Constructing Dampened LTI Systems Generating Polynomial Bases

Andreas Stöckel

Centre for Theoretical Neuroscience

University of Waterloo

March 9, 2021

Abstract

We present an alternative derivation of the LTI system underlying the Legendre Delay Network (LDN). To this end, we first construct an LTI system that generates the Legendre polynomials. We then dampen the system by approximating a windowed impulse response, using what we call a "delay re-encoder". The resulting LTI system is equivalent to the LDN system. This technique can be applied to arbitrary polynomial bases, although there typically is no closed-form equation that describes the state-transition matrix.

1 Introduction

The Delay Network, originally proposed by Voelker and Eliasmith (2018), is a recurrent neural network capable of delaying an input signal $u(t)$ by θ seconds. Voelker (2019) points out that the impulse response of the linear time-invariant (LTI) system underlying the delay network traces out the shifted Legendre polynomials. We hence refer to this network as the *Legendre Delay Network* (LDN), and to the LTI system underlying the LDN as the *LDN system*.

The LDN has been derived from the Padé approximants of a Laplace domain delay $e^{-\theta s}$ and a subsequent conditioning coordinate transformation. From this perspective, the relationship to the Legendre polynomials is rather surprising.

Gu, Dao, Ermon, Rudra, and Re (2020) have proposed LTI systems similar to the LDN system for other polynomial bases and various window functions. These systems are derived in the opposite direction of Voelker's original approach. Given a polynomial basis and a window function, Gu et al. derive an LTI system that realizes this basis with the desired weighting applied.

In this report, we use a similar approach. Although the method presented here has been developed independently, our approach differs from Gu et al. mostly in terms of presentation. Our goal is to provide a simple derivation assuming a minimal mathematical background; readers are encouraged to consult Gu et al. for a more general treatment of the topic.

1

2 Deriving the LDN system

We derive the LDN system $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$ in two steps. First, we construct an LTI system $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$ that traces out the Legendre polynomials as its impulse response over the interval $[0, \theta]$. Second, we derive a matrix $\bar{\boldsymbol{\Gamma}}$ that decodes a delayed signal $u(t - \theta)$ from the state vector $\mathbf{m}(t)$ and re-encodes this delayed function in terms of the Legendre basis. We call this matrix $\bar{\boldsymbol{\Gamma}}$ "delay re-encoder". Subtracting $\bar{\boldsymbol{\Gamma}}$ from $\bar{\mathbf{A}}$ results in a dampened impulse response; the LDN system is simply given as $\hat{\mathbf{A}} = \bar{\mathbf{A}} - \bar{\boldsymbol{\Gamma}}$ and $\hat{\mathbf{B}} = \bar{\mathbf{B}}$.

2.1 The LDN System

Before we discuss an alternative derivation of the LDN system $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$, we should first define this system itself. Note that we discuss a scaled version of the original LDN system. The impulse response of the scaled system matches the shifted Legendre polynomials for $q \to \infty$ (see Section 2.4).

More precisely, compared to the original system, we divide each state dimension $i \in \{1, \dots, q\}$ by $2i + 1$. This can be easily accomplished by constructing a diagonal matrix $\mathbf{M}$ of scaling factors and using this $\mathbf{M}$ as a coordinate transformation. In other words, it holds $\hat{\mathbf{A}} = \mathbf{M}\hat{\mathbf{A}}'\mathbf{M}^{-1}$ and $\hat{\mathbf{B}} = \mathbf{M}\hat{\mathbf{B}}'$, where $\hat{\mathbf{A}}'$, $\hat{\mathbf{B}}'$ is the original LDN system (cf. Voelker, 2019, Section 6.3.1, pp. 133-135).

Definition 1 (LDN System). Let $\mathbf{m} \in \mathbb{R}^q$, $\hat{\mathbf{A}} \in \mathbb{R}^{q \times q}$ and $\hat{\mathbf{B}} \in \mathbb{R}^{q \times 1}$. The scaled LDN system is given as

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta\mathbf{m}(t) = \hat{\mathbf{A}}\mathbf{m}(t) + \hat{\mathbf{B}}u(t)$$

$$\left(\hat{\mathbf{A}}\right)_{ij} = (2j - 1)\begin{cases} -1 & \text{if } i \leq j \text{ or } i + j \text{ is even}, \\ 1 & \text{if } i > j \text{ and } i + j \text{ is odd}, \end{cases} \tag{1}$$

$$\left(\hat{\mathbf{B}}\right)_i = (-1)^{i+1}.$$

This equation is implemented in the function `mk_ldn_lti`*.*

The window-length θ determines how fast the system evolves.

Notation: Due to an unfortunate shortage of letters in the Latin alphabet, we resort to a somewhat confusing, but hopefully consistent notation. In particular, we use the following symbols to denote linear time-invariant (LTI) system matrices:

Symbols	Description
$\mathbf{A}$, $\mathbf{B}$	General LTI system with feedback matrix $\mathbf{A}$ and input matrix $\mathbf{B}$
$\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$	The scaled Legendre Delay Network (LDN) system
$\hat{\mathbf{A}}'$, $\hat{\mathbf{B}}'$	The original LDN system proposed by Voelker (2019)
$\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$, $\bar{\boldsymbol{\Gamma}}$	The Legendre system and the corresponding delay re-encoder

Source code: We provide Python code for many of the equations in this report (see margin notes). This code can be found here:

`https://github.com/astoeckel/dlop_ldn_function_bases`

Example 1. The LDN system for $\theta = 1$ and $q = 6$ is given as

$$
\hat{\mathbf{A}} = \begin{pmatrix}
-1 & -3 & -5 & -7 & -9 & -11 \\
1 & -3 & -5 & -7 & -9 & -11 \\
-1 & 3 & -5 & -7 & -9 & -11 \\
1 & -3 & 5 & -7 & -9 & -11 \\
-1 & 3 & -5 & 7 & -9 & -11 \\
1 & -3 & 5 & -7 & 9 & -11
\end{pmatrix}, \qquad
\hat{\mathbf{B}} = \begin{pmatrix}
1 \\ -1 \\ 1 \\ -1 \\ 1 \\ -1
\end{pmatrix}.
$$

The impulse response of this system is depicted in Figure 1B.

