Statement of Changes

Aaron R. Voelker and Chris Eliasmith

August 27, 2017

Dear Editor,

Thank you and both reviewers for taking the time to read and assess our manuscript, titled "Improving Spiking Dynamical Networks: Accurate Delays, Higher-order Synapses, and Time Cells". We have attempted to address all of the comments, and believe the work has improved in clarity of exposition and contact with related literature as a result of your feedback. We have included our replies inline below, along with descriptions of the corresponding changes.

In addition, if you require a more detailed account of all changes, we refer you to `https://github.com/arvoelke/delay2017/commits/master` which contains a `diff` of our Python code and LaTeX since the date of the initial submission.

Reviewer # 1

Neurons in the brain as well as neurons in many neuromorphic computing architectures communicate with spikes, short impulses sent and received at discrete times. In contrast, machine learning approaches e.g. towards fitting dynamical systems with neural networks usually use continuously coupled units. Transferring such approaches to the domain of spiking neural networks may deepen our understanding of the brain and allow for novel, perhaps more efficient ways of artificial computing.

The current manuscript provides a step towards both these aims. It shows how feedback and output filtering as typical for sophisticated biological synapses can be compensated in linear filtering dynamical systems by changes in the coupling matrices and it shows how limitations of neuromorphic computing setups can be precisely accounted for. Furthermore, it shows that taking into account synaptic/axonal delays as present in biological neural systems improves the abilities of neural networks to achieve short-term memory tasks. For this, it combines the novel considerations in the domain of linear filtering systems with an established translation scheme between continuous and spiking dynamical systems, the neural engineering framework.

The manuscript is very interesting, well written, concise, timely and important.

I have only a few suggestions for improvements:

1. It would be important to discuss other models for efficient short term memory such as the models of White and Sompolinsky PRL 04, Ganguli, Huh and Sompolinsky PNAS 08, Goldman Neuron 09, which also use linear dynamical systems to perform an input delay task and could also be readily implemented using the NEF.

> **1.1. Thank you for the encouraging evaluation, and highlighting this work for us to discuss. We have reviewed the above papers by Sompolinsky, Goldman, and colleagues, in addition to some of their follow-up papers. From what we understand the Sompolinsky papers appear to rely heavily on discrete time-stepping. Their approach is to use N rate-neurons coupled by single time-step delays in order to encode at most N bits of information. Although there are numerous extensions, none seem to move away from this fundamental approach. On the other hand, our method decouples the simulation time-step from the delay length. We now cite these two papers when discussing Reservoir Computing (RC; see reply 1.3) in the**

introduction. The Goldman (2009) paper discusses a different kind of memory: a *perfect integrator*, as opposed to a delay line. In a very early iteration of our work, we experimented with a modification of Goldman's feedforward approach to readout a delay, but this was less robust. The "gamma model" by Bert de Vries and Jose C. Principe (see reply 2.1) and delay-coupled Reservoir Computing (see reply 2.2) are also relevant. Specifically, we have discussed these approaches, and others, in the introduction, as well as incorporated consideration of them into sections 3 and 4.4.

2. Eq. (2) deviates from the usual NEF by adding a convolution with h on the left hand side. It would be good to explain why this modification was chosen.

1.2. When we present this elsewhere, we often think of $\mathbf{x}(t)$ as already being filtered, in which case the convolution with $h(t)$ on the left-hand side is implicit. Here we felt that including it explicitly better reflects the setup of the network. Note, this modification also appears in equation 3 for consistency. Then this segues better into Principle 3 without needing to reconsider the variables from equation 3 any differently. For instance, the recurrent function $f(\mathbf{x}) = (\tau A + I)\mathbf{x}$ is decoded from the activities and then filtered by h. Also, it is more accurate to apply the filter to both sides, since in general the (time-invariant) decoders alone cannot compensate for the filter applied to the activities. Principle 3 then exploits this filter by treating it as the basis for the system-level dynamics. We have added this note to section 2.1, but left the beginning of this discussion out of the manuscript in order to keep it self-contained.

3. p. (13), after Eq. (9) might want to discuss previous attempts to solve the problem with spiking neural networks such as Maass, Natschläger and Markram Neural Computation 02 (cf. also Wallace, Maei and Latham Neural Comp 13).

