRN of the system can be described as follows:

$$\begin{aligned}
&DNA \xrightarrow{f/DNA} DNA + mRNA\\
&mRNA \xrightarrow{\nu} \emptyset\\
&mRNA \xrightarrow{g/mRNA} mRNA + protein\\
&protein \xrightarrow{\delta} \emptyset
\end{aligned}$$

The process of protein annihilating the generation of mRNA is realized by the λ function, so it is not marked separately in the above CRN.

The implementation of the three functional parts (signal difference, gain, and integrator) as well as the representation of positive and negative signals are elaborated as follows:

- **Signal Difference**:
  To achieve negative feedback, a protein is chosen that inhibits DNA transcription, with the degree of inhibition determined by the protein concentration.

- **Gain**:
  The amplification of a signal can be achieved by a process of production and degradation. The researchers aimed to achieve rapid amplification, allowing the system to reach a steady state within a reasonable time. Therefore, the production and degrada-

tion rates must not be too different in order of magnitude. For this reason, mRNA is chosen to achieve this function.
From *Figure 3.6* and the CRN, we can get

$$\dot{\Phi}_{mRNA} = f - v \cdot \Phi_{mRNA}$$

By choosing a sufficiently large value for v and assuming a linear relationship between f and u ($f \approx \alpha u$), the steady state of mRNA becomes:

$$\Phi_{mRNA_{ss}} = \frac{f}{v} \approx \frac{\alpha u}{v}$$

Hence, the gain K can be expressed as:

$$K = \frac{\alpha}{v}$$

- **Integrator**:
  Similar to the gain function, integration can be achieved through production and degradation processes. However, in this case, the annihilation rate is chosen to be as low as possible, effectively behaving as an integrator. This is exactly what happens with the degradation rate of a protein. Thus, protein is chosen for this function.

- **Positive and Negative Signals**:
  In order to make this differentiator more versatile, the input u and output y are not restricted to non-negative numbers. To accommodate both positive and negative signals, the dual rail method is employed. The signal is split into positive and negative components (e.g., $u = u^+ - u^-$).
  To save input and output signal resources, Oishi and Klavins generate two additional species for each functional block. The overall layout of the system is depicted in the graph below, where G represents the DNA, P represents the protein and M is the mRNA.

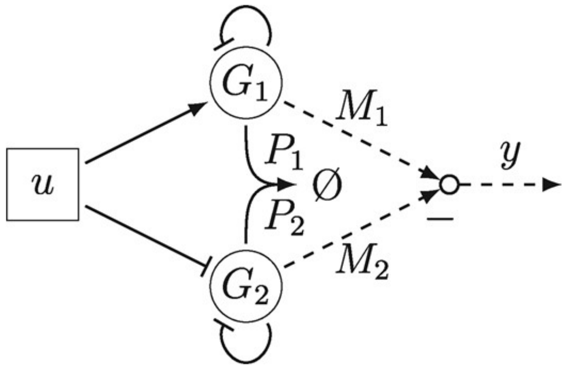


Figure 3.7: Dual Rail Representation of TFD [27]

From *Figure 3.7,* the detailed dynamic of the system can be described by the CME below (the notations here are the same as in the description of *Figure 3.6*):

$$\begin{aligned}
\dot{M}_1 &= fG_1 \cdot k(u - P_1) - \nu_1 M_1 \\
\dot{P}_1 &= gM_1 - \delta_1 P_1 - \delta_{12} P_1 P_2 \\
\dot{M}_2 &= fG_2 \cdot k(k_0 - u - P_2) - \nu_2 M_2 \\
\dot{P}_2 &= gM_2 - \delta_2 P_2 - \delta_{12} P_1 P_2
\end{aligned}$$

with the function

$$k(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

3.2.2 System Performance

The feasibility of the Transfer Function Differentiator approach is validated through *in silico* experiments. Two different signal inputs are tested, and the results are presented in*Figure 3.8*.

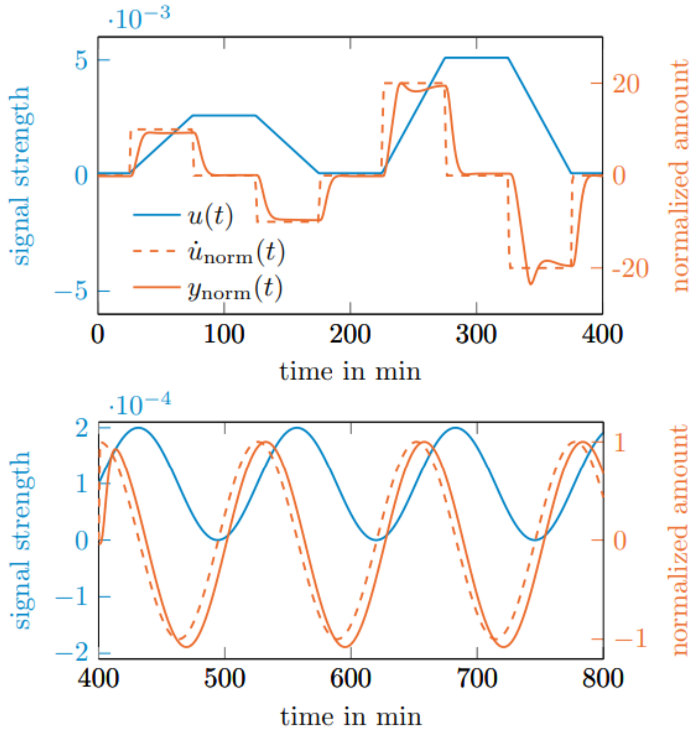


Figure 3.8: Test Results of Transfer Function Differentiator

From the graphs, it can be observed that there are some delays in the outputs $y_{\text{norm}}(t)$ of TFD compared to the calculated values $\dot{u}_{\text{norm}}(t)$. These delays are attributed to the time required for transcription and translation processes within the system. The accuracy of the outputs is also dependent on the type of input signals. The outputs of the sinusoidal waveform closely follow the trends of the calculated results, while the outputs of the first input waveform exhibit some overshoots, undershoots and slightly lower accuracy. But in general, the outputs remain reasonably close to the calculated results.

Furthermore, additional experiments demonstrated the robustness of the Transfer Function Differentiator under a certain degree of parameter uncertainty [28].

It is important to note that the method assumes a large value for the gain K. Generally, a larger gain leads to more accurate differentiation. However, to maintain the behaviour of mRNA as a gain element, it is crucial that the production rate and degradation rate do not differ significantly. This constraint imposes a constraint on the value of K, as a balanced production and degradation rate is necessary for the proper operation of the system.

3.3 COMPARISON

Both the BioSD and TFD have their own advantages and drawbacks, and for the purpose of this thesis, the goal is to select a differentiator that offers accuracy while being easy to implement, tune, and test.

First, we compare their performance. TFD exhibits noticeable delays in its output, whereas BioSD has a negligible delay. The accuracy of TFD is highly dependent on the signal type, and it tends to exhibit errors even with relatively slow square wave signals. On the other hand, the accuracy of BioSD may seem good, but the related paper only tested it with one "complex" signal. Therefore, a wider range of input signals is needed to be determined its general accuracy.

Next, we consider the convenience of the experiment. BioSD offers good modularity and is easy to implement. Conversely, the implementation of reaction rates f and g in TFD can be challenging. The good point of TFD is that the tuning process is rather easy since it only has one parameter K, whereas the workload of BioSD to find the optimal solution is no less. As for the test time, the calculation process of TFD is quite slow, with one square wave test taking 100 minutes, whereas BioSD exhibits faster response times. Additionally, TFD can handle both positive and negative signals, while BioSD is limited to non-negative inputs.

Considering these factors, BioSD appears to have advantages over TFD in terms of implementation ease, response time, and modularity. Therefore, BioSD is chosen as the selected differentiator for this thesis. However, further experimentation and evaluation are needed to fully assess the general accuracy and performance of BioSD across a wider range of input signals.

Chapter 4

Simulation Preparation

As previously mentioned, biomolecular systems are subjected to both external and internal noise sources. Within this thesis, our focus centres on exploring the impact of internal noise.

Understanding the noise in biochemical networks is pivotal for our research. Thus, our exploration begins by investigating the effects of noise. While in vitro experiments lie beyond the scope of this work, our investigation into noise behaviour primarily relies on computer simulations. Consequently, we proceed to review and select the tools and methods for our analysis.

We undertake a comprehensive review of diverse methods for evaluating noise. Then we research noise suppression methods, aiming to identify effective strategies to mitigate noise and enhance the reliability of our biochemical system. We also discuss two software tools, Kaemika and ERODE, and undertake a comparative analysis to determine the most suitable choice. Additionally, we carefully choose input signals for the tests, including hyperbolic tangent, Antithetic Integral Feedback (AIF), and sinusoidal wave. These selected test signals are then input into BioSD, the results of which are compared with the chosen reference derivative, allowing us to assess their accuracy. This comprehensive examination of noise-related aspects serves as a crucial stepping stone for the subsequent experimental phase of our study.

4.1 Effects of Noise in Biochemical Networks

The role of noise in biochemical systems has long been a subject of study. Intuitively, noise has been considered detrimental, compromising the accuracy of signal monitoring and suppressing the functionality of biochemical switches [31]. Additionally, uncontrolled noise has the potential to significantly disrupt essential cellular processes and even contribute to the development of diseases, including cancer [15].

However, it is essential to acknowledge that the impact of noise on biochemical systems is not uniformly negative. In certain contexts, noise can play a constructive role, enabling system regulation and facilitating probabilistic cellular differentiation or evolution processes, ultimately leading to Robust Perfect Adaption (RPA) [13]. And contrary to common belief, research indicates that the presence of input noise doesn't inevitably lead to an increase in output variance [31].

Furthermore, we possess the capability to manipulate and harness noise to our advantage. Mechanisms that modulate noise, such as those increasing heterogeneity within systems, can be advantageous in coping with environmental stress, diversifying antibodies, and promoting functional heterogeneity [15].

Consequently, the effects of noise in biochemical systems encompass both positive and negative aspects. It has the potential to either enhance or hinder system functionality, depending on the noise level and the environment. When noise proves beneficial, we can exploit it to our advantage. Conversely, when noise poses a threat, regulatory measures or alternative tools can be deployed to mitigate its adverse effects.

In this thesis, BioSD serves as a tool facilitating the derivation of derivative information from biochemical systems. Therefore, it is important to avoid excessive noise in the outputs, thereby ensuring accuracy and clarity. As such, noise is regarded as a detrimental factor in this context and should be reduced.

4.2 Noise Evaluation Methods

When evaluating noise, simple measurement of noise is often inadequate as it overlooks the varied characteristics exhibited by different signals. Thus, a comprehensive assessment of noise levels necessitates the application of suitable methods. In this section, we discuss three commonly employed techniques: **Signal-to-Noise Ratio (SNR)**, **Root Mean Square (RMS)**, and the **Fano Factor**.

- **Signal-to-Noise Ration (SNR)**: The SNR is a widely utilized metric in electromagnetic systems to evaluate signal quality. It quantifies the ratio between the level of a desired signal and the level of noise within the system. This concept has also been extended to the field of biology, as highlighted in [5], where it measures the degree of noise deviation from the ideal representation. In this context, the ideal case is interpreted as "true," and the presence of noise is interpreted as "false". The SNR can then be calculated using the following equation:

$$SNR_{dB} = 20\log_{10}\left(\frac{\left|\log_{10}(\mu_{\text{true}}/\mu_{\text{false}})\right|}{2\log_{10}(\sigma)}\right)$$

where μ_{true} represents the value of the true signal, μ_{false} represents the geometric mean value of the noise, and σ denotes the geometric standard deviation for both states.

- **Root Mean Square (RMS)**: RMS is a measure calculated as the square root of the mean value of the squared values of each signal. Mathematically, it can be expressed as:

$$RMS(t) = \sqrt{\frac{x_1^2 + x_2^2 + \cdots + x_n^2}{n}}$$

where n denotes the number of signals, and x_i $(i \in n)$ represents the value of each signal. When all x_i values correspond to noise, the RMS value represents the noise level at the given moment. A lower RMS indicates better signal quality. Additionally, RMS can be utilized in the calculation of SNR using the following relationship:

$$SNR = \left(\frac{RMS_{\text{signal}}}{RMS_{\text{noise}}}\right)^2$$

- **Fano Factor**: The Fano factor is a commonly used measure for analyzing noise [18]. It is first introduced by U. Fano in 1947 to quantify the level of fluctuations of ions [16]. Since then, it has found applications in various fields, including computational neuroscience [47], nuclear physics [36], and synthetic biology [19].
The Fano factor is defined as the ratio of the variance to the mean value, expressed as

$$Fano = \frac{variance}{mean}$$

This factor serves as a valuable criterion for comparing noise levels across different systems, as it normalizes the noise by dividing it by the mean value. Moreover, the Fano factor can be used to assess the reliability of a system, with smaller values indicating greater stability and reliability.
In the case of a Poisson Process, which counts the number of random events occurring in a fixed time interval, the Fano factor has a special interpretation. Under this circumstance, the variance is equal to the mean value, resulting in a Fano factor of 1. Fluctuations characterized by a Fano factor of 1 are referred to as being at the **Poisson**

level.

All of these methods are effective for noise evaluation, and determining a clear winner among them is challenging. Therefore, we continue with selecting a suitable software tool and then choose the noise analysis method based on the convenience of the software.

4.3 Noise Suppression Methods

In the context of utilizing BioSD as a tool for calculating signal derivatives, maintaining the accuracy and reliability of the biochemical system is important. Excessive noise in BioSD's outputs can compromise the integrity of the derived information. To deal with this issue, various noise suppression methods can be employed.

This section reviews two approaches for noise reduction: **Noise Filters** and **Biological PID Controller**. By exploring these noise suppression methods, we aim to gain insights into their applicability in biochemical systems.

4.3.1 Noise Filters

Inspired by the principles of low-pass filters, Laurenti et al. introduced a series of innovative molecular filters designed to effectively mitigate both extrinsic and intrinsic noises within biochemical systems [38]. These noise filters include a **Linear Filter**, an **Annihilation Module**, and an **Annihilation Filter**.

The **Linear Filter** ($\mathscr{F}$) is the simplest one among the three. It is represented by the following CRN in which an input species (A) reacts to produce an output species (B):

$$\mathscr{F} : A \xrightarrow{k_1} A + B \quad B \xrightarrow{k_2} \emptyset \tag{4.1}$$

The linear filter effectively reduces the noise level to the Poisson level for positively correlated elements. With the incorporation of a feedback loop, it can further suppress noise below the Poisson level. It is important to note that the linear filter is limited to uni-molecular reactions, which leads to an interest in exploring nonlinear filters.