For finite q, the impulse response *approximates* the Legendre polynomials. In contrast to the Legendre polynomials, the impulse response quickly converges to zero. In other words, the impulse response can be thought of as a "dampened" version of the Legendre polynomials. This almost finite impulse response[1] is the most practically useful property of the LDN system. Feeding an input signal $u(t)$ into the LDN system can be thought of as performing an online basis transformation. That is, the state $\mathbf{m}(t)$ approximately represents a window of the input signal $u_{[t-\theta,t]}$ in terms of a linear combination of the Legendre polynomials; we discuss this in more detail below. In general, such transformations are known as "sliding transformations".

2.2 The Legendre System

Instead of *approximating* the Legendre polynomials, we can construct an LTI system $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$ that *perfectly* traces out the Legendre polynomials as its impulse response. In other words, the system "generates" the Legendre polynomials. As we will see in the next subsection, we can then subtract a dampening term $\mathbf{\Gamma}$ from this "Legendre system" to obtain the LDN.

Lemma 1. *The impulse response of the linear time-invariant system $\frac{d}{dt}\theta\mathbf{m}(t) = \bar{\mathbf{A}}\mathbf{m}(t) + \bar{\mathbf{B}}u(t)$ with $\mathbf{m} \in \mathbb{R}^q$, $\bar{\mathbf{A}} \in \mathbb{R}^{q \times q}$ and $\bar{\mathbf{B}} \in \mathbb{R}^{q \times 1}$*

$$
(\bar{\mathbf{A}})_{ij} = (4j-2) \begin{cases} 0 & \text{if } i \leq j \text{ or } i+j \text{ is even}, \\ 4j-2 & \text{if } i > j \text{ and } i+j \text{ is odd}, \end{cases} \quad (\bar{\mathbf{B}})_i = (-1)^{i+1}, \quad (2)
$$

This equation is implemented in the function `mk_leg_lti`*.*

are the first q shifted Legendre polynomials $\tilde{P}_n(t\theta^{-1})$ over $t \in [0, \theta]$.

Example 2. The LTI system constructing the first six shifted Legendre polynomials as its impulse response for $\theta = 1$ is given as

$$
\bar{\mathbf{A}} = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & 0 & 0 & 0 \\
0 & 6 & 0 & 0 & 0 & 0 \\
2 & 0 & 10 & 0 & 0 & 0 \\
0 & 6 & 0 & 14 & 0 & 0 \\
2 & 0 & 10 & 0 & 18 & 0
\end{pmatrix}, \qquad
\bar{\mathbf{B}} = \begin{pmatrix}
1 \\ -1 \\ 1 \\ -1 \\ 1 \\ -1
\end{pmatrix}.
$$

---

[1] We refer to the impulse response as "almost finite", because it decays exponentially for $t > \theta$. Hence, technically, the system does *not* have a finite impulse response. However, this is negligible for all practical purposes.
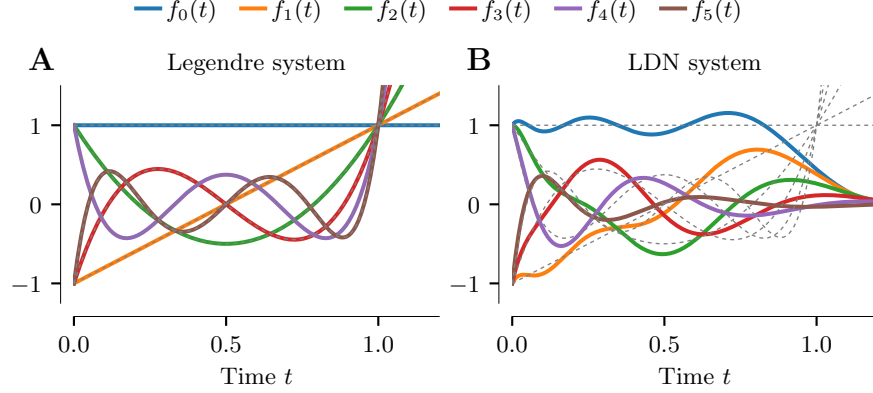
Figure 1: Impulse responses $f_n(t)$ of the Legendre and LDN LTI systems for $\theta = 1$ of order $q = 6$. Dashed grey lines correspond to the first six shifted Legendre polynomials $\tilde{P}_n(t)$. **(A)** Impulse response of the Legendre system. The impulse response perfectly traces out the Legendre polynomials. The system diverges for $t \to \infty$. **(B)** Impulse response of the LDN system. The system quickly converges to zero for $t \to \infty$.

The impulse response of this system is depicted in Figure 1A.

Proof of Lemma 1. As pointed out by Gu et al. (2020, Appendix B.1.1), and originally described in Arfken and Weber (2005, Chapter 12.2, p. 751), Legendre polynomials fulfil the following recurrence relation with respect to their derivative:

$$\frac{\mathrm{d}}{\mathrm{d}t}\big(P_{n+1}(t) - P_{n-1}(t)\big) = (2n + 1)P_n(t).$$

Substituting in $\tilde{P}_n(t\theta^{-1}) = P_n((2t - 1)\theta^{-1})$ and rearranging we get

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta\tilde{P}_{n+1}(t\theta^{-1}) = (4n + 2)\tilde{P}_n(t\theta^{-1}) + \frac{\mathrm{d}}{\mathrm{d}x}\tilde{P}_{n-1}(t\theta^{-1})$$
$$= (4n + 2)\tilde{P}_n(t\theta^{-1}) + (4(n - 2) + 2)\tilde{P}_{n-2}(t\theta^{-1}) + \ldots.$$

This recurrence relation terminates with $\tilde{P}_0$ or $\tilde{P}_1$ depending on whether n is even or odd. Crucially, this recurrence relation implies that the differential of the nth Legendre polynomial can be expressed as a linear combination of the preceding Legendre polynomials. Let $\mathbf{m}(t) = (\tilde{P}_0(t\theta^{-1}), \ldots, \tilde{P}_{q-1}(t\theta^{-1}))$. We can now write the above equation as a vector-matrix product

