A Geometric Interpretation of Feedback Alignment

Andreas Stöckel (astoecke@uwaterloo.ca)
Terrence C. Stewart (tcstewar@uwaterloo.ca)
Chris Eliasmith (celiasmith@uwaterloo.ca)
Centre for Theoretical Neuroscience, University of Waterloo, 200 University Avenue West
Waterloo, ON N2L 3G1 Canada

Abstract

Feedback alignment has been proposed as a biologically plausible alternative to error backpropagation in multi-layer perceptrons. However, feedback alignment currently has not been demonstrated to scale beyond relatively shallow network topologies, or to solve cognitively interesting tasks such as high-resolution image classification. In this paper, we provide an overview of feedback alignment and review suggested mappings of feedback alignment onto biological neural networks. We then discuss a novel geometric interpretation of the feedback alignment algorithm that can be used to analyze its limitations. Finally, we discuss a series of experiments in which we compare the performance of backpropagation and feedback alignment. We hope that these insights can be used to systematically improve feedback alignment under biological constraints, which may allow us to build better models of learning in cognitive systems.

Keywords: feedback alignment; biologically plausible backpropagation; weight normalization; Neural Engineering Framework

Introduction

Error backpropagation has been employed with great success in the field of machine learning as a supervised method for training neural networks. Networks trained with backpropagation achieve near or even above human-level performance for a variety of classification tasks (LeCun, Bengio, & Hinton, 2015).

Unfortunately, it is difficult to see how the backpropagation algorithm could be implemented in biological neurons. The main reason that is often cited for concern is that it makes use of perfect bidirectional weight transport (e.g., Stork, 1989). In other words, the error signal that is used for one layer of the network needs to be multiplied by the connection weights of all the later layers in the network; or, mathematically, $\vec{\delta}^{\nu} = W^T \vec{\delta}^{\nu+1}$, where W is the synaptic weight matrix for layer ν.

Lillicrap, Cownden, Tweed, and Akerman (2016) discovered that the problematic W^T matrix can be replaced with a *random* matrix B without affecting the quality of the learned function too much. In practice, learning with random weights is not quite as good as backpropagation, yet under certain conditions the added randomness can act as a regularizer, resulting in even better performance (Mesnard & Richards, 2018). This modified algorithm has been called "Feedback Alignment", based on the observation that the learning process tends to drive the learned W matrix to be similar to the transpose of the randomly generated learning matrix B.

Because this new algorithm does not have the strict mathematical basis that is present for backpropagation, it difficult to understand why exactly feedback alignment works. In this paper, we present a novel interpretation of feedback alignment that leads to both a better understanding of how it works and also to new variants of feedback alignment that are more suited to biological implementation.

Feedback Alignment and Pyramidal Neurons

Currently, there is an existing mapping of feedback alignment onto neocortical pyramidal neurons proposed by Guerguiev, Lillicrap, and Richards (2017). In particular, they map the normal feed-forward neural network connections onto the basal dendrites of a pyramidal neuron, and the randomly generated feedback connections to the apical dendrite. With this structure, the output firing pattern of the pyramidal neuron is almost entirely driven by the feed-forward connections, as would be desired. After all, we do not want the error feedback to *directly* modify the neuron's behaviour – rather, we would like it to adjust the synaptic weights, which in turn will modify the neuron's behaviour. However, this error feedback does affect the *bursting* behaviour of the neuron. That is, a large input to the apical dendrite (i.e., a large error feedback) renders the neuron more prone to generating bursts of spikes. So a basal input that would cause the neuron to spike once if there is no apical input might lead to a burst of multiple spikes if there is apical input.

Guerguiev et al. (2017) point out that this is exactly the behaviour that would be needed for feedback alignment to be implemented biologically. Given this basis, we believe it is important to further investigate the feedback alignment algorithm in order to understand what it is doing, how it differs from standard backpropagation, and how all of the steps in the algorithm can be mapped onto biological processes.

It should also be noted that feedback alignment does not perform well for the extremely deep networks that are seen in modern machine learning algorithms. These networks are often dozens of layers deep, and in these cases feedback alignment learns much more slowly, if at all, compared to backpropagation. However, we note that biological systems tend not to have such extremely deep networks, and so that restriction may not be important for realistic neural systems.