The **Annihilation Module** ($\mathscr{M}$) utilizes second-order reactions to improve noise filtering. The corresponding CRN is as follows:

$$\mathscr{M} : A \xrightarrow{r_1} A + B + C \quad B + C \xrightarrow{r_2} \emptyset \tag{4.2}$$

Within this module, species A acts as the input, and species B and C are substantially the same, making either one suitable as the output. Through the co-expression of B and C, the low-frequency portion of the input undergoes annihilation, thus reducing noise without introducing any delay. To maintain adherence to the expectation value of the input ($E[A]$), the reaction rate coefficients must satisfy the condition $\frac{r_1}{r_2} = E[A]$. Under this constraint, the Fano factor consistently remains below the Poisson level. However, when the expected value of A is not constant, the output fails to accurately track the input value.

The **Annihilation Filter** ($\mathscr{A}$) addresses this limitation and can consistently track the input, regardless of parameter values. It is the most complex one among the three filters. The corresponding CRN is defined as:

$$\begin{aligned} \mathscr{A}: A \xrightarrow{r_1} A+B+D \quad D \xrightarrow{r_1} E \quad B+E \xrightarrow{r_2} \emptyset \\ B+A \xrightarrow{r_3} A+C \quad E+C \xrightarrow{r_3} \emptyset \end{aligned} \tag{4.3}$$

For this filter, species A is the input and species C serves as the output. In addition to employing the co-expression method, the annihilation filter introduces a delay to the system via the reaction $D \xrightarrow{r_1} E$, resulting in further noise reduction. The Fano factor can even converge to zero with this filter. The reaction rate coefficients r_1 and r_2 should be as large as possible, while r_3 is the crucial parameter that determines the duration of the delay. With the initial concentrations of B and E set to be the same, the output species eventually converges to the expected value of the input.

The different noise suppression effects of these three filters are demonstrated in the example below:

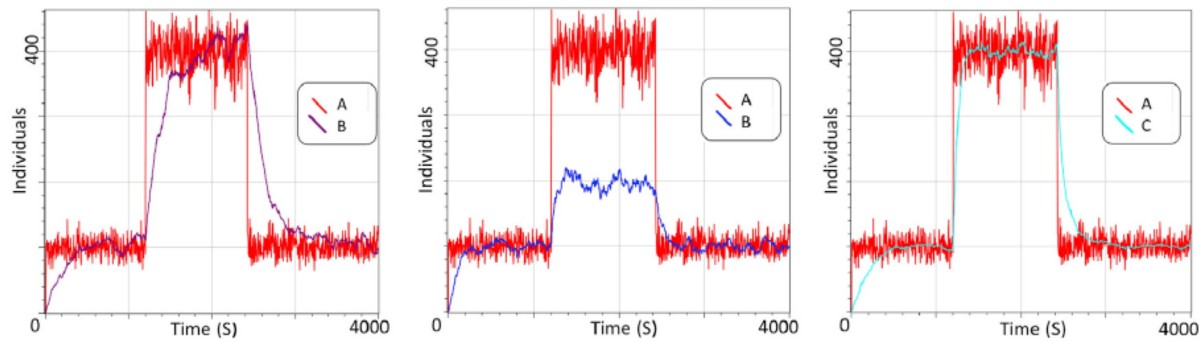


Figure 4.1: Comparison of Three Noise Filters

The noisy input A is generated by $\emptyset \xrightarrow{k_p} A+A \quad A \xrightarrow{k_d} \emptyset$. From left to right: Linear Filter ($k_1 = 0.0064, k_2 = 0.0064$), Annihilation Module ($r_1 = 0.005, r_2 = 0.00005$) and Annihilation Filter ($r_1 = 100, r_2 = 1000, r_3 = 0.000055$).

Based on the information provided, it is evident that each of the three noise filters possesses distinct advantages and disadvantages, rendering them suitable for systems with different conditions and different noise suppression requirements.

4.3.2 Biological PID Controller

In control systems, the PID (Proportional-Integral-Derivative) controller is a widely used tool for system regulation and noise suppression. It offers robust performance and functional simplicity [55]. The PID controller combines three main components, namely the proportional, integral, and derivative parts, to achieve control over the system dynamics. The structure of a traditional PID system is depicted in Figure 4.2.

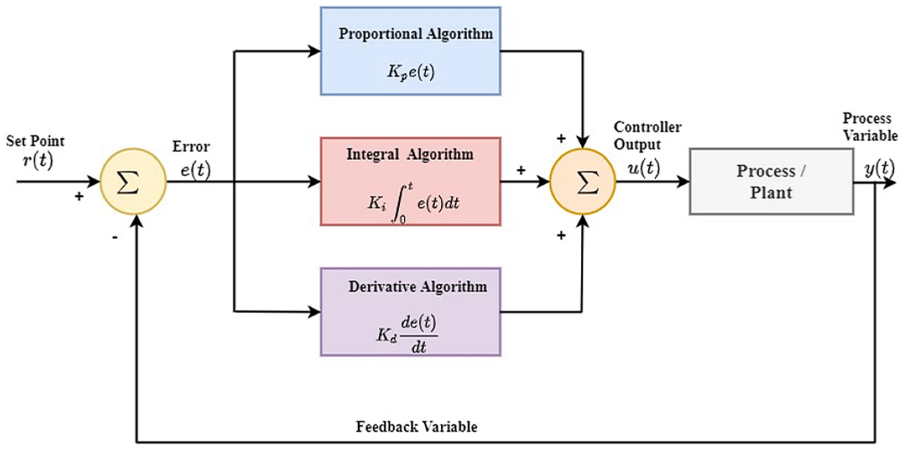


Figure 4.2: Block Diagram of Process Control using PID [6]
The error signal $e(t)$ is obtained by comparing the desired set-point $r(t)$ with the actual system output $y(t)$. The proportional term $K_p \cdot e(t)$ produces an output that is directly proportional to the current error. The integral term $K_i \cdot \int e(t)dt$ integrates the error signal over time to account for accumulated errors. Finally, the derivative term $K_d \cdot \frac{de(t)}{dt}$ calculates the rate of change of the error and adds a corrective component based on the error's temporal behaviour.

In 2021, Modi et al. introduced a biochemical PID controller that incorporates specific adaptations, aiming to explore the influence of each component on both internal and external noise [40]. The design and noise suppression effects of each PID component are investigated in detail below.

4.3.2.1 Proportional Part

Instead of using the error signal $e(t)$, the biochemical proportional algorithm operates on the target signal $y(t)$. This is achieved by introducing a sensor species Z. By configuring the birth rate of Z as $k_z Y$, the sensor species responds proportionally to the concentration of the target species Y. To prevent unlimited growth, Z decays with a rate coefficient γ_z. Additionally, Z inhibits the production of Y, closing the loop for regulation (see *Figure 4.3 (a)*).

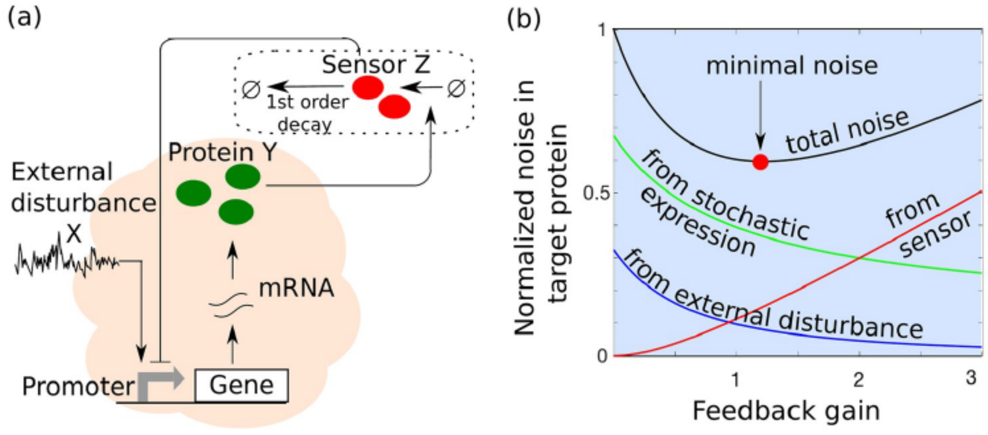


Figure 4.3: Structure and Noise of Proportional Part [40]

The CRN of the part in the dotted rectangular box is as follows:

$$\emptyset \xrightarrow{k_z Y} Z$$
$$Z \xrightarrow{\gamma_z} \emptyset$$

The normalized noise in Y concerning feedback gain is presented in *Figure 4.3 (b)*. The graph illustrates that increasing the feedback gain decreases intrinsic noise (stochastic expression noise) without eliminating it entirely, and reduces external noise to zero. Nevertheless, the introduction of the sensor species Z brings about another type of noise, as it has its own noise characteristics. Specifically, the sensor noise increases with the proportional gain. An optimal proportional gain exists (represented by the red point) where the cumulative noise is minimized.

It's noteworthy that higher feedback gains lead to reduced output static sensitivity in the proportional controller.

4.3.2.2 Integral Part

The biochemical integral controller utilizes the error signal $e(t)$ to regulate the system. In this case, integration is achieved by setting the creation rate and annihilation rate of the sensor species Z. The creation rate remains consistent with that in the proportional part, while the decay rate of Z is $k_z\overline{\langle Y \rangle}$, where $\overline{\langle Y \rangle}$ represents the steady-state value of the target species

Y. Therefore, the concentration change of Z at each moment reflects the multiplied error between the desired and actual values of Y, and it integrals over time. Additionally, Z inhibits Y as in the previous section, allowing the controller output to be passed to Y and closing the loop (see *Figure 4.4 (a)*).

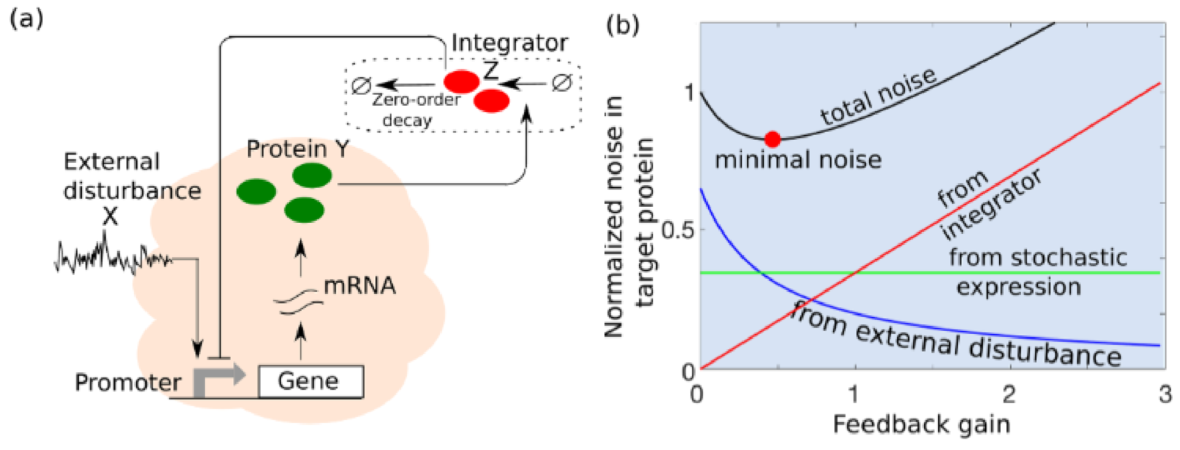


Figure 4.4: Structure and Noise of Integral Part [40]

The CRN of the integral algorithm in the dotted rectangular box can be written as:

$$\emptyset \overset{k_z Y}{\longrightarrow} Z$$

$$Z \overset{k_z \overline{\langle Y \rangle} / Z}{\longrightarrow} \emptyset$$

Compared to the proportional controller, this design of the integral feedback loop has no impact on the intrinsic noise (as shown in *Figure 4.4 (b)*). This is because the integrator treats noises differently. By performing an integration operation on the input, high-frequency fluctuations can be effectively eliminated, while low-frequency fluctuations remain unaffected. In other words, the integrator can be approximated as a low-pass filter. Since internal noise is typically considered a low-frequency disturbance, it remains unaffected by the integrator's operation [17].

The external noise decreases as the feedback gain increases, and the integrator species introduces additional sensor noise, similar to what is observed in the proportional controller. In this case, the magnitude of the noise from the integrator is directly proportional to the feedback gain. The total noise exhibited by the integral controller follows a U-shaped pattern concerning the increment of the integral gain, with the optimal gain value located near the origin.

Despite its lack of input-output sensitivity, the integral controller effectively eliminates steady-state errors and is therefore a valuable component of the PID controller.

4.3.2.3 Derivative Part

The derivative part also uses the target signal $y(t)$. The differentiation is achieved through the principle of buffering, where species Y transmits its current value to the sensor species Z while annihilating itself to obtain a new value. Simultaneously, Z stimulates the creation of Y, thereby passing the previous value forward. In this manner, the multiplied difference between the previous and current values is the changing rate of Y, enabling the differentiation process (refer to *Figure 4.5 (a)*). To prevent unlimited growth, Z decays with a rate coefficient γ_z.

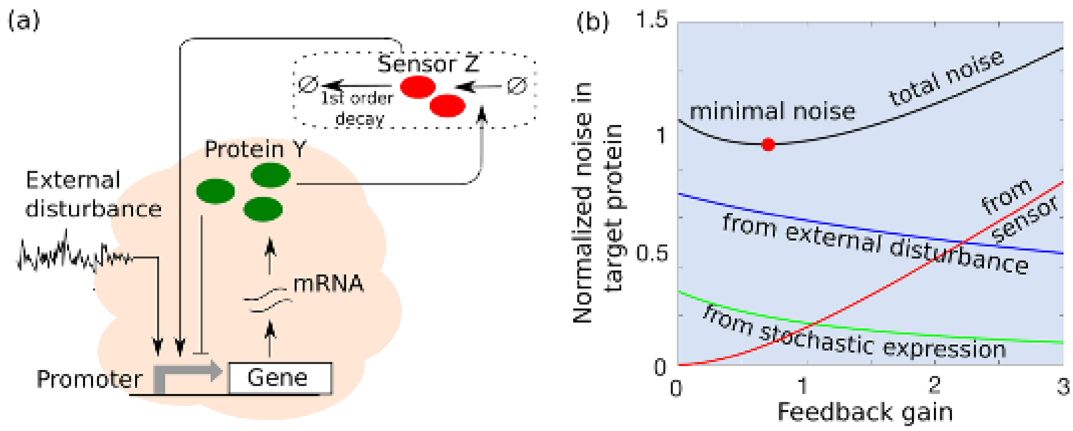


Figure 4.5: Structure and Noise of Derivative Part [40]

The dynamics of the algorithm in the dotted rectangular box can be described as follows:

$$\emptyset \xrightarrow{k_z Y} Z$$

$$Z \xrightarrow{\gamma_z} \emptyset$$

In *Figure 4.5 (b)*, it is evident that the derivative controller effectively reduces both intrinsic and external noise. As the derivative gain increases, the noise suppression effect becomes more pronounced. However, the introduction of the sensor species leads to additional noise, resulting in a larger overall noise when the gain is set to high values. Similar to the proportional and integral controllers, the derivative controller exhibits an optimal gain value that minimizes total noise.