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta\mathbf{m}(t) = \bar{\mathbf{A}}\mathbf{m}(t),$$

where $\bar{\mathbf{A}}$ is as defined in eq. (2). The vector $\bar{\mathbf{B}} = \big(\tilde{P}_0(0), \ldots, \tilde{P}_{q-1}(0)\big)$ defines the initial value of each state dimension in response to a Dirac pulse $u(t) = \delta(t)$. $\quad\square$

4

Sliding transformations Feeding a signal $u(t)$ into the Legendre system convolves this signal with the Legendre polynomials. The state vector $\mathbf{m}(t) = (m_0(t), \ldots, m_{q-1}(t))$ is the convolution between the impulse response $\exp(\bar{\mathbf{A}}t)\bar{\mathbf{B}} = \tilde{P}_n(t\theta^{-1})$ and the input signal $u(t)$:

$$m_n(t) = \int_0^t \left(\exp(\bar{\mathbf{A}}\tau)\bar{\mathbf{B}} \right)_{n+1} u(t - \tau) \, \mathrm{d}\tau = \int_0^t \tilde{P}_n(\tau\theta^{-1}) u(t - \tau) \, \mathrm{d}\tau. \tag{3}$$

At $t = \theta$ a segment of the input $u_{[0,\theta]}$ is stored in $\mathbf{m}(\theta)$ as a linear combination of the basis functions $\tilde{P}_n$. At this point, the convolution operator is mathematically equivalent to the inner product between the Legendre polynomials and the input $u(t)$ up to $t = \theta$, i.e., $m_n(\theta) = \langle u_{[0,\theta]}, \tilde{P}_n \rangle$. However, for $t > \theta$, the Legendre polynomials $P_n(t\theta^{-1})$ and our impulse response quickly diverge.

If we were able to limit the impulse response to a window $[0, \theta]$ (i.e., $t > \theta \Rightarrow \exp(\mathbf{A}\tau)\mathbf{B} = 0$), then the state vector $\mathbf{m}(t)$ would always represent a slice of the most recent input history window $u_{[t-\theta,t]}$. This suggests a convenient technique for implementing a "sliding transformation". Advancing the LTI system computes the spectral coefficients $\mathbf{m}(t)$ describing a recent input slice. Using more general notation, we would optimally like to implement a windowed convolution

$$m_n(t) = \int_0^\theta f_n(\tau) u(t - \tau) \, \mathrm{d}\tau = \langle f_n, u_{[t-\theta,t]} \rangle, \tag{4}$$

where the f_n are the desired basis functions over $[0, \theta]$ generated by the LTI system $\mathbf{A}$, $\mathbf{B}$, i.e., $f_n(t) = (\exp(\mathbf{A}t)\mathbf{B})_{n+1}$ for $n \in \{0, \ldots, q-1\}$.

2.3 Dampening Through Information Erasure

To enforce a finite impulse response—to *dampen* the system—we somehow need to prevent the impulse response from evolving past θ. We first discuss how to accomplish this under the assumption that we have access to a perfect delay—i.e., we have a recording of the input signal over the past θ seconds. In a second step we approximate a perfect delay by decoding a delayed version of the input signal from the system state $\mathbf{m}(t)$ itself.

Constructing a rectangle window using a perfect delay If we have access to a perfect delay line of length θ, we can easily construct a system with a finite impulse response. All we need to do this, is to subtract the delayed $u(t-\theta)$ from the system state $\mathbf{m}(t)$ using an "encoding vector" $\mathbf{e}(\theta)$. Put differently, we erase information about $u(t)$ older than θ seconds from the system state. We hence refer to this method as "information erasure".

Lemma 2. *Let $\mathbf{A}$, $\mathbf{B}$ describe an LTI system and let $u(t)$ be some input signal. The impulse response of the following modified system is unchanged compared to the original LTI system for $0 \leq t < \theta$ but zero for all $t \geq \theta$*

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{m}(t) = \mathbf{A}\mathbf{m}(t) + \mathbf{B}u(t) - \mathbf{e}(\theta)u(t - \theta), \quad \text{where } \mathbf{e}(\theta) = \exp(\mathbf{A}\theta)\mathbf{B}. \tag{5}$$

Proof of Lemma 2. Consider the impulse response, i.e., $u(t) = \delta(t)$, where $\delta(t)$ is a Dirac pulse. The system state is unchanged for $t < \theta$ as $\delta(t - \theta) = 0$ for all $t \neq \theta$; hence the impulse response of the system is $\mathbf{m}(t) = \exp(\mathbf{A}t)\mathbf{B}$ for $0 \leq t < \theta$. At $t = \theta$, according to the definition of the Dirac pulse, we subtract $\mathbf{e}(\theta) = \exp(\mathbf{A}\theta)\mathbf{B}$ from the system state $\mathbf{m}(\theta)$, exactly when $\mathbf{m}(\theta)$ is equal to $\mathbf{e}(\theta)$. The resulting $\mathbf{m}$ is zero and remains zero as $u(t) = \delta(t) = 0$ for $t \neq 0$. $\square$

Approximating a delayed input signal   In practice, we may not have access to the delayed input signal $u(t - \theta)$. However, remember that our goal is to construct a system such that the system state $\mathbf{m}(t)$ represents a slice of u over the interval $[t - \theta, t]$. This representation is in terms of a linear combination of the q polynomial basis functions. We can thus decode an approximate $u(t - \theta')$ from the state $\mathbf{m}(t)$ using a delay decoder $\mathbf{d}(\theta')$.

Definition 2 (Delay decoder). Let $\hat{u} : [0, \theta] \longrightarrow \mathbb{R}$ be a linear combination of q basis functions $f_n : [0, \theta] \longrightarrow \mathbb{R}$ with weighting coefficients ξ_n. Let furthermore $\mathbf{m}(t) = (m_0(t), \dots, m_{q-1}(t))$ denote the state of a q-dimensional linear dynamical system with impulse responses f_n and input $\hat{u}$ (cf. eq. 4), i.e.,

$$m_n(t) = \int_0^t f_n(\tau)\hat{u}(t - \tau)\,\mathrm{d}\tau = \int_0^t f_n(\tau)\sum_{m=0}^{q-1}\xi_m f_m(t - \tau)\,\mathrm{d}\tau.$$