1.3. We have a separate paper currently in progress that features experiments comparing our delay network to standard RC networks. Our experiments show that even hyper-optimized RC networks are very inaccurate when using spiking LIF neurons (a.k.a., Liquid State Machines) to implement a delay line. Likewise, when using randomly connected networks of sigmoid units (a.k.a., Echo State Networks), again using hyper-optimized time-constants, they are also sub-optimal for longer delay lengths relative to the frequency of the input signal, in comparison to our approach using the same rate-based neuron model. Thus, RC methods were found to be neither accurate nor efficient in comparison to our approach. This should be expected since RC methods are not optimized to have any dynamics in particular. We aim to publish the aforementioned work after final submission of this manuscript. In the mean time, we have modified our manuscript to cite these alternative approaches in the introduction, and to mention this is the topic of future work.

4. p. 13 bottom: The sentence "the power spectrum of a signal may be expressed as a nonlinear combination of delays" seems unclear.

1.4. Changed to "the spectrogram of a signal may be computed by a nonlinear combination of delays".

5. p. 14 bottom, the paragraph "Conversely, a transfer function can be converted" should also provide references to the literature.

1.5. Cited: Brogan, W. L. (1991). *Modern Control Theory (3rd Edition).*

6. p. 15 "representational issues" should be made concrete.

1.6. Qualified as "...leads to numerical issues in the representation (i.e., dimensions that grow exponentially in magnitude)...".

7. Fig. 3: Why is the additional low-pass filtering of the activities with tau=0.2 s done? To get rid of peaks generated by the rapid increase of the synaptic currents?

1.7. Previously, this was being done (with $\tau = 0.02\,\mathrm{s}$) because we were decoding $y(t) = C\mathbf{x}(t)$ from spikes. Since this is a weighted combination of spikes, some filter is required. However, as may be evident, we can use the PSC from the recurrent connection to get $\mathbf{x}(t)$—without needing any additional filtering—and then transform this by C. We do this now in the updated Fig. 3, and the RMSE only increased slightly ($0.043 \rightarrow 0.048$) due to the lack of additional filtering.

8. Fig. 4, right hand side panel: The header of the panel, "Decoding at theta=Frequency^-1", seems unclear.

1.8. This means $\theta = \frac{1}{\texttt{Frequency}}$ or equivalently $\texttt{Frequency} = \frac{1}{\theta}$. Please note that the axes of both graphs are also scaled in analogous ways to make everything unitless. This is mentioned in the body of the text.

9. p. 20: Is there an intuitive explanation for the qualitative changes in the shape of trajectory lengths?

1.9. The slope of the curve is $\|\dot{\mathbf{x}}(t')\|$, which is the amount of "energy" in the impulse response $\mathbf{x}(t)$. Thus, some intuition for low dimensions is that there are only enough degrees of freedom in a linear system to start fast (given the impulse) and continuously slow down while being exponentially stable (think of a decaying 2D oscillator). In theory, as $q \rightarrow \infty$, the impulse response of the system becomes $y(t) = \delta_\theta(t)$. It then makes sense that the curve for this case would tend toward a Heaviside step function (shifted by $t = \theta$). In between these two extremes, the curve is being pushed into the shape of this step function! We have omitted this intuition from the manuscript since it would require a more careful proof to be made concrete. Lastly, note that we tried extending the time axis past the point of θ, but this makes no qualitative changes to Figure 6. Apart from this intuition, it is very difficult for us to get an analytical handle on $\int_0^t \|\dot{\mathbf{x}}(t')\|\, dt'$. Some Lyapunov theory may help here, but we have not yet pushed this avenue. We intend to rely on the current intuition and numerics until there is some need to do otherwise.

10. p. 21, lower paragraph: adding a figure or mentioning "data not shown" may improve clarity.

1.10. Changed, specifically to: "This means that other delays will simply stretch or compress the impulse response linearly in time (not shown)."