The derivative controller proves to be highly effective in noise reduction while preserving the system's static sensitivity. This characteristic makes it a valuable tool for precise control tasks.

4.3.3 Summary

The noise filters presented in this section offer modular solutions for noise suppression, allowing easy integration into the target system. For our specific BioSD system, which is not a uni-molecular system, we can exclude the linear filter initially. Furthermore, our priority lies in ensuring that the filtered output signal remains accurate, and a little delay does not pose a significant concern. As such, the annihilation filter is a suitable choice for our purpose.

In the context of the biological PID controller, we can exclude the integral part due to its inability to reduce stochastic noise and its requirement for additional implementation to obtain the steady-state value of Y. Regarding the proportional and derivative parts, duplicating species X in BioSD as protein Y might help reduce intrinsic noise. This deviation prevents the concentration of Y from accurately tracking that of species X, rendering the attachment of these "P" and "D" controllers to BioSD unsuitable for noise reduction. Nevertheless, these findings provide valuable insights into the noise characteristics of a differentiator and contribute to our understanding of BioSD noise behavior.

In conclusion, we ultimately select the annihilation filter as our preferred method for noise suppression.

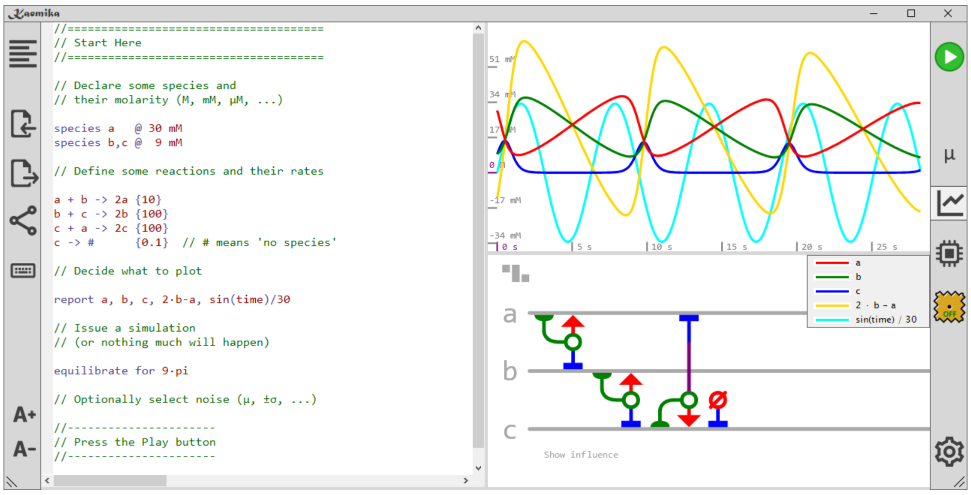
4.4 SOFTWARE TOOLS

For accurate exploration and analysis of noise characteristics, the selection of an appropriate software simulation tool that can effectively simulate and quantify noise is crucial. Ideally, these tools should be equipped with noise evaluation methods. Currently, there are several up-to-date applications available to fulfil these requirements. Among them, two widely used tools, namely **Kaemika** [9] and **ERODE** [10], have gained recognition for their capabilities in simulating biochemical systems and assessing noise within the models. In this section, we introduce and discuss these software tools, highlighting their features and functionalities for studying noise properties.

4.4.1 Kaemika

Kaemika is a relatively new software tool developed by Luca Cardelli in 2020 [9]. This integrated application serves as a versatile platform for simulating the performance of both deterministic and stochastic biochemical models. It incorporates essential modelling techniques such as Mass Action Kinetics and Linear Noise Approximation (LNA). Kaemika's capabilities extend beyond simulation, as it also supports implementation on microfluidic devices through the utilization of liquid-handling protocols.

The user interface of the Kaemika application consists of three main components, as depicted in *Figure 4.6*. The left part is the script editor, which adopts the notation of Chemical Reaction Networks and additionally incorporates basic programming syntax, such as parameters, conditional control, and loop statements, simplifying the modelling process. The top right corner features the plot component, enabling the visualization of expected concentration changes over time for the selected duration. Alongside the mean value, Kaemika offers various output options, including standard deviation and the Fano factor. Each output is represented by a distinct colour on the plotted curves. The system representation section is located in the lower right corner, providing different ways to visualize the system. Notably, a novel and clear representation resembling "Petri Nets" known as the reaction score is available. This representation visually captures the relationships between species, offering enhanced insights.



```
//=====================================
// Start Here
//=====================================

// Declare some species and
// their molarity (M, mM, μM, ...)

species a   @ 30 mM
species b,c @  9 mM

// Define some reactions and their rates

a + b -> 2a {10}
b + c -> 2b {100}
c + a -> 2c {100}
c -> #      {0.1}  // # means 'no species'

// Decide what to plot

report a, b, c, 2·b-a, sin(time)/30

// Issue a simulation
// (or nothing much will happen)

equilibrate for 9·pi

// Optionally select noise (μ, ±σ, ...)

//---------------------
// Press the Play button
//---------------------
```

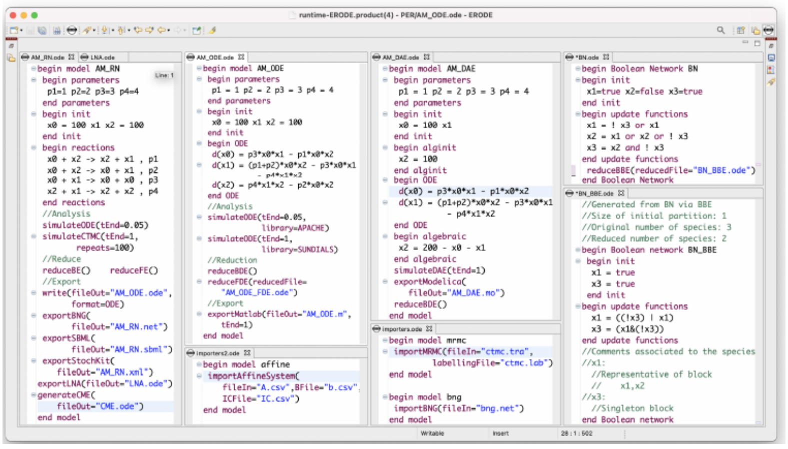
Figure 4.6: Example of Kaemika Interface

Overall, Kaemika's comprehensive features and user-friendly interface make it a powerful software tool for simulating and analyzing biochemical systems. It has gained popularity as a valuable tool and has already been utilized in research studies such as [12] and [1], where it was employed to calculate species distributions and track variance evolution.

4.4.2 ERODE

ERODE is specifically designed for the analysis and reduction of Ordinary Differential Equations (ODEs), Chemical Reaction Networks (CRNs), and other quantitative models [10]. Its user interface and functionalities resemble those of other coding tools, providing a familiar environment for users.

The script editor in ERODE follows a "code-based" approach, where users create a file and type commands in a notebook-like tab for execution. The integrated development environment of the code editor includes features such as syntax highlighting, in-line error annotation, fix suggestions, and auto-completion. To model CRNs, users can define reactions and rates, and input simulation commands. An example of the ERODE script editor interface is shown in *Figure 4.7*.



```
begin model AM_RN
 begin parameters
  p1=1 p2=2 p3=3 p4=4
 end parameters
 begin init
  x0 = 100 x1 x2 = 100
 end init
 begin reactions
  x0 + x2 -> x2 + x1 , p1
  x0 + x2 -> x0 + x1 , p2
  x0 + x1 -> x0 + x0 , p3
  x2 + x1 -> x2 + x2 , p4
 end reactions
 //Analysis
 simulateODE(tEnd=0.05)
 simulateCTMC(tEnd=1,
        repeats=100)
 //Reduce
 reduceBE()    reduceFE()
 //Export
 write(fileOut="AM_ODE.ode",
       format=ODE)
 exportBNG(
       fileOut="AM_RN.net")
 exportSBML(
       fileOut="AM_RN.sbml")
 exportStochKit(
    fileOut="AM_RN.xml")
exportLNA(fileOut="LNA.ode")
generateCME(
    fileOut="CME.ode")
end model
```

```
begin model AM_ODE
 begin parameters
  p1 = 1 p2 = 2 p3 = 3 p4 = 4
 end parameters
 begin init
  x0 = 100 x1 x2 = 100
 end init
 begin ODE
  d(x0) = p3*x0*x1 - p1*x0*x2
  d(x1) = (p1+p2)*x0*x2 - p3*x0*x1
             - p4*x1*x2
  d(x2) = p4*x1*x2 - p2*x0*x2
 end ODE
 //Analysis
 simulateODE(tEnd=0.05,
            library=APACHE)
 simulateODE(tEnd=1,
            library=SUNDIALS)
 //Reduction
 reduceBDE()
 reduceFDE(reducedFile=
    "AM_ODE_FDE.ode")
 //Export
 exportMatlab(fileOut="AM_ODE.m",
    tEnd=1)
end model
```

```
begin model affine
 importAffineSystem(
    fileIn="A.csv",BFile="b.csv",
    ICFile="IC.csv")
end model
```

```
begin model AM_DAE
 begin parameters
  p1 = 1 p2 = 2 p3 = 3 p4 = 4
 end parameters
 begin init
  x0 = 100 x1
 end init
 begin alginit
  x2 = 100
 end alginit
 begin ODE
  d(x0) = p3*x0*x1 - p1*x0*x2
  d(x1) = (p1+p2)*x0*x2 - p3*x0*x1
            - p4*x1*x2
 end ODE
 begin algebraic
  x2 = 200 - x0 - x1
 end algebraic
 simulateDAE(tEnd=1)
 exportModelica(
    fileOut="AM_DAE.mo")
 reduceBDE()
end model
```

```
begin model mrmc
 importMRMC(fileIn="ctmc.tra",
        labellingFile="ctmc.lab")
end model

begin model bng
 importBNG(fileIn="bng.net")
end model
```

```
begin Boolean Network BN
begin init
 x1=true x2=false x3=true
end init
begin update functions
 x1 = ! x3 or x1
 x2 = x1 or x2 or ! x3
 x3 = x2 and ! x3
end update functions
 reduceBBE(reducedFile="BN_BBE.ode")
end Boolean Network
```

```
//Generated from BN via BBE
//Size of initial partition: 1
//Original number of species: 3
//Reduced number of species: 2
begin Boolean network BN_BBE
 begin init
  x1 = true
  x3 = true
 end init
begin update functions
  x1 = ((!x3) | x1)
  x3 = (x1&(!x3))
end update functions
//Comments associated to the species
//x1:
  //Representative of block
  //    x1,x2
//x3:
  //Singleton block
end Boolean network
```

Figure 4.7: Example Script Editor of ERODE [10]

The plotting functionality in ERODE enables not only deterministic species evolution analysis but also stochastic analysis using Gillespie's Direct Method or Linear Noise Approximation (LNA). Mean and variance plots can be generated, with different colours representing various species. An example of variance is shown in *Figure 4.8*.

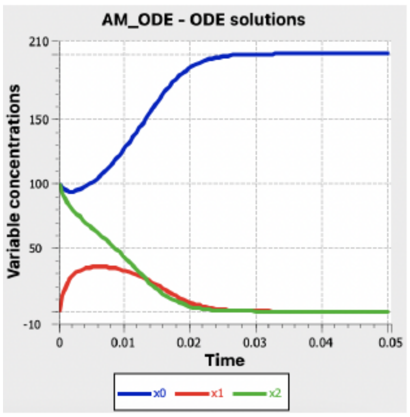


Figure 4.8: Example Plot of Variance in ERODE [10]

ERODE's exchange interface offers considerable flexibility in modelling. Users can import Continuous-Time Markov Chains (CTMCs) into the editor and export CTMCs to platforms like Matlab and PRISM. Additionally, ERODE provides model reduction capabilities, allowing users to simplify models without compromising their primary dynamic behaviour.

In conclusion, ERODE serves as a powerful software tool, offering a coding-based environment for the analysis, reduction, and exchange of CTMCs. It has found extensive usage in various studies. For instance, Argyris et al. utilized ERODE to reduce Boolean Networks in [4], and G. Pogudin and X. Zhang employed ERODE to compare the correctness of Linear Reduction in [45].

4.4.3 Comparison

Both Kaemika and ERODE offer simulation capabilities for both deterministic and stochastic models. However, there are some differences in their noise evaluation, user interface and functionalities.

Kaemika provides the Fano factor directly as an output, while ERODE calculates the Fano factor using variance and mean derived from LNA, which is available in both software tools. In this study, **Fano factor** is chosen for evaluating noise due to its availability in both applications.

From the perspective of programming difficulty, Kaemika stands out as it requires users to input information about the CRNs, and the noise analysis can be performed with a simple click. On the other hand, although ERODE provides a more user-friendly coding environment, it involves coding all the necessary operations in addition to specifying the reactions, making it a more complex tool.

Furthermore, Kaemika features an additional species relationship graph, which enhances the visualization of system dynamics. ERODE's interface allows for connections with other applications like Matlab, making it advantageous for data analysis. Additionally, ERODE provides model simplification capabilities, which can be valuable for more complex systems. However, given the simplicity of the CRNs in this thesis, the data exchange and model reduction functions are deemed unnecessary.

Considering these factors, Kaemika is selected as the software tool for this thesis due to its ease of use, and direct availability of the Fano factor for noise evaluation.

4.5 Input Signals for Testing

To comprehensively assess the effect of BioSD on noise, it is essential to select a variety of input signals with diverse characteristics. While a previous study [2] utilized the AIF and a simple linear system as input signals, we substitute the linear system with a hyperbolic signal, keep the AIF, and introduce a sinusoidal signal as an additional input, aiming to enhance the variety and complexity of the test signals. These adjustments allow for a more robust evaluation of BioSD's impact on noise under varying signal conditions.

Another crucial consideration when selecting test signals is the accuracy of the derivative results computed by BioSD. If the computed derivatives do not accurately match the expected values, it indicates that BioSD may struggle to handle that particular signal accurately, rendering it unsuitable for noise analysis.

In this section, we first introduce each input signal and then test their derivative accuracy calculated by BioSD against the selected reference derivative. This process not only lays a robust foundation for the noise analysis but also provides a comprehensive assessment of BioSD's performance across various scenarios.

4.5.1 Hyperbolic Tangent

The hyperbolic tangent is defined as

$$tanh\, z \equiv \frac{sinh\, z}{cosh\, z} = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

The mathematical graph of the hyperbolic tangent is depicted in *Figure 4.9*. The curve rises for approximately 6 seconds with a varying slope, while remaining nearly constant elsewhere. This simple input exhibits different types of behaviour, offering a slight challenge in verifying the instantaneous behaviour and response speed of BioSD.

To implement the hyperbolic tangent in Kaemika, certain modifications are necessary. Due to the limitation that BioSD cannot handle negative input signals, the hyperbolic tangent is shifted up by one unit. Additionally, Kaemika does not support drawing signals before 0 seconds, so the curve is shifted to the right by five units to include the rising part. These adjustments can be achieved using a simple function in Kaemika, as shown in *S1 of Supplementary Information*. The resulting graph in Kaemika is depicted in *Figure 4.10*.