Then $\mathbf{d}(\theta') = (d_0(\theta'), \dots, d_{q-1}(\theta'))$ is called a *delay decoder* if

$$
\begin{aligned}
\hat{u}(\theta - \theta') &= \sum_{m=0}^{q-1}\xi_m f_m(\theta - \theta') = \big\langle \mathbf{d}(\theta'), \mathbf{m}(\theta) \big\rangle \\
&= \sum_{n=0}^{q-1} d_n(\theta')\int_0^\theta f_n(\tau)\sum_{m=0}^{q-1}\xi_m f_m(\theta - \tau)\,\mathrm{d}\tau.
\end{aligned}
\tag{6}
$$

Lemma 3. *For the Legendre polynomials* $\tilde{P}_n(t\theta^{-1})$, *the delay decoder* $\mathbf{d}(\theta')$ *is*

$$\big(\mathbf{d}(\theta')\big)_m = \frac{2m+1}{\theta}\tilde{P}_m(\theta'\theta^{-1}).$$

Proof. Let $f_n(t) = \tilde{P}_n(t\theta^{-1})$. Combining the proposed delay decoder $\mathbf{d}(\theta')$ with the delay decoder definition yields

$$
\begin{aligned}
\big\langle \mathbf{d}(\theta'), \mathbf{m}(\theta) \big\rangle &= \sum_{n=0}^{q-1}\frac{2n+1}{\theta}\tilde{P}_n(\theta'\theta^{-1})\int_0^\theta \tilde{P}_n(\tau\theta^{-1})\sum_{m=0}^{q-1}\xi_m\tilde{P}_m((\theta - \tau)\theta^{-1})\,\mathrm{d}\tau \\
&= \sum_{n=0}^{q-1}\sum_{m=0}^{q-1}\xi_m\frac{2n+1}{\theta}\tilde{P}_n(\theta'\theta^{-1})\int_0^\theta \tilde{P}_n(\tau\theta^{-1})\tilde{P}_m((\theta - \tau)\theta^{-1})\,\mathrm{d}\tau.
\end{aligned}
$$

The integral can be simplified using two properties of the Legendre polynomials

$$\int_0^1 \tilde{P}_n(\tau)\tilde{P}_m(\tau)\,\mathrm{d}\tau = \frac{\delta_{nm}}{2m+1}, \qquad\qquad \text{(Orthogonality)}$$

$$\tilde{P}_n(t) = (-1)^n\tilde{P}_n(1 - t), \qquad\qquad \text{(Parity)}$$

6

where δ_{ij} is the Kronecker delta. Continuing the above set of equations we get

$$\langle \mathbf{d}(\theta'), \mathbf{m}(\theta) \rangle = \sum_{n=0}^{q-1} \sum_{m=0}^{q-1} \xi_m \frac{2n+1}{\theta} \tilde{P}_n(\theta'\theta^{-1})(-1)^n \frac{\theta \delta_{mn}}{2n+1}$$

$$= \sum_{n=0}^{q-1} \xi_n (-1)^n \tilde{P}_n(\theta'\theta^{-1}) = \hat{u}(\theta - \theta'). \qquad \square$$

Given the concept of a "delay decoder" we can construct an approximate version of eq. (5) that reconstructs $u(t-\theta)$ from the system state $\mathbf{m}(t)$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{m}(t) = \mathbf{A}\mathbf{m}(t) + \mathbf{B}u(t) - \mathbf{e}(\theta)\langle \mathbf{d}(\theta), \mathbf{m}(t) \rangle. \qquad (7)$$

The approximate nature of this equation comes from $u(t)$ not necessarily being expressible as a linear combination of the basis functions, as required in our definition of "delay decoder". We discuss the consequences of this in more detail in Section 2.4; before we get there, we first simplify eq. (7) a little further by collapsing the encoder $\mathbf{e}(\theta)$ and the delay decoder $\mathbf{d}(\theta)$ into a single matrix $\mathbf{\Gamma}$.

Definition 3 (Delay re-encoder)**.** Let $\mathbf{e}(\theta)$ be the encoding vector containing the impulse response of a linear system $\mathbf{A}$, $\mathbf{B}$ at time θ, i.e.,

$$\mathbf{e}(\theta) = \exp(\mathbf{A}\theta)\mathbf{B} = \big(f_0(\theta), \dots, f_{q-1}(\theta)\big), \qquad (8)$$

and $\mathbf{d}(\theta)$ denote the delay decoder of order q for the basis functions f_n produced by the impulse response of the dynamical system. Furthermore, let "$\odot$" denote the outer product. Then $\mathbf{\Gamma} = \mathbf{e}(\theta) \odot \mathbf{d}(\theta)$ with $\mathbf{\Gamma} \in \mathbb{R}^{q \times q}$ is the *delay re-encoder* of order q. This matrix can be used to write eq. (7) more compactly

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{m}(t) = \big(\mathbf{A} - \mathbf{\Gamma}\big)\mathbf{m}(t) + \mathbf{B}u(t)$$

Example 3. For the shifted Legendre polynomials (and, correspondingly, the Legendre system) the delay re-encoder $\bar{\mathbf{\Gamma}}$ is simply given as

$$\big(\bar{\mathbf{\Gamma}}\big)_{ij} = e_i(\theta)d_j(\theta) = \frac{2j+1}{\theta}\tilde{P}_{i-1}(\theta\theta^{-1})\tilde{P}_{j-1}(\theta\theta^{-1}) = \frac{2j+1}{\theta}. \qquad (9)$$

For $\theta = 1$ and $q = 6$ the delay re-encoder is

$$\bar{\mathbf{\Gamma}} = \begin{pmatrix} 1 & 3 & 5 & 7 & 9 & 11 \\ 1 & 3 & 5 & 7 & 9 & 11 \\ 1 & 3 & 5 & 7 & 9 & 11 \\ 1 & 3 & 5 & 7 & 9 & 11 \\ 1 & 3 & 5 & 7 & 9 & 11 \\ 1 & 3 & 5 & 7 & 9 & 11 \end{pmatrix}.$$

Notice that, in general, $\hat{\mathbf{A}} = \bar{\mathbf{A}} - \bar{\mathbf{\Gamma}}$. That is, the LDN feedback matrix $\hat{\mathbf{A}}$ (eq. 1) is simply the difference between the Legendre system feedback matrix $\bar{\mathbf{A}}$ (eq. 2) and the delay re-encoder $\bar{\mathbf{\Gamma}}$ for the Legendre system (eq. 9).