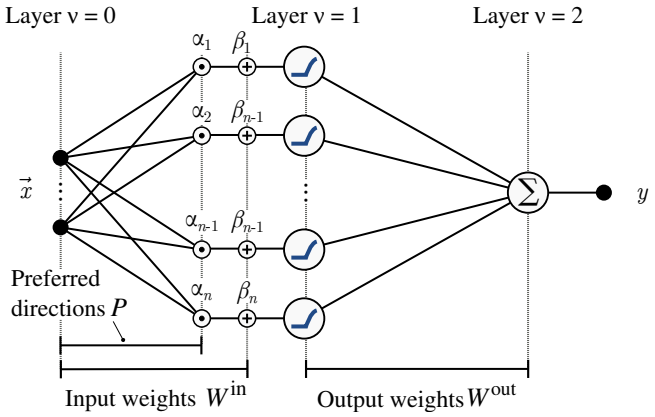

Figure 1: Our single hidden-layer network setup.

A New Geometric Interpretation of Feedback Alignment

We start by considering a single-hidden-layer network. For simplicity, we only include a neuron non-linearity in the hidden layer. This does not reduce the computational power of the network (Hornik, 1991), and the same conclusions will also hold for more traditional networks with non-linearities at each layer. The resulting system is shown in Figure 1.

We can now interpret the activity of the hidden-layer neurons in a similar manner as in cognitive neuroscience. In particular, each neuron is assigned a *tuning curve*: it will respond to inputs $\vec{x}$ by firing at pre-defined rates a_i

$$a_i(\vec{x}) = f(\langle \vec{w}_i^{\text{in}}, \vec{x} \rangle + \beta_i),$$

where $\langle \cdot, \cdot \rangle$ denotes the dot-product, $\vec{w}^{\text{in}}$ is a vector of synaptic weights, and β_i is the bias.

As proposed in the Neural Engineering Framework (NEF), the synaptic weight vector $\vec{w}^{\text{in}}$ can be decomposed into two parts (Eliasmith & Anderson, 2003): a scalar gain α_i and a unit-length *preferred direction* or *encoding* vector. That is, there is a vector $\vec{p}_i$ for each neuron such that inputs $\vec{x}$ that are similar to that vector (in terms of the dot-product) will produce higher firing rates. This sort of preferred direction vector has been found throughout the brain, most famously by Georgopoulos, Schwartz, and Kettner (1986), who demonstrated that neurons in motor areas respond preferentially to specific directions during movement execution. We show this for two neurons in Figure 2.

Importantly, the four terms f, $\vec{p}_i$, α_i, and β_i relate to different geometric aspects of the tuning curves. The neuron non-linearity f controls the one-dimensional shape of tuning curve (sigmoid, rectified linear, etc.), and $\vec{p}_i$ controls which direction the tuning curve is pointing in. The gain term α_i scales the tuning curve, making the transition sharper or more gradual. Finally, the bias term β_i shifts the tuning curve along the direction $\vec{p}_i$. As we shall see, separating the weight $\vec{w}^{\text{in}}$ into α and $\vec{p}_i$ will lead to a novel interpretation of the weight changes caused by feedback alignment.