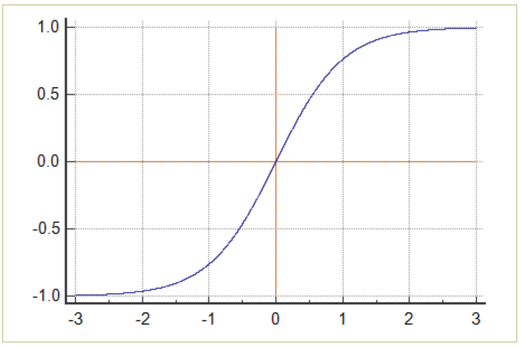


Figure 4.9: Hyperbolic Tangent

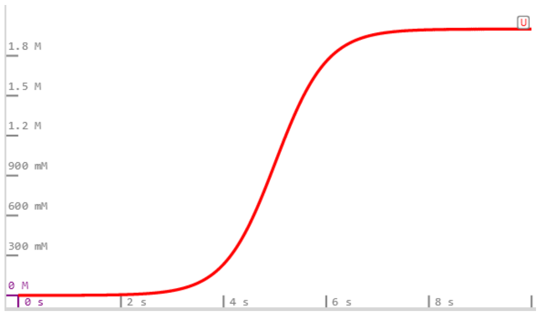


Figure 4.10: Hyperbolic Tangent in Kaemika

4.5.2 Antithetic Integral Feedback (AIF)

AIF represents a type of integral control that takes noise into account for system regularization, enabling robust set-point tracking and perfect adaptation even in the presence of random noise [7]. The design of AIF consists of two main components: the control part (left part) and the reaction network (right part), as illustrated in *Figure 4.11*:

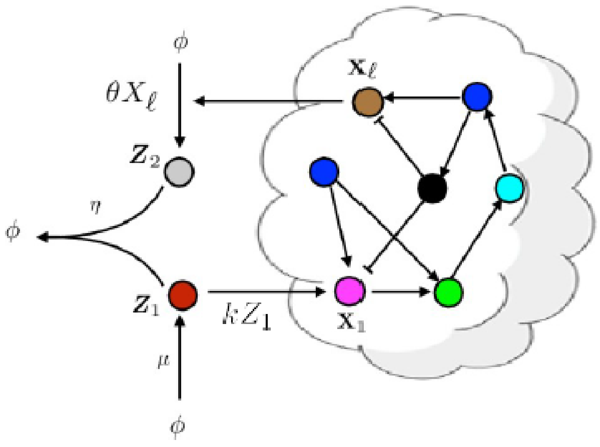


Figure 4.11: Structure of Antithetic Integral Feedback [7]

We select a simple reaction network and the CRN of the system becomes:

$$\emptyset \xrightarrow{\mu} Z_1, \qquad \emptyset \xrightarrow{kZ_1} X_1$$
$$\emptyset \xrightarrow{\theta X_2} Z_2, \qquad Z_1 + Z_2 \xrightarrow{\eta} \emptyset$$
$$X_1 \xrightarrow{v_1} X_1 + X_2$$
$$X_1 \xrightarrow{v_2} \emptyset \quad X_2 \xrightarrow{v_3} \emptyset$$

In this CRN, X_1 is the actuated species, and X_2 is the controller species as the output. The concentration of the output affects the generation rate of Z_2, essentially storing the output value in Z_2. The annihilation of Z_1 and Z_2 acts as a subtraction operation. The remaining concentration of Z_1 becomes the new input signal sent back to the reaction network.

These reaction rate coefficients determine whether the system is stable or not. The system is considered stable if the output eventually reaches its desired value, μ/θ. In some cases, this process may be oscillatory, like the system used in [2] (see *Figure 4.12* below).

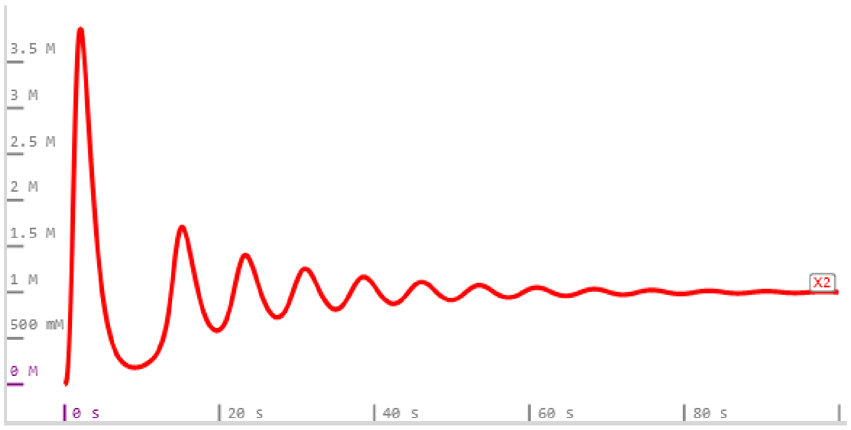


Figure 4.12: Antithetic Integral Feedback in Kaemika (t = 100 s)
$\mu = k = \theta = 2, \eta = 12, v_1 = 4, v_2 = v_3 = 1$

The dynamics of the AIF system can be challenging to differentiate as it starts with a sharp increase and then continues to oscillate with decreasing amplitudes. This behaviour can be used to test the performance of BioSD, with X_2 serving as the input signal.

To avoid conflicts and ensure consistency throughout our implementation, we make some modifications to the code of AIF since some of the notations used in AIF are the same as those in BioSD. The modified code can be found in *S1 of Supplementary Information.*

4.5.3 Sinusoidal Wave

As mentioned in *subsection 3.1.2*, when utilizing BioSD, the focus lies on the proximity to their respective steady states. For the previous two test signals, they eventually flatten out, leading to derivatives that are almost zero at the end, indicating BioSD's closeness to steady states. However, signals like sinusoidal wave prevent BioSD from reaching a steady state, and such signals have not been extensively studied in research. By investigating the responses of BioSD to such signals, we can gain valuable insights into the limitations and robustness of the system. The mathematical graph of $sin(x)$ is depicted in *Figure 4.13*.

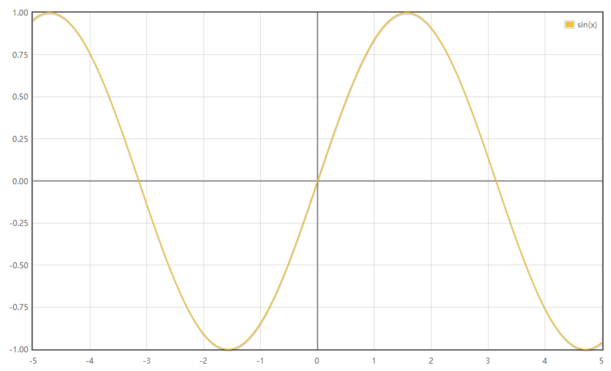


Figure 4.13: Sinusoidal Wave

Directly using $sin(x)$ as the input signal for BioSD is not feasible due to the negative portions of the curve. To address this, we shift the signal upwards and introduce a phase shift to ensure that the signal begins at 0. Without this phase shift, the curve would experience a sudden jump at the start, leading to an extremely high derivative. This abrupt transition would result in a significant but unnecessary alteration in the species concentration within BioSD (please refer to the figures above, which illustrate the curves in Kaemika, for a clearer demonstration). To implement this in Kaemika, we employ a function as outlined in *S1 of Supplementary Information*.

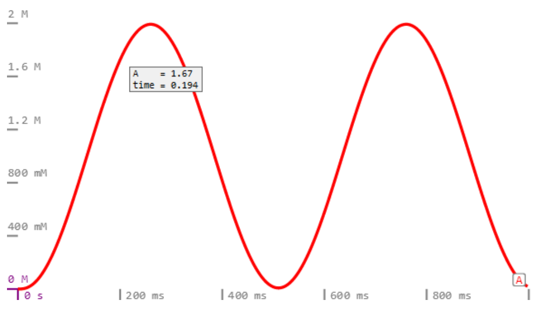


Figure 4.14: phase = $\frac{3}{2}\pi$

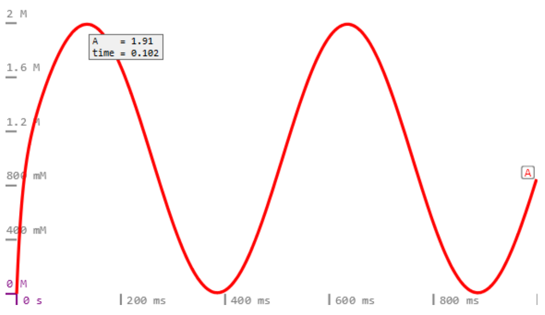


Figure 4.15: phase = 0

4.5.4 Accuracy Tests of Input Signals

In this subsection, we first select a reference derivative for comparison, and then evaluate the accuracy of BioSD on the three input signals.

4.5.4.1 Reference Derivative

To assess the calculation performance of BioSD, we need a reference derivative for comparison. Since direct differential functions are not available in Kaemika, we have to use an alternative method involving reactions.

In selecting a reference derivative, our primary concerns are accuracy and computational speed. Additionally, we prefer a simpler composition to avoid excessive computational load in Kaemika, which could result in longer computation times for obtaining the graph.

With these requirements in mind, we choose the *"Derivative1"* method from the example code provided by Kaemika as our reference derivative, whose code in Kaemika can be found in *S1 of Supplementary Information*. This method records the input value from the previous time step and calculates the difference between the current value and the recorded value to obtain the derivative. It has a single-rail input, denoted as A, and a dual-rail output, represented by $B^+ - B^-$. Since BioSD can only handle non-negative inputs, the single-rail input is enough, and we refer to this method as the Single-Rail Differentiator (SRD) in the subsequent sections.

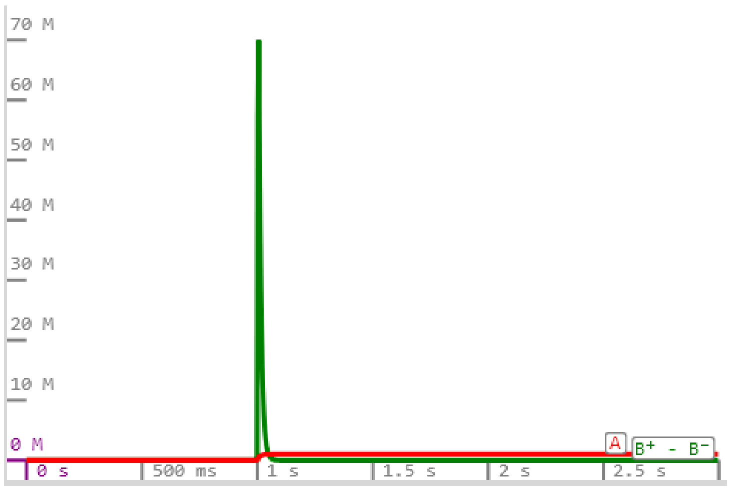


Figure 4.16: Output of SRD on Step Signal in Kaemika

We test the performance of this reference derivative using a challenging step signal A, which rises from 0 to 1 at $t = 1$ second. The derivative result is shown in *Figure 4.16*. It can be observed that SRD experiences an immediate surge at $t = 1$ second and quickly returns to 0 in a very short span of time, indicating that the reference derivative is not only accurate but also fast.

4.5.4.2 Accuracy Results

In the subsequent figures of BioSD outputs, we use the following notations: X_1 represents the output of BioSD-I, X_2 represents the output of BioSD-II, and X_3 represents the output of BioSD-III.

To facilitate a more straightforward comparison between the outputs and the reference derivatives, we employ specific methods. Firstly, for each test signal, all three differentiators of BioSD share one set of parameters (except for parameter b). This ensures that they all have the same amplification and bias, making them close to each other in the graphs. Secondly, we multiply the reference derivative by the same amplification and then add the same bias as the BioSD outputs. This approach ensures that the reference derivative and the BioSD outputs are scaled to the same magnitude. Thirdly, the outputs are initialized as the bias to avoid the effects of the "warm-up" phase. By adopting these techniques, we can conveniently assess the performance of BioSD by analyzing the deviations of its outputs from the adjusted reference derivative.

Hyperbolic Tangent:
The reference derivative of the hyperbolic tangent in Kaemika is represented by the green curve in *Figure 4.17*:

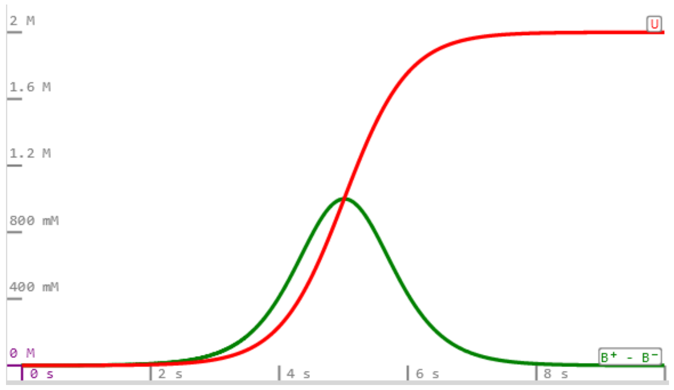


Figure 4.17: Reference Derivative of Hyperbolic Tangent

After conducting tests and adjusting the parameters with variations of 1 for k_{in} and b, 0.1 for k_1, k_2, k_3 and δ, and 10 for η, the results shown in *Figure 4.18* are obtained.

From the figure, we can observe that the outputs of BioSD-I and BioSD-II exhibit small fluctuations at the beginning, and the derivatives of all three BioSDs slightly lag behind the reference derivative during the rising period of the hyperbolic tangent. Overall, the errors in using BioSD for derivative calculation are quite small, and we can consider the computed results to be fairly accurate.

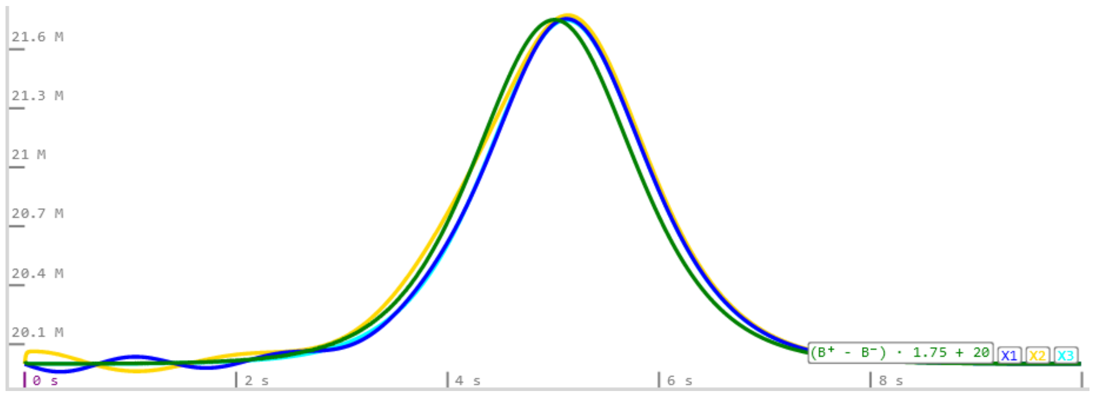


Figure 4.18: Modified Reference Derivative and BioSD Outputs for Hyperbolic Tangent
$k_{in} = 35$, $k_1 = 1$, $k_2 = 1$, $k_3 = 20$, $\eta = 230$, $\delta = 0.5$. For BioSD-I and BioSD-III, $b = 10$ (corresponding to $\epsilon = 0.0125 \ll 1$), while for BioSD-II, b is chosen as 16 (corresponding to $\epsilon = 0.032 \ll 1$). These settings result in an amplification of $\frac{k_{in}}{k_1 k_3} = 1.75$ and a bias of $\frac{k_3}{k_2} = 20$.