2.4 Impulse response in the limit $q \to \infty$

The "information erasure" technique from eq. (7) uses a delay decoder $\mathbf{d}(\theta')$ to reconstruct a delayed input signal $u(t - \theta')$ from the system state $\mathbf{m}(t)$. In general, this reconstruction is an approximation, since most $u(t)$ cannot be expressed as a linear combination of the basis functions f_n generated by the LTI system—a condition we assumed to be true when we derived $\mathbf{d}(\theta')$.

Aliasing of $u(t)$ onto $\hat{u}(t)$ Fortunately, this is less of a problem as it may seem—convolving with the impulse response automatically represents the signal $u(t)$ with respect to f_n. To see why this is, consider the case of a *perfectly windowed* LTI system generating a function basis (using eq. (5) from Lemma 2). The system continuously maps ("aliases") a segment of the input signal $u_{[t-\theta,t]}$ onto a signal $\hat{u}_{[t-\theta,t]}$. This $\hat{u}_{[t-\theta,t]}$ is a linear combination of the basis functions f_n generated by the LTI system and is exactly the function obtained when applying the delay decoder $\mathbf{d}(\theta')$ to the system state $\mathbf{m}(t)$

$$\hat{u}_{[t-\theta,t]}(\theta') = \langle \mathbf{d}(\theta'), \mathbf{m}(t) \rangle = \sum_{n=0}^{q-1} \xi_n f_n(\theta - \theta') \quad \text{for } 0 \le \theta' \le \theta \,.$$

Note that this aliasing process merely discards components of the input signal that cannot be represented in the function basis—there are no destructive aliasing artefacts as those encountered when violating the Nyquist-Shannon theorem.

Aliasing of an impulse input   As we saw in Figure 1, the impulse response of the LDN system—i.e., the Legendre system with a delay re-encoder (eq. 7)—does not abruptly disappear for $t > \theta$. Furthermore, the impulse response no longer perfectly traces out the Legendre polynomials.

Correspondingly, functions are no longer represented with respect to the originally desired function basis (e.g., the shifted Legendre polynomials), but with respect to the actual impulse response of the system. While we derive the delay re-encoder under the assumption that the system generates a desired function basis, subtracting the delay re-encoder from the feedback matrix inevitably causes this assumption to be violated.

Hence, there is no guarantee that this method will actually generate an (almost) finite impulse response. For example, this technique will not work properly for periodic bases such as the Fourier series, as $\mathbf{d}(\theta) = \mathbf{d}(0)$ in this case.

The discrepancy in the impulse responses stems from aliasing of the impulse input $u(t) = \delta(t)$. There is no finite LTI system that can produce a delayed Dirac pulse $\delta(t - \theta)$; in fact, the aliased function $\hat{u}(t)$ is likely non-zero almost everywhere and has finite energy at $t = 0$. The former discrepancy results in the ringing artefacts in the LDN system response (fig. 1B), the latter discrepancy in the only gradually disappearing impulse response for $t > \theta$.

The aliased impulses are depicted in Figure 2 for the Legendre system (with a perfect window applied) and the LDN system (i.e., the Legendre system with delay re-encoder $\bar{\mathbf{\Gamma}}$ applied). Although we do not have a rigorous proof for this, it

Perfect rectangle window **Approximated rectangle window**
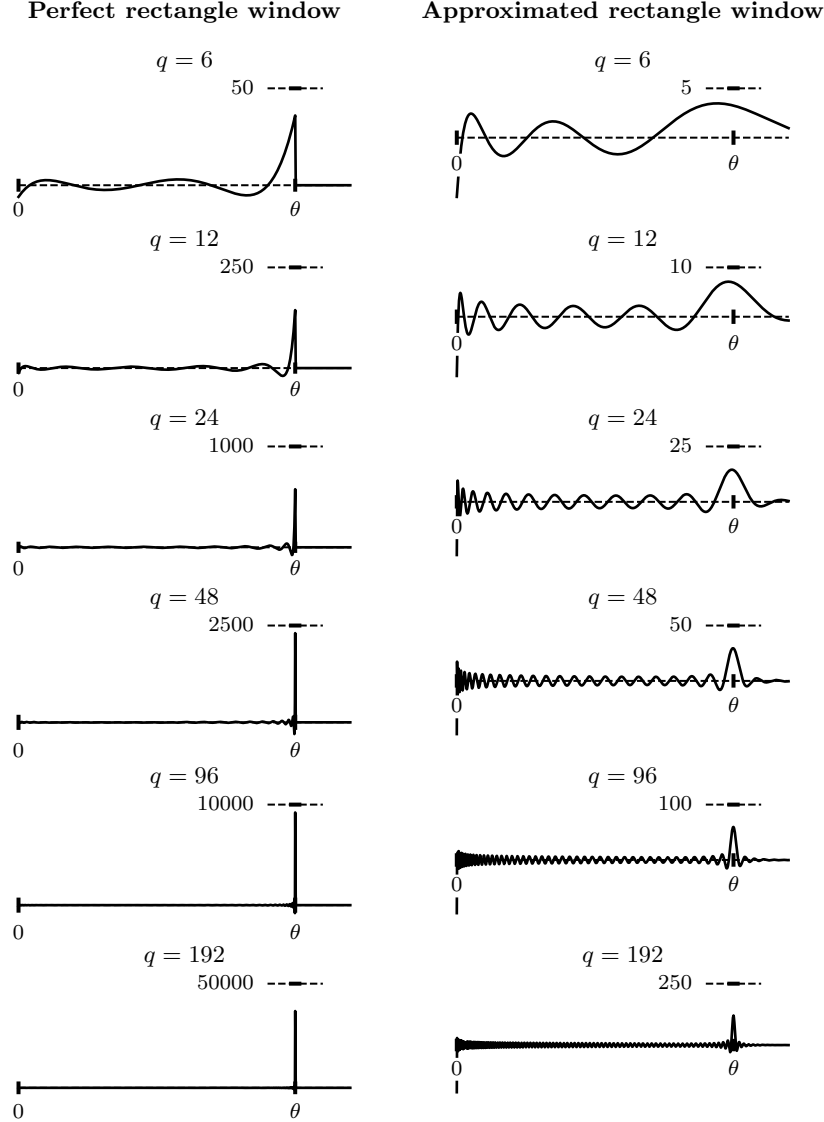
Figure 2: Decoding a delayed impulse input $u(t) = \delta(t)$ for the Legendre/LDN system with different basis function counts q. Individual graphs depict $\hat{u}_{[t-\theta,t]}(\theta) = \langle \mathbf{d}(\theta), \mathbf{m}(t) \rangle$, where $\mathbf{d}(\theta)$ is the delay decoder and $\mathbf{m}(t)$ is the impulse response of the LTI system. *Left:* Delay decoder applied to the impulse response of the Legendre system $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$ with a perfect window (eq. 5). *Right:* Delay decoder applied to the LDN system $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$. The decoded function gradually converges to a Dirac delta $\delta(t-\theta)$ in both cases, though convergence is slower for the LDN.