12. p. 22 second paragraph: "introducing the synapse model means replacing the integrator s^-1 with H(s)." This should be better explained; it is, in fact, not really explained in section 2.3 either. Of course, the reader can work it out, e.g. as follows: We do a Laplace transform of Eqs. 7 after endowing all A,B,C,D, X, and Y with an upper index H. Then we replace the s in front of the left hand side X(s) in the upper equation by H(s)^-1, because the synaptic filtering (multiplication with H(s)) should yield X(s) after applying it to the output of A^H X^H(s)+B^H U(s). This is equivalent to formally replacing the free factor s by H(s)^-1 in F.

1.11. Yes, which can also be thought of as a substitution of variables. $s^{-1} = \frac{1}{s}$ is an integrator, when understood in the context of block-diagrams as an operator (see Figures 1 and 2). Then, $H(s)$ is also an operator, that takes the place of the integrator. Thus, we have the change of variables: replacing s^{-1} with $H(s)$, or, equivalently, replacing s with $H(s)^{-1}$. This is also the idea behind Principle 3: using the dynamics of the synapse to perform the role of the integrator. We have updated section 2.3 to better explain the idea behind this substitution, in particular

connecting better to the substitution shown between Figures 1 and 2. We have also clarified this step in the beginning of section 4. Also see reply 2.4 for related changes and additional information.

13. Eq. (26): It remains unclear how the formulas were obtained in the first place. I found it particularly simple to use that `h(t)` *satisfies* `sum_i=0^k c_i d^i/dt^i h(t)=delta(t)`*, so the convolution property* `(h*(d/dt x))=((d/dt h)*x)(t)` *yields* `(h*sum_i=0^k c_i d^i/dt^i x)(t)=x(t)`*. Thus,* `A^H x(t)+B^H u(t)+B^H_2 d/dt u(t)+...` *should generate* `sum_i=0^k c_i d^i/dt^i x(t)`*. One can obtain the derivatives of* `x(t)` *by writing down* `d^2/dt^2 x=A^2 x+...`*,* `d^3/dt^3 x=A^3 x+....` *and deducing the general structure. Did the authors also change to the time domain for the derivation?*

1.12. We have added the citation: Voelker & Eliasmith (2017), *Methods for applying the Neural Engineering Framework to neuromorphic hardware*, arXiv. In that report, and in our recent ISCAS paper (Voelker et al., 2017), we include a derivation that sounds analogous to the one that you have proposed. But this is not needed for the proof in this manuscript, which relies on the algebraic identity after equation 26. While both perspectives are useful, the linear transfer function method is more flexible for this manuscript (e.g., for section 4.3). The method from this manuscript is also more amenable to analysis of the overall (network-level) transfer function resulting from swapping one synapse for another (e.g., for appendix A.4). As a result, and for brevity, we have omitted such alternative derivations.

14. p. 34 bottom paragraph: It may be helpful for readers who are not experts in the NEF scheme to shortly highlight the precise architecture and couplings used for the implementation.

1.13. The architecture here is the same from section 2 onward. We have added some citations to section 5.1, as well as some additional exposition to section 2.3 (see reply 2.4) to highlight the architecture more directly.

15. p. 46 bottom paragraph: Which models use the unrealistic assumptions mentioned?

1.14. None of the models mentioned in this paper or its citations use the unrealistic assumptions mentioned here. However, from personal communication, we have found it important to keep in mind that our weights do not need to be changed on-the-fly in order to achieve this kind of multiplicative control (a common misunderstanding with NEF models).

Reviewer #2

In response to the invitation to review the paper 'Improving Spiking Dynamical Networks: Accurate Delays, Higher Order Synapses, and Time Cells':

The paper addresses a very important topic in neuroscience, i.e. the coding and decoding of information based on nonlinear and time-delayed interaction in recurrent systems. It provides an extended framework that is based on the previous work of Chris Eliasmith. The extension incorporates non-exponential dynamics of synaptic responses and delay coupling described by time-delayed axonal spikes.

Overall, the paper is well written. The introduction is good and concise. However, work in the domain on delay-coupled networks and their use for computation is not sufficiently reflected. Especially work in the domain of delay-coupled reservoir computing that is very closely related to this paper is not covered (for details, see recommendationlater . The main part of the paper is again well written. At the same time, several sections are rather technical and use extensive technical slang. Furthermore, several rather strong statements are not sufficiently supported by citations. Eventhough I have a strong mathematical background, I still had to google several expressions and proofs. It seems to me as if the paper could benefit from clarifying several parts concerning required background knowledge.