AIF:
The reference derivative of the AIF is represented by the green curve $B^+ - B^-$ in Kaemika (see *Figure 4.19 below*):

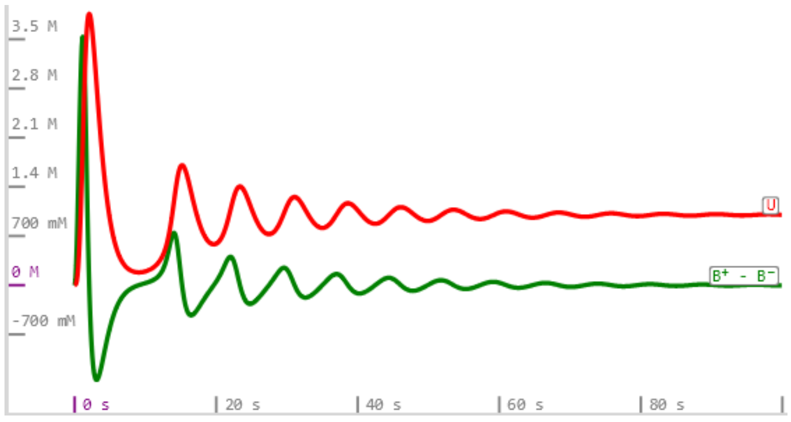


Figure 4.19: Reference Derivative of Antithetic Integral Feedback (t = 100 s)
$v_1 = v_2 = v_4 = 2, v_3 = 4, v_5 = 12, v_6 = v_7 = 1$

After conducting tests and adjusting the parameters with variations of 1 for k_{in}, b and k_3, 0.1 for k_1, k_2, and δ, and 50 for η, we obtain the results presented in *Figure 4.20*.

When the time is set to 100 seconds, the calculated derivatives obtained by BioSD seem to perfectly match the modified reference derivative.

To further verify the accuracy of BioSD, we zoom in and observe the difference between the results obtained by BioSD and the reference derivative during the first 10 seconds, where the most intense fluctuations of AIF occur(see *Figure 4.21*). In this figure, we can observe that

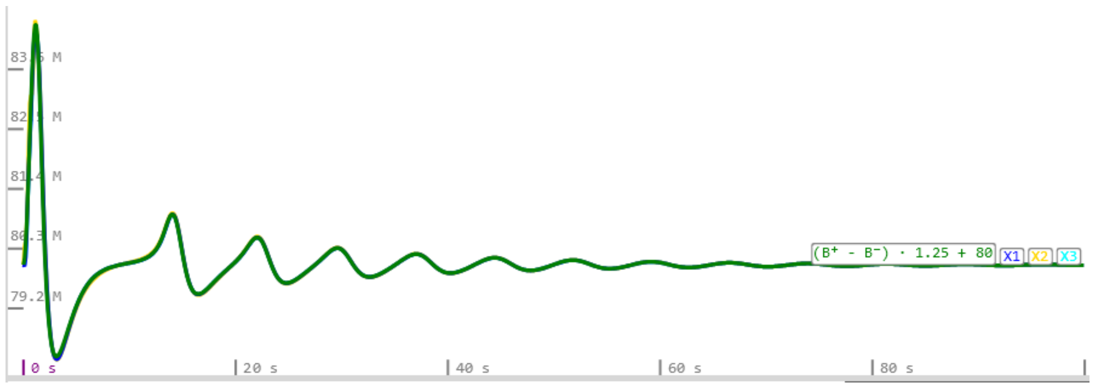


Figure 4.20: Modified Reference Derivative and BioSD Outputs for AIF (t = 100s)
k_{in} = 190, k_1 = 1.9, k_2 = 1, k_3 = 80, η = 1050, δ = 0.5. For BioSD-I and BioSD-III, b = 41 (corresponding to ϵ = 0.055 $\ll$ 1), while for BioSD-II, b is selected as 85 (corresponding to ϵ = 0.0777 $\ll$ 1). These settings result in an amplification of $\frac{k_{in}}{k_1 k_3}$ = 1.25 and a bias of $\frac{k_3}{k_2}$ = 80.

the results of the three BioSD exhibit slight deviations from the reference derivative. Among them, BioSD-III has the smallest deviation from the reference derivative, followed by BioSD-I, while BioSD-II has the largest deviation. However, these errors are significantly less than 0.1, leading us to consider BioSD to be relatively accurate.

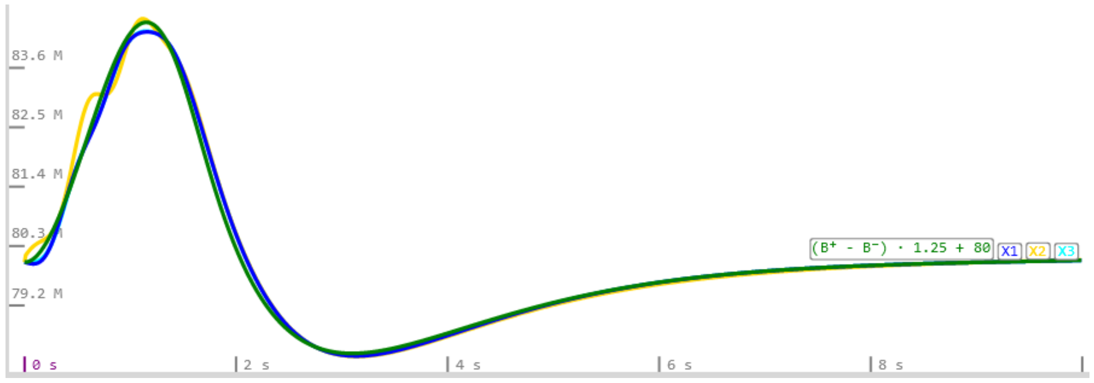


Figure 4.21: Modified Reference Derivative and BioSD Outputs for AIF (t = 10s)
All the other settings are the same as those in *Figure 4.20*.

Sinusoidal Wave:
The green curve $B^+ - B^-$ in *Figure 4.22* represents the reference derivative of the sinusoidal wave obtained in Kaemika.

After conducting tests and adjusting the parameters with variations of 1 for k_{in}, b and k_3, 0.1 for k_1, k_2, and δ, and 1000 for η, we obtain the results presented in *Figure 4.23*.

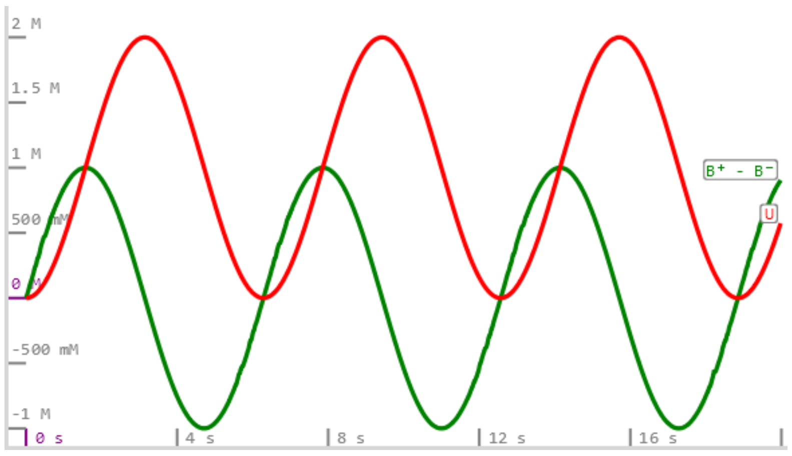


Figure 4.22: Reference Derivative of Sinusoidal Wave

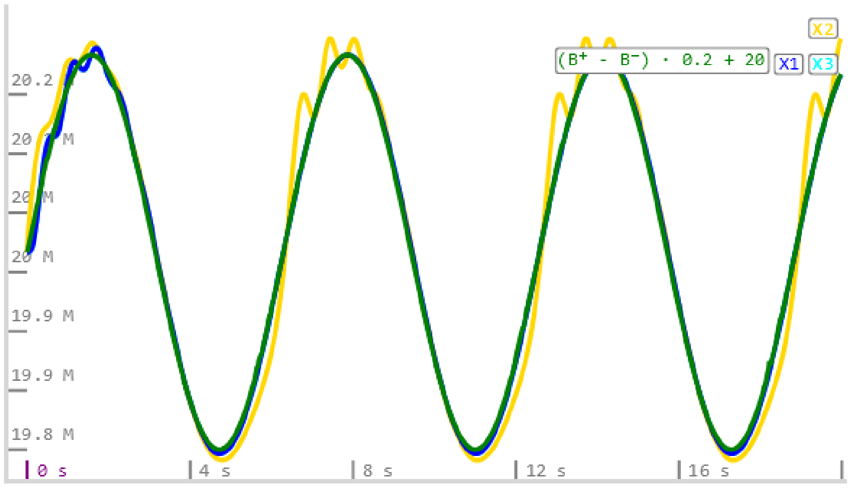


Figure 4.23: Modified Reference Derivative and BioSD Outputs for Sinusoidal Wave
$k_{in} = 20$, $k_1 = 5$, $k_2 = 1$, $k_3 = 20$, $\eta = 10000$, $\delta = 0.5$. For BioSD-I, $b = 11$ (corresponding to $\epsilon = 0.003 \ll 1$), for BioSD-II, b is selected as 15 (corresponding to $\epsilon = 0.0056 \ll 1$), and for BioSD-III, b is selected as 10 (corresponding to $\epsilon = 0.0025 \ll 1$) . These settings result in an amplification of $\frac{k_{in}}{k_1 k_3} = 0.2$ and a bias of $\frac{k_3}{k_2} = 20$.

In this figure, we can observe that the results of the three BioSD exhibit deviations from the reference derivative. Among them, BioSD-III has the smallest deviation from the reference derivative, followed by BioSD-I, while BioSD-II has the largest deviation. However, these errors are significantly less than 0.1, leading us to consider BioSD to be relatively accurate.

Interestingly, unlike the previous test signals, BioSD-I and BioSD-III now have different values for parameter b, despite their relatively similar values. This divergence highlights the sensitivity of the differentiators to parameter variations and underscores the importance of selecting appropriate parameter sets for each specific application.

4.5.4.3 Accuracy Summary

As mentioned earlier, to facilitate easier observation and comparison, we utilized common sets of parameters for BioSD in each experiment. However, due to the structural differences among the three differentiators, they may not share the same optimal parameters, particularly BioSD-II. Achieving improved results would require more meticulous parameter tuning. While further fine-tuning based on the previously discussed parameters with smaller variations could potentially lead to a perfect match with the reference derivative for BioSD-I and BioSD-III, achieving accurate calculation results for BioSD-II demands the exploration of different parameter sets. However, this tuning process is time and effort-intensive, and the *Condition 3.11* provides limited assistance, especially for the sinusoidal wave, which poses more intricacies compared to the previous two signals. Hence, we do not conduct more precise tuning at this stage.

Nevertheless, despite these considerations, the BioSD results obtained demonstrate a commendable level of accuracy, with the largest deviation being well below 0.1. As a result, all three differentiators are considered qualified for further noise analysis.

Chapter 5

Noise Analysis Result

In this chapter, we employ the tools and methodologies outlined in the previous chapter to conduct a comprehensive experimental analysis of the noise characteristics exhibited by BioSD. Each section of this chapter is dedicated to a specific input signal that has been selected for examination.

Our approach begins with the determination of the Fano factor associated with the chosen input signal. Subsequently, we utilize BioSD to process the input signal, generating Fano factors for each of the BioSD outputs.

Furthermore, we delve into an exploration of the relationship between parameter b and the noise levels by introducing modifications to the value of b. In instances where noise levels are deemed excessively high, we implement the annihilation filter to assess its potential to reduce the Fano factor to levels lower than that of the input signal or even approaching the Poisson level.

By undertaking these investigations, we aim to gain insights into the noise properties of BioSD and understand the impact of parameter b on its overall noise behaviour. Additionally, the integration of a noise filter explores the possibility of enhancing reliability and expanding applications of BioSD.

Throughout this chapter, graphical representations will utilize the following conventions: X_1 signifies the output of BioSD-I, X_2 corresponds to the output of BioSD-II, and X_3 pertains to the output of BioSD-III. Additionally, the designations F_1, F_2, and F_3 denote the filtered outputs of BioSD-I, BioSD-II, and BioSD-III respectively. Legends featuring μ represent the mean value, while those featuring σ^2/μ denote the Fano factor.

5.1 Hyperbolic Tangent

We first analyze the noise level of the hyperbolic tangent. The Fano factor of the hyperbolic tangent is consistently equal to 1 (denoted as $U\sigma^2/\mu$ in *Figure 5.1*), indicating a Poisson level of noise.

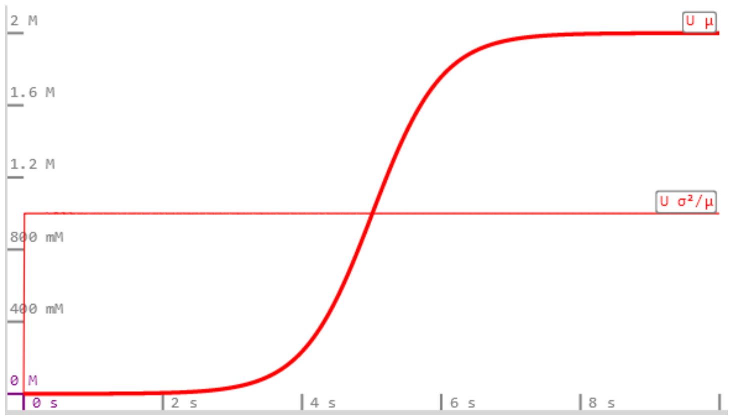


Figure 5.1: Noise Level of Hyperbolic Tangent in Kaemika

Then the hyperbolic tangent works as the input to BioSD, and we get the Fano factor of BioSD outputs, which is illustrated in *Figure 5.2*. The noise levels of all three BioSD are significantly higher than the noise of the input signal and the Poisson level. BioSD-I and BioSD-III exhibit similar noise levels, while BioSD-II shows lower noise compared to the other two.