is safe to assume that in both cases the impulse response converges to a delayed Dirac pulse $\delta(t - \theta)$ for $q \to \infty$; in this case the LDN system and the perfectly windowed Legendre system are exactly equal.

3 Arbitrary Polynomial Bases

The techniques we developed in the previous section can be applied to arbitrary polynomial bases, and not just the Legendre polynomials. In particular, consider q polynomial basis functions p_n of order q

$$f_n(t) = p_n(t) = \sum_{k=0}^{q-1} \lambda_{n,k} t^k \qquad \text{for } n \in \{0, \ldots, q-1\}.$$

As we will see, it is quite trivial to construct a q-dimensional LTI system that generates the polynomials p_n as its impulse response. Unfortunately, computing the delay decoder $\mathbf{d}(\theta')$ tends to be numerically unstable.

3.1 Constructing an LTI system generating a basis

The approach used to prove Lemma 1 can be applied to any set of polynomial basis functions that fulfils two very mild conditions. First, the polynomials must be linearly independent. This condition is automatically fulfilled if the functions are indeed chosen from a single function basis. Second, there must be no $t \in [0, \theta)$ such that $p_n(t) = 0$ for all $n \in \{0, \ldots, q-1\}$—otherwise the impulse response would be extinguished at that point.

Solving for differentials The LTI system feedback matrix $\mathbf{A}$ maps the system state $\mathbf{m}(t)$ onto the differential $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{m}(t)$. For an impulse input, the state is supposed to be equal to the polynomials, i.e., $\mathbf{m}(t) = (p_0(t), \ldots, p_{q-1}(t))$. Furthermore, the differential of a polynomial is just another polynomial. It holds

$$\frac{\mathrm{d}}{\mathrm{d}t} p_n(t) = \frac{\mathrm{d}}{\mathrm{d}t} \sum_{k=0}^{q-1} \lambda_{n,k} t^k = \sum_{k=1}^{q-1} k \lambda_{n,k} t^{k-1}.$$

We hence need to solve for an $\mathbf{A}$ that linearly combines the polynomials p_n to form the individual derivatives:

$$\begin{pmatrix} \frac{\mathrm{d}}{\mathrm{d}t} p_0(t) \\ \vdots \\ \frac{\mathrm{d}}{\mathrm{d}t} p_{q-1}(t) \end{pmatrix} = \mathbf{A} \begin{pmatrix} p_0(t) \\ \vdots \\ p_{q-1}(t) \end{pmatrix}. \tag{10}$$

Linearly combining two polynomials simply generates a new polynomial where each coefficient k is a weighted sum of the orginal polynomial coefficients. We
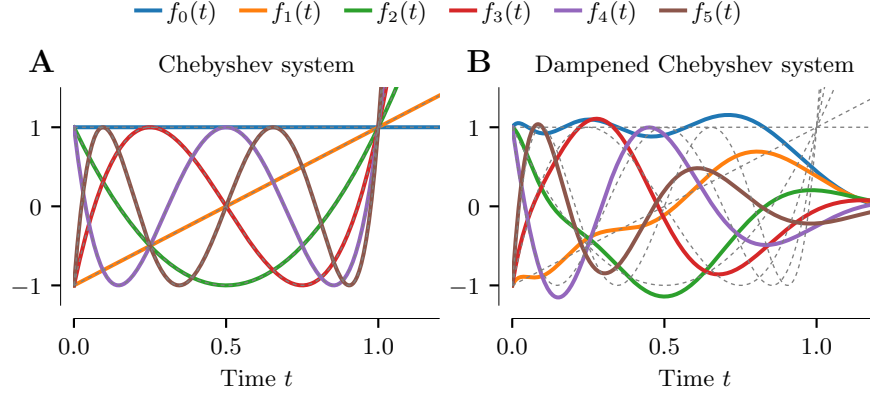
Figure 3: Impulse responses $f_n(t)$ of the Chebyshev system and the corresponding dampened version ($\theta = 1$ and $q = 6$). Dashed grey lines correspond to the first six shifted Chebyshev polynomials. **(A)** Impulse response of the Chebyshev system. **(B)** Impulse response of the dampened Chebyshev system.

can hence write eq. (10) as as system of linear equations and solve for $\mathbf{A}$

$$\sum_{m=0}^{q-1} a_{m,k}\lambda_{m,k} = \begin{cases} k\lambda_{n,k-1} & \text{if } k > 0, \\ 0 & \text{if } k = 0, \end{cases} \quad \text{for } n, k \in \{0, \ldots, q-1\},$$
and where $\left(\mathbf{A}\right)_{ij} = a_{i-1,j-1}.$

This equation is implemented in the function `mk_poly_basis_lti`.

The input matrix $\mathbf{B}$ is the x-intercept of the polynomials, i.e., $\left(\mathbf{B}\right)_i = \lambda_{i-1,0}$.

Example 4 (The Chebyshev system). The above method can be used to determine the LTI system generating the Chebyshev polynomials. The following is an extrapolation of numerical results; we did not prove that these equations are indeed correct.

$$\left(\mathbf{A}\right)_{ij} = (2i-2) \begin{cases} 0 & \text{if } i \leq j \text{ or } i+j \text{ is even}, \\ 1 & \text{if } j = 1 \text{ and } i > j \text{ and } i+j \text{ is odd}, \\ 2 & \text{if } j > 1 \text{ and } i > j \text{ and } i+j \text{ is odd}, \end{cases} \quad \left(\mathbf{B}\right)_i = (-1)^{i+1}.$$

These equations are implemented in `mk_cheb_lti`. *These matrices can also be approximated using the function* `mk_cheb_poly_basis` *and passing the result into* `mk_poly_basis_lti`.