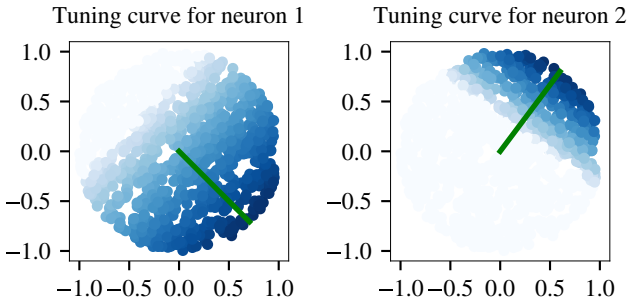

Figure 2: The tuning curves (firing rates) of two neurons for a random sampling of input $\vec{x}$ values inside the unit circle. The input weights for the neurons are $\vec{w}_1 = [1, -1]$ and $\vec{w}_2 = [3, 4]$. This gives preferred direction vectors of $\vec{p}_1 = [1/\sqrt{2}, -1/\sqrt{2}]$ and $\vec{p}_2 = [3/5, 4/5]$. The gains are $\alpha_1 = \sqrt{2}$ and $\alpha_2 = 5$, which corresponds to the second neuron having a much steeper tuning curve than the first neuron (i.e., it transitions from no firing (white) to full firing (dark blue) over a much shorter distance). The first neuron has a positive bias ($\beta_1 > 0$) and the second neuron has a negative bias ($\beta_2 < 0$), which has the effect of shifting the tuning curve along the direction $\vec{p}$.

Weight changes in the output layer $\vec{w}^{\text{out}}$

In both feedback alignment and standard backpropagation, the learning rule for the output layer can be expressed using the standard delta learning rule (Widrow & Hoff, 1960), which can be written in terms of an outer product

$$\vec{\delta}^{\text{out}} = (\vec{y} - \vec{t})^T, \qquad \Delta W^{\text{out}} = -\kappa \vec{a}^{\text{out}} \left(\vec{\delta}^{\text{out}}\right)^T, \qquad (1)$$

where κ is the learning rate and $\vec{a}$ are the activities of the output neurons given an input $\vec{x}$. The rule adjusts the weight matrix W^{out} to whatever values minimize the difference between actual output of the system $\vec{y}$ and the desired $\vec{t}$.

Assuming that all samples are known beforehand, and that the rest of the model is held fixed, we can directly solve for optimal W^{out} using least-squares minimization (Eliasmith & Anderson, 2003). This optimization approach has been shown to be robust across a wide range of neuron nonlinearities, including spiking neurons such as the standard leaky integrate-and-fire (LIF) neurons. We use this optimization in this paper, which corresponds to an assumption that the learning rate for the output weights is significantly higher than the learning rate for the input weights.

Weight changes in the input layer $\vec{w}^{\text{in}}$

In the standard backpropagation algorithm, the learning rule for the input layer takes into account the connection weights in the output layer, the derivative of the neuron non-linearity, and the input values. Generally speaking, the weight changes ΔW^{ν} in any layer ν except for the output layer are

$$\vec{\delta}^{\nu} = f'(\vec{x}^{\nu})(W^{\nu+1})^T \vec{\delta}^{\nu+1}, \quad \Delta W^{\nu} = -\kappa \vec{a}^{\nu-1} \left(\vec{\delta}^{\nu}\right)^T \qquad (2)$$

where $f'(\vec{x}^{\nu})$ is the derivative of the neuron non-linearity for its current input value $\vec{x}^{\nu}$. The presence of W^{out} in this equation makes a biological implementation of backpropagation

problematic, as there seems to be no good way for the synapses between the input layer and the hidden layers to have access to the connection weight strengths.

In direct feedback alignment (Lillicrap et al., 2016), the weight transport problem is eliminated by replacing $W^{\nu+1}$ in eq. (2) with a random matrix B^{ν}

$$\vec{\delta}^{\nu} = f'(\vec{x}^{\nu}) B^{\nu} \vec{\delta}^{\text{out}} \tag{3}$$

It is quite surprising that using randomly generated weights still results in a useful learning algorithm, and indeed it has been shown that feedback alignment often has almost the same performance as backpropagation. The purpose of this paper is to present a geometric interpretation of eq. (3) that helps to explain and understand this algorithm.

Visualizing feedback alignment

The feedback alignment learning rule presented in the previous section (eq. 3) consists of five terms: the learning rate κ, the activities of the previous layer $\vec{a}^{\nu-1}$, the derivative of the neuron activation f', the random feedback weights B, and the error from the output layer $\vec{\delta}^{\text{out}} = \vec{y} - \vec{t}$. For simplicity, we can fix each column in B to be normalized, so we do not need to keep track of an extra gain factor.