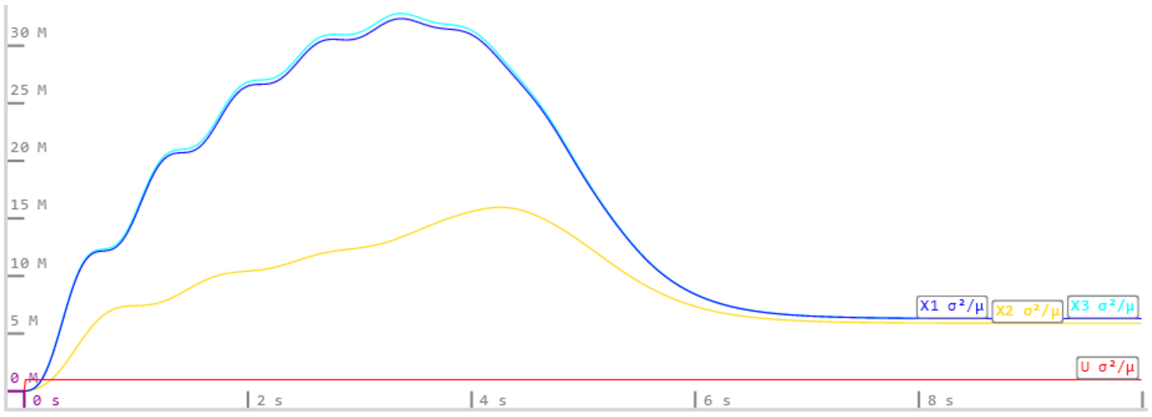


Figure 5.2: Noise Levels of Hyperbolic Tangent and BioSD Outputs in Kaemika
The parameter set used is the same as depicted in *Figure 4.18*, where $k_{in} = 35$, $k_1 = 1$, $k_2 = 1$, $k_3 = 20$, $\eta = 230$, $\delta = 0.5$. For BioSD-I and BioSD-III, $b = 10$, while for BioSD-II, b is selected as 16.

Parameter Modification

In addition to the inherent structural differences among the BioSD models, the disparity in noise levels, particularly the lower noise level observed in BioSD-II, might also be attributed to distinct parameter configurations, specifically parameter b.

To investigate this relationship, we conduct an experiment involving the modification of parameter b for BioSD-III. By increasing the value of b from 10 to 16, aligning it with the value used for BioSD-II, we observe a substantial reduction in the noise level of BioSD-III compared to BioSD-I, as evident from *Figure 5.3*. However, this decline in noise level is coupled with a compromise in accuracy, as the output of BioSD-III oscillates for about 5 seconds.

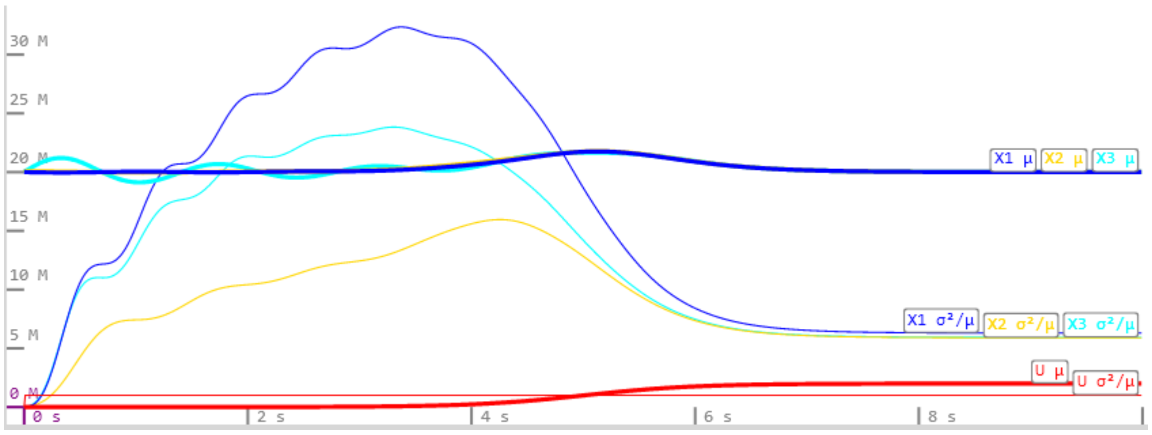


Figure 5.3: Noise Level of Hyperbolic Tangent and BioSD Outputs in Kaemika (Modified *b*)
The parameter set illustrated in *Figure 5.2* is also used here, except for $b = 16$ for BioSD-III.

Noise Suppression

To mitigate the noise depicted in *Figure 5.2*, we implement the annihilation filter as discussed in *Section 4.3.1*. It's essential to recognize that this filter achieves noise suppression by introducing a system delay. As a consequence, a trade-off emerges: while employing a sufficiently long delay can yield a Fano factor akin to or even lower than the Poisson level, the output of the filter will significantly deviate from that of BioSD due to the extended delay. Hence, taking into account both the impact of delay and noise level, one desirable filter is shown in *Figure 5.4*.

From the graph, it is evident that the corresponding Fano factor for each BioSD significantly decreases, resulting in substantial improvement. However, these Fano factor reductions come at the cost of slight delays and decreases in peak values (see *Figure 5.5* for clearer demonstration). Notably, the initial warm-up phases of the noise filter outputs cannot be eliminated through species initialization, as is done for BioSD.

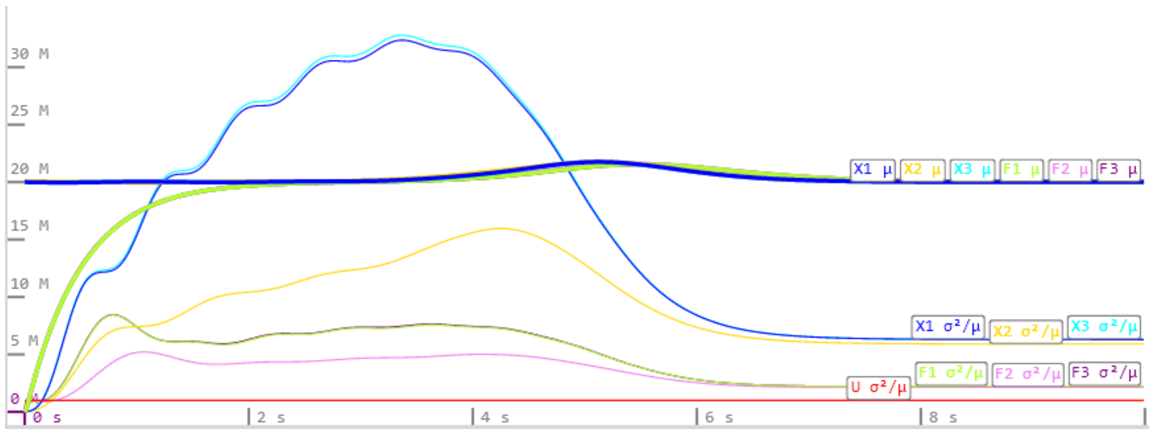
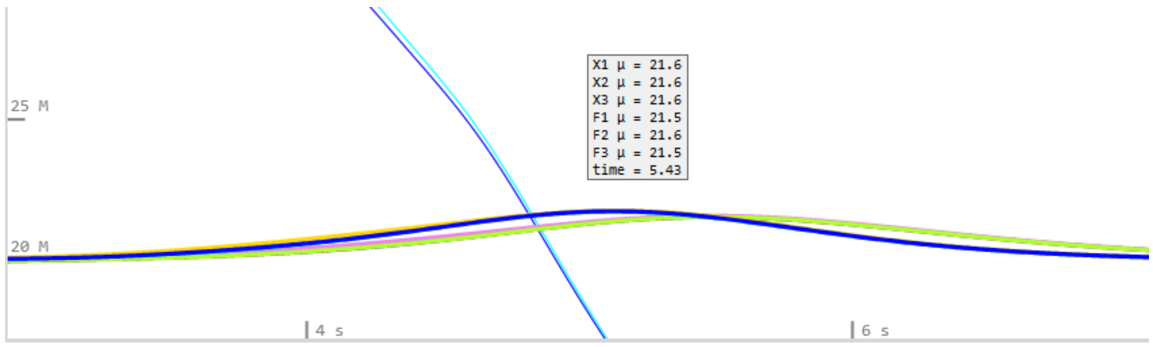


Figure 5.4: Hyperbolic Tangent and BioSD with Noise Filter in Kaemika
The parameter set used for BioSD is the same as depicted in *Figure 5.2.* The parameters of the Noise Filter are set as follows: $r_1 = 100$, $r_2 = 1000$, $r_3 = 0.1$.



```
X1 μ = 21.6
X2 μ = 21.6
X3 μ = 21.6
F1 μ = 21.5
F2 μ = 21.6
F3 μ = 21.5
time = 5.43
```

Figure 5.5: Zoomed-In Detail: BioSD Outputs with Noise Filter in Kaemika

5.2 Antithetic Integral Feedback (AIF)

The noise level of AIF is analyzed before further experiments, which exhibits pronounced oscillations and does not converge to the Poisson level(see *Figure 5.6*). After the initial few seconds, the Fano factor of U begins to oscillate with an approximate period of 8 seconds. Within each cycle, two oscillations occur, with one exhibiting a larger amplitude and the other a smaller amplitude. The amplitudes of these oscillations gradually diminish and eventually stabilize around 16.

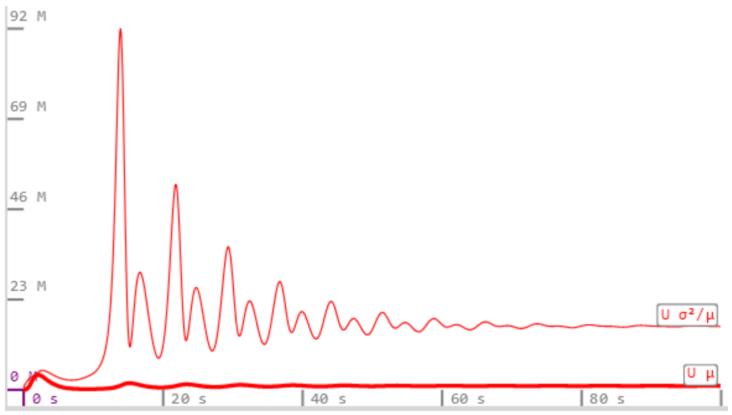


Figure 5.6: Noise Level of Antithetic Integral Feedback (t = 100 s)
$v_1 = v_2 = v_4 = 2$, $v_3 = 4$, $v_5 = 12$, $v_6 = v_7 = 1$

We subsequently delve into the analysis of the noise levels generated by BioSD. The noise levels of the three outputs exceed the Fano factor of the input signal for most of the time, with BioSD-I and BioSD-III exhibiting similar Fano factors and BioSD-II displaying a smaller one.

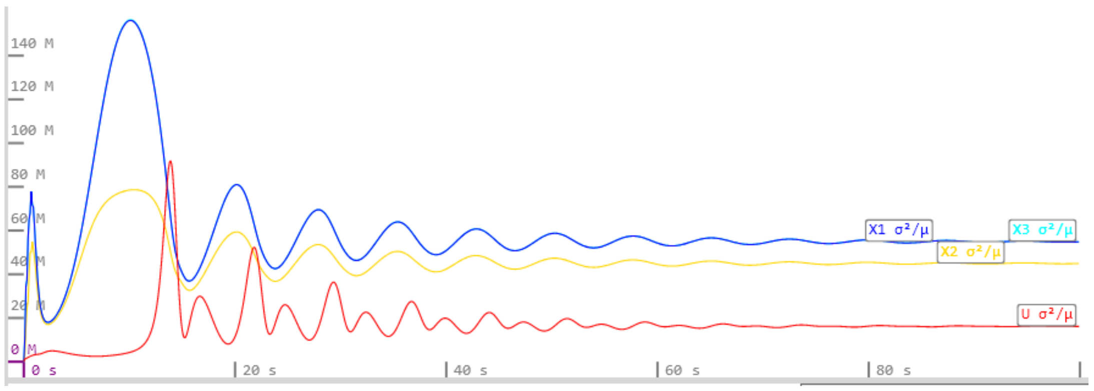


Figure 5.7: Noise Levels of AIF and BioSD Outputs in Kaemika (t = 100 s)
The parameter set used is the same as depicted in *Figure 5.6* and *Figure 4.20*, where $k_{in} = 190$, $k_1 = 1.9$, $k_2 = 1$, $k_3 = 80$, $\eta = 1050$, $\delta = 0.5$. For BioSD-I and BioSD-III, $b = 41$, while for BioSD-II, b is selected as 85.

An intriguing observation is that when the noise level of the input signal reaches its universal peak, the noise of all three outputs exhibits a downward trend and becomes lower than the noise of the input signal. It is worth noting that despite the Fano factors of all signals exhibiting oscillations every 8 seconds, the local noise peaks in the input and output signals do not synchronize.

Parameter Modification

Similar to the previous scenario, the level of the Fano factor is also partially attributed to the parameter b. This relationship is demonstrated by the experiment depicted in *Figure 5.8*, where the value of parameter b for BioSD-III is increased to 85, matching the value of b used for BioSD-II. As a result, the noise level of BioSD-III significantly decreases and becomes much lower than that of BioSD-I and closer to that of BioSD-II. This enhancement in noise reduction also adds oscillation at the beginning (see *Figure 5.9*).

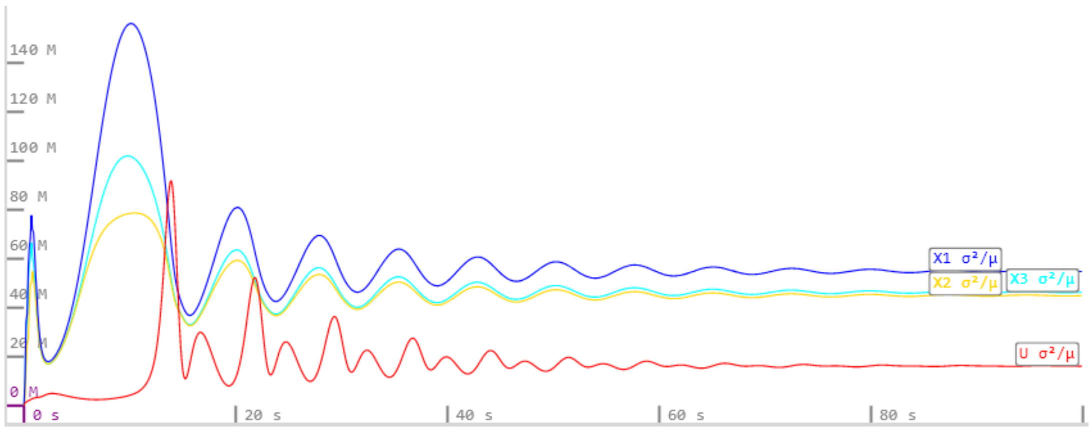


Figure 5.8: Noise Levels of AIF and BioSD Outputs in Kaemika (Modified *b*) (t =100 seconds)
The parameter set illustrated in *Figure 5.7* is also used here, except for b = 85 for BioSD-III.