For $q = 6$ the matrix $\mathbf{A}$ takes the following shape

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 & 0 & 0 \\ 6 & 0 & 12 & 0 & 0 & 0 \\ 0 & 16 & 0 & 16 & 0 & 0 \\ 10 & 0 & 20 & 0 & 20 & 0 \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ -1 \end{pmatrix}.$$

The Chebyshev polynomials and the impulse response of the corresponding LTI system are depicted in Figure 3A.

11

3.2 Computing the delay re-encoder

To compute the delay re-encoder $\mathbf{\Gamma} = \mathbf{e}(\theta) \odot \mathbf{d}(\theta)$ we need both the encoder $\mathbf{e}(\theta)$ and the delay-decoder $\mathbf{d}(\theta')$. While the encoder can be easily computed according to eq. (8), obtaining $\mathbf{d}(\theta)$ is a little trickier.

We present three methods for computing the delay decoder $\mathbf{d}(\theta)$. The first method directly solves for $\mathbf{d}(\theta)$; the second method inverts the polynomial basis. Both methods are unstable for large basis function counts q. The third method is to discretize the polynomials and to use a matrix pseudo-inverse to approximate an inverse of the polynomial basis, which can circumvent some of the instabilities.

Directly solving for $\mathbf{d}(\theta)$ Taking the definition of the delay re-encoder $\mathbf{d}(\theta)$ from eq. (6) and substituting f_n with a polynomial basis we obtain

$$
\sum_{i=0}^{q-1} d_i(\theta') m_i(\theta) = \sum_{i=0}^{q-1} d_i(\theta') \int_0^\theta f_i(\tau) \hat{u}(\theta - \tau)\, \mathrm{d}\tau
$$

$$
= \sum_{i=0}^{q-1} d_i(\theta') \int_0^\theta f_i(\tau) \sum_{j=0}^{q-1} \xi_j f_j(\theta - \tau)\, \mathrm{d}\tau
$$

$$
= \sum_{i=0}^{q-1} d_i(\theta') \int_0^\theta \left(\sum_{k=0}^{q-1} \lambda_{i,k} \tau^k \right) \sum_{j=0}^{q-1} \xi_j \left(\sum_{k=0}^{q-1} \lambda_{j,k} (\theta - \tau)^k \right) \mathrm{d}\tau
$$

$$
= \sum_{j=0}^{q-1} \xi_j \sum_{i=0}^{q-1} d_i(\theta') \int_0^\theta \left(\sum_{k=0}^{q-1} \lambda_{i,k} \tau^k \right) \left(\sum_{k=0}^{q-1} \lambda_{j,k} (\theta - \tau)^k \right) \mathrm{d}\tau
$$

$$
\overset{!}{=} \sum_{j=0}^{q-1} \xi_j \sum_{k=0}^{q-1} \lambda_{j,k} (\theta - \theta')^k \quad \text{for all } \xi_j \in \mathbb{R}
$$

$$
= \hat{u}(\theta - \theta')\,,
$$

Substituting $p_j(\theta - \tau)$ with a polynomial $p'_j(\theta) = p_j(\theta - \tau)$ with coefficients $\rho_{j,k}$:

$$
\sum_{k=0}^{q-1} \lambda_{j,k} (at + b)^k = \sum_{k=0}^{q-1} \underbrace{\left(\sum_{n=k}^{q-1} \binom{n}{k} a^k b^{n-k} \lambda_{j,n} \right)}_{\rho_{j,k}} t^k = \sum_{k=0}^{q-1} \rho_{j,k} t^k \,,
$$

where $a = -1$, $b = \theta$. Since the above equality holds for all possible input signals $\hat{u}$ (i.e., any combination of ξ_j) we get for all $j \in \{0, \ldots, q-1\}$

$$
\sum_{i=0}^{q-1} d_i(\theta') \int_0^\theta \left(\sum_{k=0}^{q-1} \lambda_{i,k} \tau^k \right) \left(\sum_{k=0}^{q-1} \rho_{j,k} \tau^k \right) \mathrm{d}\tau = \sum_{k=0}^{q-1} \lambda_{j,k} (\theta - \theta')^k \,. \tag{11}
$$

The integral can be evaluated in closed form. It holds

$$
\int_0^\theta \left(\sum_{k=0}^{q-1} \lambda_{i,k} \tau^k \right) \left(\sum_{k=0}^{q-1} \rho_{j,k} \tau^k \right) \mathrm{d}\tau = \sum_{k=0}^{q-1} \sum_{n=0}^{q-1} \frac{\theta^{1+n+k}}{1+n+k} \rho_{j,n} \lambda_{i,k} = (\rho_j)^T \mathbf{Q}_\theta^q \lambda_i \,,
$$

where ρ_j and λ_i are vectors of polynomial coefficients. We can write eq. (11) in matrix-vector equation

$$\mathbf{P}\mathbf{Q}_\theta^q \mathbf{\Lambda}^T \mathbf{d}(\theta') = \mathbf{y}^{\theta'}$$

where $\mathbf{\Lambda}, \mathbf{P}$ denote matrices of coefficients $\lambda_{i,k}$, $\rho_{j,k}$, respectively. The vector $\mathbf{y}^{\theta'}$ is the right-hand side of eq. (11).

Solving this system of equations tends to be numerically unstable. This is partially due to the magnitude of the polynomial coefficients, and partially due to $\mathbf{Q}_\theta^q$. For $\theta = 1$, this matrix is known as the "Hilbert matrix", which is notoriously ill-conditioned—although a closed-form inverse exists (Press et al., 2007, Section 2.8, p. 94; Choi, 1983). For $q = 4$ the Hilbert matrix is

$$\mathbf{Q}_1^4 = \begin{pmatrix} \frac{1}{1} & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{pmatrix}.$$

Inverting the polynomial basis Above, we directly solved for a single delay decoder $\mathbf{d}(\theta')$. Instead, we can compute an "inverse" of the q polynomial basis functions. Evaluating this inverse basis at any point θ' will result in the corresponding delay decoder.