Taking these components in reverse order, we start by noting that $B(\vec{y} - \vec{t})$ can be interpreted as projecting the error onto a random direction, where each neuron has a different random direction in the error space that it is sensitive to. That is, just as each neuron i has a preferred direction vector $\vec{p}_i$ (based on the normalized ith row of the matrix W^{in}), each neuron also has a preferred *error* direction $\vec{e}_i$ (based on the normalized ith column of the B matrix).

As an example, consider the case where our output y is one-dimensional, but our network input $\vec{x}$ is two-dimensional. In this case, each hidden layer neuron will have a two-dimensional tuning curve based on its input (as in fig. 2), but will also have a learning input that is based on the one-dimensional error, mediated by that neuron's column in the B matrix. Correspondingly, some neurons will get a learning input that is larger if the projected error is positive (if $\langle \vec{e}_i, \vec{\delta}^{\text{out}} \rangle > 0$), and other neurons will get a learning input that is larger if the projected error is negative (if $\langle \vec{e}_i, \vec{\delta}^{\text{out}} \rangle < 0$). Importantly, this learning input does *not* affect the effective output of the neuron itself – rather, it will only used for the synaptic update rule.

The next component of the learning rule is the derivative of the neuron activation given the current input, f'. While differentiation is a difficult problem for a biological neuron, it can be approximated by a step function determining whether the neuron is currently firing or not (Eliasmith & Anderson, 2003, p. 295). That is, if the neuron is not firing for its current input, then increasing the input a small amount is not likely to cause the neuron to start firing, so $f' = 0$ when the neuron is not firing. If it is firing, then increasing the input is likely to cause the neuron to increase its firing rate, so we say $f' = 1$ when the neuron is firing. This approximation is exactly correct for the special case of a rectified linear neuron.

Returning to our geometric interpretation of feedback alignment, the fact that the learning rule says to multiply by f' means that the neuron will only be sensitive to its error input *if the neuron is already firing*. That is, the neuron is sensitive to errors in a particular direction (due to $\vec{e}_i$), but only if it is already firing for the input that caused that error. Again, this is consistent with Guerguiev et al. (2017).

Finally, the last two components of the learning rule are the activities of the previous layer $\vec{a}^{\nu-1}$ (or, in the case of a single hidden layer, the network input $\vec{x}$) and the learning rate κ. These are multiplied by the above components and added to the input weights $\vec{w}_i^{\text{in}}$. Adjusting these input weights adjusts the preferred direction vector $\vec{p}_i$ for that neuron.

This leads to our geometric interpretation of the feedback alignment learning rule: feedback alignment adjusts the preferred direction vectors of neurons such that they are more aligned with the inputs that lead to errors in the direction that this neuron is sensitive to.

As a point of comparison, traditional Hebbian learning can be interpreted in this way as a rule that adjusts the preferred direction vectors of neurons such that they are more aligned with the main principal component of the inputs. Correspondingly, feedback alignment can be seen as an error-modulated Hebbian rule, where the error modulation is done by having each neuron pick a (random) direction in the error space that it will be sensitive to, and only adjust its preferred direction towards the current input if the neuron is firing *and* the error is in its preferred error direction.

The effects of repeating this process are shown in Figure 3. Here, the network is being trained to compute the product of its inputs ($y = x_1 x_2$). Furthermore, we show in Figure 4 that as this process continues, the preferred direction vectors $\vec{p}$ change from being randomly distributed to being aligned along the diagonals. As can be seen in the centre column of Figure 3, the network makes the largest errors on those diagonals. This is our interpretation as to why Feedback Alignment works: it moves the neuron's preferred direction vectors towards the parts of the input space where the largest errors are being made.

Weight normalization

Decomposing neural tuning curves into gains α_i, biases β_i and normalized preferred direction vectors $\vec{p}_i$ as in the NEF has two potential benefits. First, fixing the gain and bias allows modellers to distribute the tuning curves across the representational space while taking neurobiological constraints such as maximum firing rates into account. Second, keeping the preferred direction vectors normalized reduces the degrees of freedom in the optimization problem, potentially resulting in faster convergence.