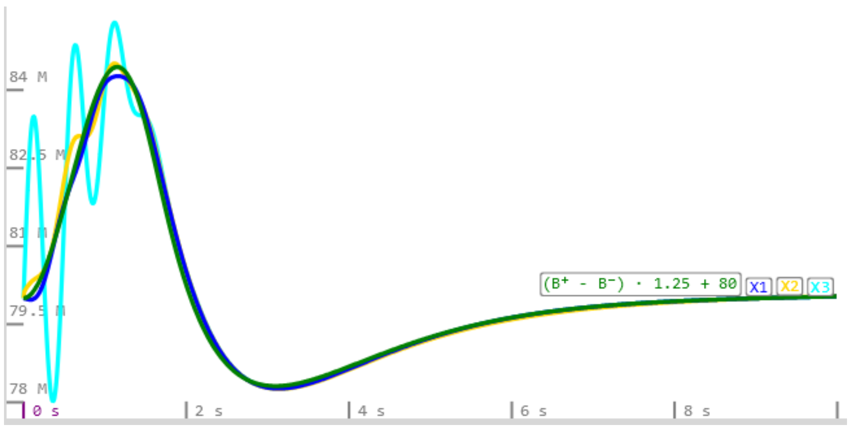


Figure 5.9: BioSD Outputs of AIF after Parameter Modification (t = 10 seconds)

Noise Suppression

To mitigate the noise levels illustrated in *Figure 5.7*, we employ the annihilation filter, as previously applied to hyperbolic tangent. Similarly, when tuning the filter parameters, we carefully balance the noise reduction against the impact on the output signal. One desirable filter setting is selected as shown in *Figure 5.10*.

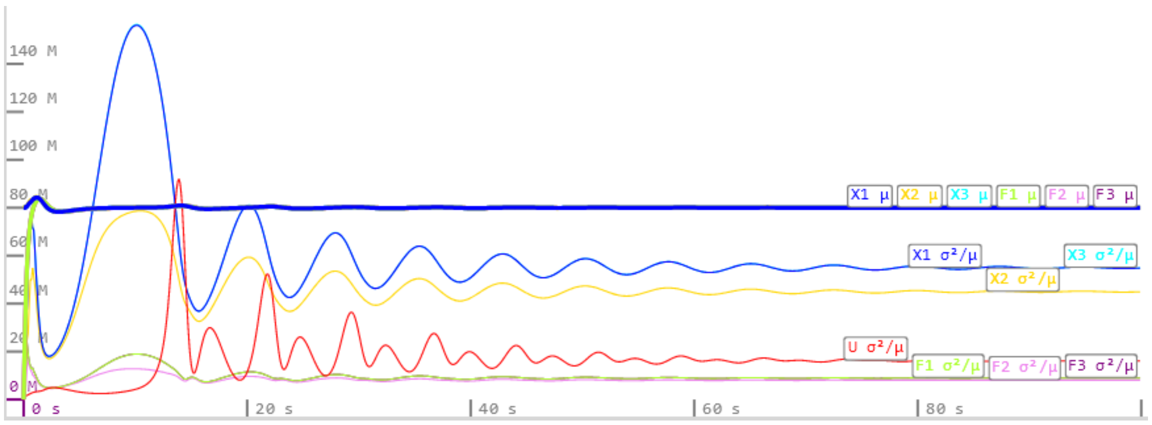


Figure 5.10: Noise Levels of AIF and BioSD with Noise Filter in Kaemika (t = 100 s)
The parameter set used for BioSD is the same as depicted in *Figure 5.7*. The parameters of the Noise Filter are set as follows: $r_1 = 100$, $r_2 = 1000$, $r_3 = 0.05$.

From the graph, it is evident that in this case, the filter can reduce the Fano factors of BioSD smaller than the noise level of the input signal for most of the time, although they still remain higher than the Poisson level. Importantly, it should be noted that the initial warm-up phases of the noise filter outputs cannot be eliminated through species initialization, as is done for BioSD.

For improved clarity, upon closer examination, it becomes apparent that the warm-up phase of the filter is time-consuming. This leads to the filtered results losing accurate tracking during the initial oscillation of AIF, resulting in delays and reduced peak values.

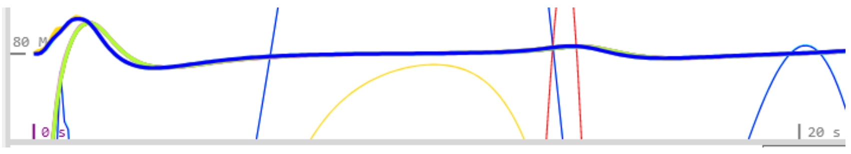


Figure 5.11: Zoomed-In Detail: Noise Levels of AIF and BioSD with Noise Filter in Kaemika (t = 20s)

5.3 Sinusoidal Wave

Since the sinusoidal wave is generated by a function instead of reactions, the Fano factor of it is the same as the hyperbolic tangent, which remains at 1 (indicated as $U\sigma^2/\mu$ in *Figure 5.12*), signifying a Poisson level of noise.

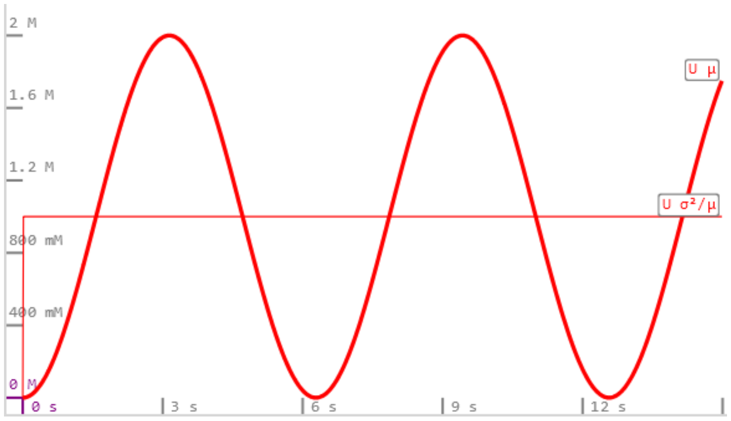


Figure 5.12: Noise Level of Sinusoidal Wave in Kaemika

The Fano factor for the outputs generated by BioSD is illustrated in *Figure 5.13*. The noise levels of all three BioSD are notably higher than the noise of the input signal as well as the Poisson level. BioSD-III has a slightly smaller value for parameter b compared to BioSD-I in this instance, and the noise of BioSD-III is marginally higher than that of BioSD-I. Meanwhile, BioSD-II, with the largest b value, exhibits the least noise.

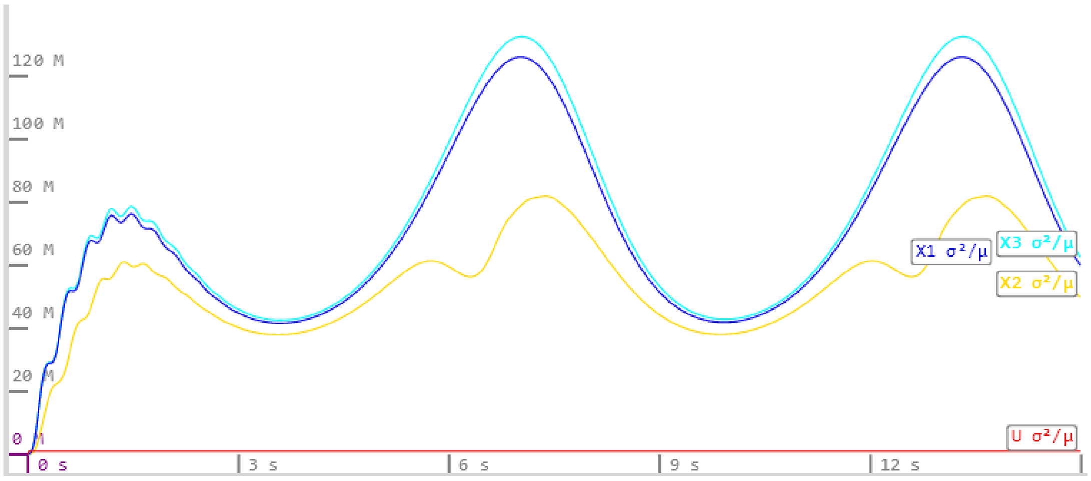


Figure 5.13: Noise Levels of Sinusoidal Wave and BioSD Outputs in Kaemika
The parameter set used is the same as depicted in *Figure 4.23*, where $k_{in} = 20$, $k_1 = 5$, $k_2 = 1$, $k_3 = 20$, $\eta = 10000$, $\delta = 0.5$. For BioSD-I, $b = 11$, for BioSD-II, b is selected as 15, and for BioSD-III, b is selected as 10.

Furthermore, it's worth noting that the noise patterns differ in this case. While BioSD-II's noise displays two local peaks per period, the other two differentiators exhibit only one peak per period.

Parameter Modification

To further investigate the influence of parameter b on noise, we increase the value of b for BioSD-III to 15, which is the same as the b of BioSD-II. The outcomes are displayed in *Figure 5.14*. As we can see from the figure, the Fano factor of BioSD-III reduces and is smaller than that of BioSD-I. This implies that a larger value of b might result in a smaller Fano factor. However, this enhancement also induces severe oscillations at the beginning (see *Figure 5.15*).

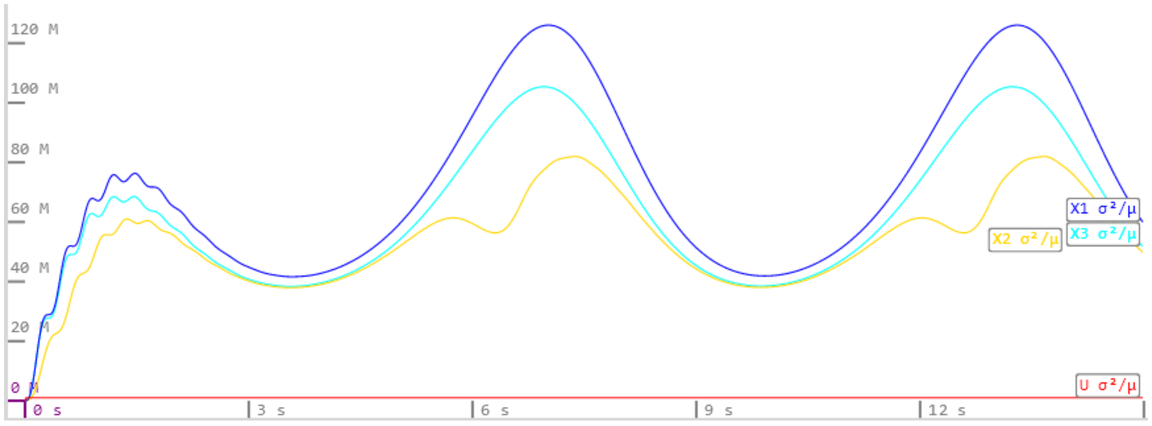


Figure 5.14: Noise Levels of Sinusoidal Wave and BioSD Outputs in Kaemika (Modified b)
The parameter set illustrated in *Figure 5.13* is also used here, except for $b = 15$ for BioSD-III.

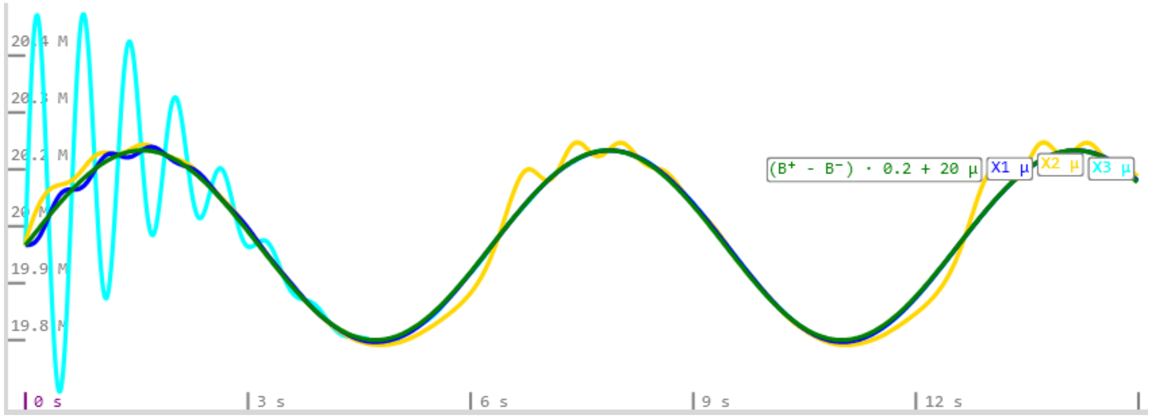


Figure 5.15: BioSD Outputs of Sinusoidal Wave after Parameter Modification

Noise Suppression

To reduce the noise exhibited in *Figure 5.13*, we implement the annihilation filter on the outcomes of BioSD. As observed in the preceding cases, tuning the filter involves a trade-off between signal quality and noise level. After testing, one desirable filter is set as below.

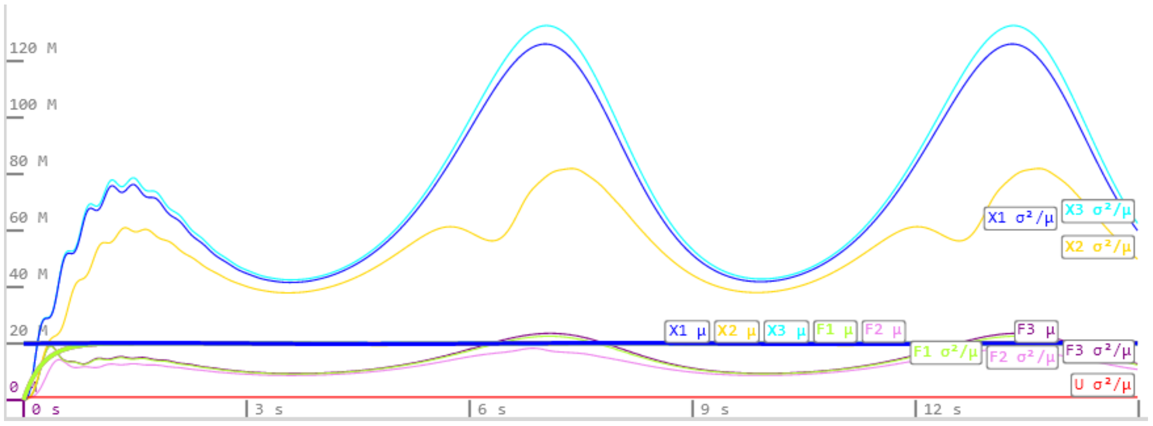


Figure 5.16: Noise Levels of Sinusoidal Wave and BioSD with Noise Filter in Kaemika
The parameter set used for BioSD is the same as depicted in *Figure 5.13*. The parameters of the Noise Filter are set as follows: $r_1 = 100$, $r_2 = 1000$, $r_3 = 0.2$.