The idea is simply that we would like to construct a set of polynomials $\tilde{p}_n$ with coefficients $\rho_{n,k}$ such that

$$\langle p_i, \tilde{p}_j \rangle = \int_0^\theta \left(\sum_{k=0}^{q-1} \lambda_{i,k} \tau^k \right) \left(\sum_{k=0}^{q-1} \rho_{j,k} \tau^k \right) \mathrm{d}\tau = \delta_{ij} \,,$$

where δ_{ij} is the Kronecker delta. As above, the integral can be expressed using the scaled Hilbert matrix $\mathbf{Q}_\theta^q$. We get the matrix-vector equation

$$\mathbf{P}\mathbf{Q}_\theta^q \mathbf{\Lambda}^T = \mathbf{\Lambda}\mathbf{Q}_\theta^q \mathbf{P}^T = \mathbf{I} \,,$$

where, again, $\mathbf{\Lambda}$ and $\mathbf{P}$ denote matrices of polynomial coefficients $\lambda_{i,k}$, $\rho_{j,k}$, respectively. This equation can be easily solved for the polynomial coefficients $\mathbf{P}$, which in turn define $\tilde{p}_n$. Evaluating this set at any point θ' yields the corresponding delay decoder $\mathbf{d}(\theta)$.[2]

Unfortunately, and unsurprisingly, this method tends to struggle with exactly the same numerical instabilities as the previous method.

Discrete inversion of the polynomial basis A rather naive approach that tends to work remarkably well is to perform the above basis inversion technique on a discretized basis. At least under the assumption that the polynomials themselves can be evaluated at any point $t \in [0, \theta]$, this tends to be more stable.

---

[2]We have not proved this formally, though showing this *should* be relatively straight forward.

Let $\mathbf{L} \in \mathbb{R}^{q \times N}$, where N is the number of samples and

$$(\mathbf{L})_{ij} = p_{i-1} \left(\frac{\theta(j-1)}{N-1} \right) .$$

The delay decoder corresponds to the individual rows of the pseudo-inverse $\mathbf{L}^+ \in \mathbb{R}^{N \times q}$ scaled by a factor N/θ:

$$\mathbf{L}^+ = \mathbf{L}^T \left(\mathbf{L}\mathbf{L}^T \right)^{-1} , \quad \text{where } \mathbf{d}(\theta') = \frac{N}{\theta} \mathbf{L}_i^+ \text{ and } i = \left\lfloor \frac{(N-1)\theta'}{\theta} \right\rfloor + 1 .$$

This method provides reasonably precise results for N on the order of 10^6.

Example 5 (The Chebyshev system delay decoder). Using any of the above methods, we obtain the following delay decoder $\mathbf{d}(\theta)$ for the Chebyshev system and $q = 6$:

$$\mathbf{d}(\theta) = \begin{pmatrix} \mathbf{4.79} & \mathbf{8.20} & \mathbf{9.23} & \mathbf{7.38} & \mathbf{8.12} & \mathbf{5.41} & \mathbf{5.87} \end{pmatrix}^T .$$

This equation is implemented in the functions `mk_poly_basis _rencoder` and `mk_poly_sys _rencoder`.

The dampened Chebyshev system is depicted in Figure 3B. Note that in contrast to the Legendre polynomials, the individual delay decoder coefficients depend on the number of basis functions q. This is generally the case for polynomial bases such as the Chebyshev basis that are not orthogonal with respect to a unit weighting. While there seems to be some systematicity in the Chebyshev delay decoder with respect to q, it is not immediately apparent how to fit a precise closed-form solution to these results.

Note that polynomial bases that are orthogonal with respect to a unit weighting can be seen as a rotated version of the Legendre polynomials; there is little to be gained from not using the Legendre polynomials.

4 Conclusion

We presented an alternative derivation of the Legendre Delay Network (LDN) from an LTI system generating the Legendre polynomials. We show that this system can be turned into the LDN by subtracting a delay re-encoder $\mathbf{\Gamma}$ from the feedback matrix $\mathbf{A}$. This operation approximates a rectangle window applied to the impulse response. This method can be used in conjunction with arbitrary polynomial bases, although numerical instabilities may make this difficult in practice.

Another downside of the presented method is that there is no formal guarantee that subtracting the delay re-encoder $\mathbf{\Gamma}$ from the feedback matrix $\mathbf{A}$ will actually result in an (almost) finite impulse response. While this seems to be the case for the polynomial bases we tested—including random polynomial bases—it would be nice to have more formal guarantees. Future work in this direction should establish a set of sufficient and necessary conditions pertaining the basis functions f_n such that the dampened system $\mathbf{A} - \mathbf{\Gamma}$ generating these functions is guaranteed to decay to zero.

Acknowledgements

References

Arfken, George B. and Hans J. Weber (2005). *Mathematical Methods for Physicists*. Sixth Edition. Burlington, MA: Elsevier Academic Press. 1182 pp. ISBN: 0-12-088584-0.

Choi, Man-Duen (1983). "Tricks or Treats with the Hilbert Matrix". In: *The American Mathematical Monthly* 90.5, pp. 301–312. DOI: `10.1080/00029890.1983.11971218`.

Gu, Albert et al. (2020). *HiPPO: Recurrent Memory with Optimal Polynomial Projections*. arXiv: `2008.07669 [cs.LG]`.

Press, William H. et al. (2007). *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. 3rd ed. New York, NY, USA: Cambridge University Press. ISBN: 0-521-88068-8 978-0-521-88068-8.

Voelker, Aaron R. (2019). "Dynamical Systems in Spiking Neuromorphic Hardware". PhD thesis. Waterloo, ON: University of Waterloo. URL: `http://hdl.handle.net/10012/14625`.

Voelker, Aaron R. and Chris Eliasmith (Mar. 2018). "Improving Spiking Dynamical Networks: Accurate Delays, Higher-Order Synapses, and Time Cells". In: *Neural Computation* 30.3, pp. 569–609. DOI: `10.1162/neco_a_01046`.

Revisions

(1) February 9, 2021. First internal version.

(2) February 26, 2021. Publication to arXiv.

(3) March 9, 2021. Corrected some typos.