Thus, while normalizing the weights $\|\vec{w}_i^{\nu}\|_2 = \alpha_i$ can be beneficial, note that the backpropagation and feedback alignment update rules discussed above do not account for normalization. While the weight vectors $\vec{w}_i^{\text{in}}$ can be renormalized to length

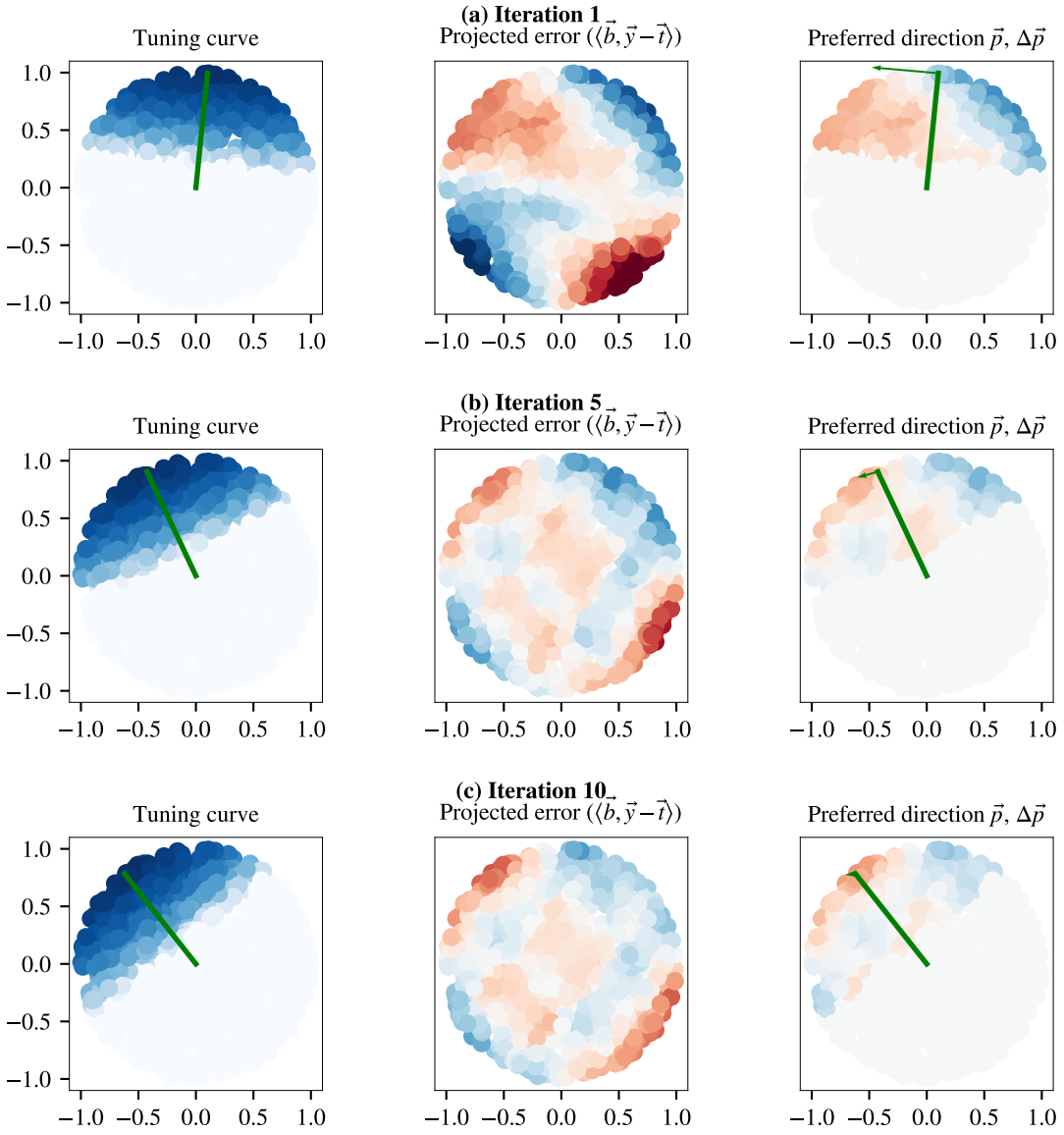

Figure 3: Geometric interpretation of feedback alignment over three iterations. The left column depicts the tuning curve of a single neuron and its preferred direction $\vec{p}$. The center column shows the error δ^{out} projected onto the preferred error direction $\vec{b}$. The right column depicts the projected error multiplied with the derivative and the resulting preferred direction update $\Delta\vec{p}$.