Concerning the scientific results, the paper is sound and precise. The claims are supported by several numerical simulations that nicely highlight the main points of this paper. So, overall, this paper is very good, but requires reworking several sections to improve readability, reduce technical slang, and include the so far ignored scientific work that appears to be very closely related to this submitted paper.

We appreciate the reviewer's overall positive take on the results, and have attempted to address the important considerations regarding clarity and past work in our responses to each of the details provided below.

Details. a) In the introduction, the work related to the gamma model should be discussed. The gamma model is a rather complete framework that highlights the flexibility of functional mapping in terms of input/output mapping based on delayed responses, implemented by delayed filter kernels. I propose including an in depth discussion and highlighting how the new approach differs from the gamma model. ('The gamma model-A new neural model for temporal processing' by Bert de Vries and Jose C. Principe, University of Florida, USA, ScienceDirect: http://www.sciencedirect.com/science/article/pii/S0893608005800358).

2.1. We agree that the gamma model is closely related and have significantly extended the introduction to incorporate a discussion highlighting similarities and differences. We begin by highlighting some of the important differences. First, spiking neurons are not considered in the paper by Bert de Vries and Jose C. Principe (which we will refer to as VP) or in the literature we found in section 2 of VP. Second, their focus is on using backpropagation to train the weights to minimize some supervised error signal, while our methods are focused on analytically mapping prescribed model dynamics onto arbitrary linear synapses. How backpropagation learning may or may not be able to learn such functions in networks of spiking neurons (which have discontinuous gradients) is a discussion beyond our scope, but remains a field of active research (e.g., "Gradient Descent for Spiking Neural Networks" by Huh & Sejnowski (2017) and "Deep learning in spiking LIF neurons" by Hunsberger & Eliasmith (2017)). We have added citations to VP and Sejnowski, among others, in the introduction to highlight these differences. Third, the use of the term "delay kernel" by VP is referring to the fact that the output of a temporal convolution at time t depends on the input at time $t - s$ (e.g., equation 8), as specified by some kernel of weights. In particular, VP refer to equation 29 as a "delay structure", even though it does not implement a pure delay, but rather it is a set of differential equations implementing their gamma kernel. In the signal processing literature this is a particular kind of lowpass filter (more on this below). This is very different from our use of the word 'delay', which we use consistently to refer to a system that acts as a pure delay line on its input.

For example, the gamma kernel $g_k^{\mu}(t)$ (see equation 19 and Figure 3 from VP) is the transfer function $H_k(s) = \frac{1}{(\mu^{-1}s+1)^k}$. This can be seen by manipulating equation 22.[1] When $k = 1$, this is the canonical first-order lowpass filter. When $k = 2$, this is the alpha synapse discussed in our extensions. Our general case applies to any integer $k > 0$. We now mention this in section 4.4. Again, VP call these "delay kernels" because, qualitatively, they convey delayed information via the peak of their impulse responses – but this comes at the cost of applying a lowpass filter k times (and so this is quantitatively quite unlike a pure delay). The overall gamma memory model uses a weighted combination of these kernels in some nonlinear feedback configuration (Figure 6 and equations 33, 34 from VP), which they train by backpropagation.

To summarize, VP use gamma kernels in place of synapse models, no spikes, and backpropagation to determine the recurrent weights from training data. Since these methods require significant work to translate into the spiking domain, there

[1] Also proof: `http://www.wolframalpha.com/input/?i=inverse+laplace+1%2F(%5Cmu%5E%7B-1%7D*s+%2B+1)%5Ek` is identical to equation 19.

is no clear way for us to use their model. More importantly, there is no apparent advantage to attempting this if the desired dynamics are already expressed by some closed-form dynamical model (as in our manuscript). Their methods could help if we did not have a dynamical model to begin with, but then we would lose our ability to say what the trained variables are actually representing. In contrast to the gamma model, our methods show how to precisely compensate for all of the lowpass filtering that occurs, and to produce a well-understood low-dimensional delay representation in a population of spiking neurons (section 3.2).