The graphical representation reveals that, in this particular scenario, the filter effectively curbs the Fano factors of BioSD outputs without introducing considerable delays. It's important to emphasize that while the aim may involve further minimizing noise levels to align with the Poisson level, achieving this might necessitate a trade-off in terms of signal integrity. This is partially due to the long initial warm-up phase associated with the filter's operation, which cannot be diminished through species initialization, unlike in the case of BioSD. The amplitudes of sinusoidal oscillations are very small compared to the amplitude of the warm-up rise, which also makes it challenging to see the sinusoidal oscillations in Kameika.

Chapter 6

Conclusion and Discussion

In this study, the central focus was on investigating the noise characteristics of BioSD as a tool for calculating derivatives in biochemical systems. Through comprehensive evaluations involving various input signals and noise analysis techniques, several important insights have been garnered.

The key finding of this research underscores that BioSD has a tendency to significantly amplify noise levels above the Poisson level or even the original noise level of the input signals. This observation underscores the necessity of incorporating noise reduction strategies into the BioSD framework.

Employing noise filters to mitigate the noise levels in the outputs of BioSD has proven effective in achieving noise reduction. However, it's important to note that while these enhancements curtail noise, they also introduce certain delays to the system. Achieving Fano factors at the Poisson level could potentially lead to information loss due to these delays, thus warranting careful consideration.

Interestingly, the utility of the conditions presented by *Equation 3.11* for parameter tuning and accuracy assessment is shown to be limited in practical scenarios. The condition fails to account for all parameters, and smaller ϵ does not necessarily indicate better performance.

Therefore, without any other help in tuning, the time and effort to find the optimal parameter sets are no less, and we acknowledge that we did not find the optimal parameter set for all differentiators, potentially influencing the order of accuracy and noise levels. As a result, we refrain from providing a definitive conclusion regarding the most accurate and least noisy differentiator. However, we suggest that for applications requiring a simple and accurate biochemical differentiator, BioSD-I would be a suitable choice. On the other hand, for situations prioritizing lower noise, BioSD-II would be the preferred option.

An intriguing observation is the robustness of BioSD's performance, as it still shows high accuracy when BioSD is not precisely tuned, or even when the input signals oscillate continuously and fail to reach a steady state. This resilience enhances the practical viability of BioSD across a spectrum of real-world situations where ideal tuning or input signals might not always be achievable.

Furthermore, we observe that larger values of parameter b might lead to reduced Fano factors for some of the BioSD.

Future research directions could encompass the exploration of machine learning techniques to optimize parameter sets for BioSD. Additionally, refining the noise filter to eliminate warm-up phases could enhance signal tracking and viewing clarity.

Bibliography

[1] Emmanouil Alexis, Luca Cardelli, and Antonis Papachristodoulou. On the design of a pid bio-controller with set point weighting and filtered derivative action, 2022.

[2] Emmanouil Alexis, Carolin CM Schulte, Luca Cardelli, and Antonis Papachristodoulou. Biomolecular mechanisms for signal differentiation. Apr 2021.

[3] Robyn P. Araujo and Lance A. Liotta. The topological requirements for robust perfect adaptation in networks of any size. *Nature Communications*, 9(1), May 2018.

[4] Georgios Argyris, Alberto Lluch Lafuente, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. Reducing boolean networks withnbsp;backward boolean equivalence. *Computational Methods in Systems Biology*, page 1–18, 2021.

[5] Jacob Beal. Signal-to-noise ratio measures efficacy of biological computing devices and circuits. *Frontiers in Bioengineering and Biotechnology*, 3, Jun 2015.

[6] Rakesh P. Borase, D. K. Maghade, S. Y. Sondkar, and S. N. Pawar. A review of pid control, tuning methods and applications. *International Journal of Dynamics and Control*, 9(2):818–827, 2020.

[7] Corentin Briat, Ankit Gupta, and Mustafa Khammash. Antithetic integral feedback ensures robust perfect adaptation in noisy biomolecular networks. *Cell Systems*, 2(1):15–26, Jan 2016.

[8] K.H.J Buschow and B.P. Burton. *4.3 Monte Carlo Methods*, page 6493–6502. Elsevier, 2 edition, 2001.

[9] Luca Cardelli. Kaemika app: Integrating protocols and chemical simulation. *Computational Methods in Systems Biology*, page 373–379, 2020.

[10] Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. Erode: A tool for the evaluation and reduction of ordinary differential equations. *Tools and Algorithms for the Construction and Analysis of Systems*, page 310–328, 2017.

[11] Luca Cardelli, Milan Češka, Martin Fränzle, Marta Kwiatkowska, Luca Laurenti, Nicola Paoletti, and Max Whitby. Syntax-guided optimal synthesis for chemical reaction networks. *Computer Aided Verification*, 10427:375–395, Jul 2017.

[12] Suchana Chakravarty and Attila Csikász-Nagy. Systematic analysis of noise reduction properties of coupled and isolated feed-forward loops. *PLOS Computational Biology*, 17(12):e1009622, 2021.

[13] Avigdor Eldar and Michael B. Elowitz. Functional roles for noise in genetic circuits. *Nature*, 467(7312):167–173, Sep 2010.

[14] Johan Elf and Måns Ehrenberg. Fast evaluation of fluctuations in biochemical networks with the linear noise approximation. *Genome Research*, 13(11):2475–2484, Nov 2003.

[15] Erickson Fajiculay and Chao-Ping Hsu. Localization of noise in biochemical networks. *ACS Omega*, 8(3):3043–3056, Jan 2023.

[16] U. Fano. Ionization yield of radiations. ii. the fluctuations of the number of ions. *Physical Review*, 72(1):26–29, Jul 1947.

[17] Steven A. Frank. Basic control architecture. *Control Theory Tutorial*, page 19–27, May 2018.

[18] Kate Franz, Abhyudai Singh, and Leor S. Weinberger. Lentiviral vectors to study stochastic noise in gene expression. *Methods in Enzymology*, 497:603–622, 2011.

[19] Kate Franz, Abhyudai Singh, and Leor S. Weinberger. Lentiviral vectors to study stochastic noise in gene expression. *Methods in Enzymology*, 497:603–622, 2011.

[20] VAN KAMPEN N G. *Chapter IV Markov Processes*. Elsevier, 2007.

[21] Saul I. Gass and Michael C. Fu, editors. *Chapman-Kolmogorov Equations*, pages 160–161. Springer US, Boston, MA, 2013.

[22] Hao Ge and Hong Qian. *Chemical Master Equation*, pages 396–399. Springer New York, New York, NY, 2013.

[23] Daniel T Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22(4):403–434, Apr 1976.

[24] Daniel T. Gillespie. *Gillespie Algorithm for Biochemical Reaction Simulation*, pages 1–5. Springer New York, New York, NY, 2016.

[25] Cato Maximilian Guldberg and Peter Waage. Studier over affiniteten. *Forhandlinger I Videnskabs-selskabet I Christiania (Transactions of the Scientific Society in Christiania) (in Danish)*, page 35–45, 1864.

[26] Sayuri K. Hahl and Andreas Kremling. A comparison of deterministic and stochastic modeling approaches for biochemical reaction systems: On fixed points, means, and modes. *Frontiers in Genetics*, 7, Aug 2016.

[27] Wolfgang Halter, Richard M Murray, and Frank Allgöwer. Analysis of primitive genetic interactions for the design of a genetic signal differentiator. *Synthetic Biology*, 4(1), 06 2019. ysz015.

[28] Wolfgang Halter, Zoltan A. Tuza, and Frank Allgöwer. Signal differentiation with genetic networks 1 1supported by the research cluster bw (www.bwbiosyn.de) of the ministry for science, research and art baden-württemberg. *IFAC-PapersOnLine*, 50(1):10938–10943, Jul 2017.

[29] Carl M. Harris. Markov chains. *Encyclopedia of Operations Research and Management Science*, page 930–934, Jan 2016.

[30] Hans Heesterbeek. The law of mass-action in epidemiology: A historical perspective. *Ecological Paradigms Lost*, page 81–105, 2005.

[31] Bo Hu, David A. Kessler, Wouter-Jan Rappel, and Herbert Levine. Effects of input noise on a simple biochemical switch. *Physical Review Letters*, 107(14), Sep 2011.

[32] Tobias Jahnke and Wilhelm Huisinga. Solving the chemical master equation for monomolecular reaction systems analytically. *Journal of Mathematical Biology*, 54(1):1–26, Sep 2006.

[33] A.M. Johansen. Markov chain monte carlo. In Penelope Peterson, Eva Baker, and Barry McGaw, editors, *International Encyclopedia of Education (Third Edition)*, pages 245–252. Elsevier, Oxford, third edition edition, 2010.

[34] James Keener and James Sneyd. Mathematical physiology. *Interdisciplinary Applied Mathematics*, 1998.

[35] Terry Kenakin. The mass action equation in pharmacology. *British Journal of Clinical Pharmacology*, 81(1):41–51, Dec 2015.

[36] Diego Alberto Lara Bustillos, Leonardo López-Hernández, N. Ramírez-Cruz, Edna M. Hernández, Ruben Fossion, Enrique López-Moreno, Carlos E. Vargas, and V. Velázquez. Nuclear energy level complexity: Fano factor signature of chaotic behavior of nearest-neighbor time-series analysis. *Physical Review C*, 102(4), Oct 2020.

[37] Luca Laurenti. *Noise and Prediction in Molecular Systems*. PhD thesis, 2018.

[38] Luca Laurenti, Attila Csikasz-Nagy, Marta Kwiatkowska, and Luca Cardelli. Molecular filters for noise reduction. *Biophysical Journal*, 114(12):3000–3011, Jun 2018.

[39] Sarangam Majumdar and Sisir Roy. Role of noise in synthetic biology. *Microbial Communication*, page 159–162, Sep 2020.

[40] Saurabh Modi, Supravat Dey, and Abhyudai Singh. Noise suppression in stochastic genetic circuits using pid controllers. *PLOS Computational Biology*, 17(7), 2021.

[41] K. Oishi and E. Klavins. Biomolecular implementation of linear i/o systems. *IET Systems Biology*, 5(4):252–260, 2011.

[42] Erdi P. and Toth J. *Mathematical models of chemical reactions: Theory and applications of deterministic and stochastic models*. Manchester University Press, 1989.

[43] Alida Palmisano and Corrado Priami. *Stochastic Simulation Algorithm*, pages 2009–2010. Springer New York, New York, NY, 2013.

[44] Penelo Peterson, Eva L. Baker, Barry McGaw, and Adam Johansen. *Monte Carlo Methods*, page 296–303. Elsevier/AP, 3 edition, 2010.

[45] Gleb Pogudin and Xingjian Zhang. Interpretable exact linear reductions via positivity. *Computational Methods in Systems Biology*, page 91–107, 2021.

[46] Miguel Prado Casanova. Noise and synthetic biology: How to deal with stochasticity? *NanoEthics*, 14(1):113–122, Apr 2020.

[47] Kamil Rajdl, Petr Lansky, and Lubomir Kostal. Fano factor: A potentially useful information. *Frontiers in Computational Neuroscience*, 14, Nov 2020.

[48] Gopinath Rao. Change in economics: Law of mass action, Feb 2020.

[49] Thomas S Shimizu, Yuhai Tu, and Howard C Berg. A modular gradient-sensing network for chemotaxis in escherichia coli revealed by responses to time-varying stimuli. *Molecular Systems Biology*, 6(1):382, Jun 2010.

[50] Kyle Siegrist. 15. markov processes.

[51] David Soloveichik, Georg Seelig, and Erik Winfree. Dna as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393–5398, Jan 2010.

[52] Zhen Tang, Zhixiang Yin, Luhui Wang, Jianzhong Cui, Jing Yang, and Risheng Wang. Solving 0–1 integer programming problem based on dna strand displacement reaction network. *ACS Synthetic Biology*, 10(9):2318–2330, 2021.

[53] Philipp Thomas, Hannes Matuschek, and Ramon Grima. How reliable is the linear noise approximation of gene regulatory networks? *BMC Genomics*, 14(S4), Oct 2013.

[54] Yiwei Wang, Chun Liu, Pei Liu, and Bob Eisenberg. Field theory of reaction-diffusion: Law of mass action with an energetic variational approach. *Physical Review E*, 102(6), Dec 2020.

[55] Max Whitby, Luca Cardelli, Marta Kwiatkowska, Luca Laurenti, Mirco Tribastone, and Max Tschaikowski. Pid control of biochemical reaction networks. *IEEE Transactions on Automatic Control*, 67(2):1023–1030, 2022.

Supplementary Information

S1 Kaemika Code

Single-Rail Differentiator

```
//===================================
// This network computes the derivative
// of an input signal A as the difference
// of two output signals B⁺ B⁻.
// I.e. the (positive or negative) output
// is represented as the difference of
// two positive signals.
//===================================

// input A, output B⁺ - B⁻

network deriv1(species A B⁺ B⁻) {
        number r = 1000
        number s = 1000
        amount B⁺ B⁻ @ 0 M
        species A' @ observe(A) M

        // A' tracks A by r
        A ->{r} A + A';      A' ->{r} Ø

        // B⁺ traks r·A by s
        A ->{r·s} A + B⁺;      B⁺ ->{s} Ø

        // B⁻ tracks r·A' by s
        A' ->{r·s} A' + B⁻;      B⁻ ->{s} Ø

        // Normalization
        B⁺ + B⁻ -> Ø
}

// Example: computing the derivative
// of the exponential function:
// d(e^t)/dt    ( = e^t )

// Input: A grows exponentially
// from 1: A(t) = e^t
species A @ 1M
A -> 2 A

// Output: differentiate A into B⁺ - B⁻

species {B⁺, B⁻}
deriv1(A, B⁺, B⁻)

// plot input, output, and also the true
// exponential function for comparison

report A, B⁺ - B⁻, exp(time)
equilibrate for 3
```

Hyperbolic Tangent

```
// Turn a positive function into a waveform
network Signal(species X, function f) {
     Ø ->{{100·f()}} X;      X ->{100} Ø
}

species U @ 0 M

Signal(U, λ(){1+tanh(time-5)})

report U
equilibrate for 10
```

Antithetic Integral Feedback (AIF)

```
species C1, C2, Y1, Y2@ 0M

number v1 = 2
number v2 = 2
number v3 = 4
number v4 = 2
number v5 = 12
number v6 = 1
number v7 = 1

Ø       ->    C1 {v1}
C1    ->    C1 + Y1 {v2}
Y1    ->    Y1 + Y2 {v3}
Y2    ->    Y2 + C2 {v4}
C1 + C2 ->    Ø {v5}
Y1    ->    Ø {v6}
Y2    ->    Ø {v7}

report Y2
equilibrate for 100
```

Sinusoidal Waveform

```
// Turn a positive function into a waveform

network Signal(species X, function f) {
     Ø ->{{1000·f()}} X;     X ->{1000} Ø
}

species A @ 0 M

number freq = 1
number ampl = 1
number phase = 3·pi/2

Signal(A, λ(){ampl·(1+sin(freq·time+phase))})

report A                  // plot test signal

equilibrate for 20
```