As noted earlier, these considerations are now included extensively in the introduction to help distinguish our contributions. As well, relevant portions have been added to sections 3 and 4.4.

b) I feel that closely related work from the domain of delay-coupled reservoir computing is not mentioned or cited. Delay-coupled reservoir computing has already introduced the idea of using delay coupling for complex encoding and subsequent linear decoding. There, it was also shown that the delay-coupled system is isomorphic to a recurrent network with constrained connectivity. Both seem to be closely related to Laplace transform equation 10 used in this paper. Relevant sections appear to be the introduction, section 3, and especially section 3.2.

2.2. See reply 1.3 in regards to Reservoir Computing in general. In summary, the manuscript now clarifies that compared to RC our methods are efficient with respect to the prescribed dynamics, are accurate despite using spiking neurons, and are well-understood when coupled using higher-order linear synapses.

Thank you for bringing the work on delay-coupled reservoir computing to our attention. We have now reviewed and cited *Information processing using a single dynamical node as complex system* by Fischer et al. (2011) in our manuscript. We have noted in section 3 that the delay network from our manuscript provides a biological basis for delay-coupled reservoir computing, by providing a mechanism that accurately implements the delay line required by that architecture.

c) I would like to stress that I am absolutely convinced that many parts in this paper are new and highly relevant. Especially the approximation of equation 10 for rendering the system finite dimensional is super relevant and exciting. At the same time, I would like to see that the paper links better with previously published research.

2.3. Thank you. We have tried to better frame this as a central contribution of our work, and at the same time provide a more comprehensive overview of related literature (also see reply 1.1).

d) While principles 1 and 2 are ok in terms of language and slang use, section 3 is much less clear. The paragraph in principle 3 starting with 'Principle 3' on page 11 is not clear enough. Figure 2 remains unclear, as well. Again, on page 12 the paragraph starting with 'Principle 3' is very technical. Especially the last part in terms of the generalization remains unclear. Given the importance of this part for the paper, I would recommend adding citations that support the claims.

2.4. Thank you for the suggestions on improving the clarity of this discussion. We have updated the manuscript significantly to alleviate this concern. Specifically, we have made section 2.3, in particular the connections to Figures 1 and 2, more clear in terms of describing the architecture and what we are trying to accomplish with Principle 3. We have included citations to the 2003 NEF book where more details may be desired. We have added a citation to a classic control theory text to support all of the general statements about linear systems. We have referenced equation 3 and added a citation to Voelker & Eliasmith (2016) to support further generalization claims. Reply 1.11 also contains related information that may be helpful in clarifying the presentation of Principle 3.

e) In section 3, delays are introduced. The authors model the delays with delayed axonal spiking. So the delay is modeled on the output side. Often, the delay is written in terms of delayed input. The authors may want to make that explicit and discuss the change in this reference.

> **2.5. To clarify, by linearity/commutativity of filtering/convolution, it is equivalent to move the axonal spike delay into the synapse. We have added a note to the beginning of section 4.3 to make this clear.**

f) In 3.1, the first part is again very technical. Moreover, the implication and limitations resulting from the low-frequency signal assumption should be discussed.

> **2.6. We have added some notes to the beginning of section 3.1 to make the relationship to section 2.3 and what we are trying to accomplish more clear. The implication and limitations resulting from the low-frequency assumption are highlighted in Figure 4-Left, and discussed in the corresponding body of the text and in the discussion on scaling. Technically, this means that higher frequencies (relative to theta) are passed through with uniform power (and some phase-shift), and thus need to be lowpass filtered to be removed from the output signal.**

g) In general, I had a hard time connecting recurrence within a recurrent network with equation 10. I feel that I might not be the only one, so that a dedicated section may help future readers to understand that link from the author's perspective, especially in respect to different perspectives in relation to my point e).

> **2.7. Equation 10 is a mathematical formulation of the transfer function that we would like the network to implement at the system level. This connects back to equation 7 and Principle 3, which tells us how to get a recurrent network to implement $F(s)$ (via equation 11). This is now described more carefully in section 3.1 (also see replies 2.4 and 2.6).**

We would like to reiterate our appreciation for the discussion and changes stimulated by the reviewers. If there are any additional questions or issues, please do not hesitate to contact us.

Sincerely,
Aaron R. Voelker and Chris Eliasmith