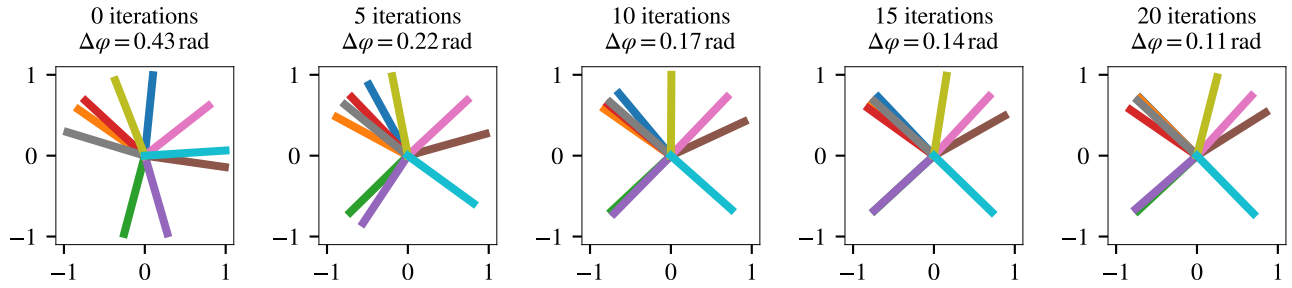

Figure 4: The movement of the preferred direction vectors as the feedback alignment algorithm progresses. $\Delta\varphi$ corresponds to the average distance of the encoding vectors to one of the diagonals in radians.

α_i after each update step, forcing normalization inevitably deflects the true direction of the gradient. Hence, at least from a theoretical perspective, it is important to take weight normalization into account when computing the gradient.

One way to compute such an augmented gradient has been previously outlined by Salimans and Kingma (2016). To summarize, we can reparameterize the weights connecting from layer ν to the ith neuron in layer $\nu+1$ as

$$\vec{w}_i^{\nu} = \gamma_i^{\nu} \vec{p}_i^{\nu}, \qquad \text{where } \gamma_i^{\nu} = \frac{\alpha_i^{\nu}}{\|\vec{p}_i^{\nu}\|_2}. \tag{4}$$

We can now derive the backpropagation equations. The augmented backpropagated error δ_i^{ν} is given as

$$\delta_i^{\nu} = \gamma_i^{\nu} f'(\vec{x}_i^{\nu})(W^{\nu+1})_i^T \vec{\delta}^{\nu+1}. \tag{5}$$

The only difference to standard backpropagation is the added normalization factor γ_i^{ν}, thus a weight-normalized version of feedback alignment can be obtained by replacing $W^{\nu+1}$ with B^{ν}, as before. The actual update rule for the connections weights is given as

$$\Delta\vec{w}_i^{\nu} = -\kappa\gamma_i^{\nu}\left(a_i^{\nu-1}\vec{\delta}^{\nu+1} - \frac{(\vec{w}_i^{\nu})^T\vec{a}^{\nu-1}\delta_i^{\nu+1}}{\|\vec{w}_i^{\nu}\|_2^2}\right). \tag{6}$$

The above equation implies that each neuron has information about its local synaptic input weight magnitude $\|\vec{w}_i^{\nu}\|_2^2$. From a biological standpoint, this is not unreasonable, since empirical evidence suggests the presence of synaptic homoeostasis within individual neurons (Keck et al., 2017).

Experiments and Results

In the previous section we discussed feedback alignment in the context of networks with normalized weight vectors and fixed gains and biases. In this section we seek to empirically validate the predictions made above.

In our first experiment, we analyze the effect of weight normalization and the proposed augmented gradient when training a network to compute multiplication. Since we know that the optimal preferred direction vectors coincide with the diagonals in this particular task (Gosmann, 2015), we can measure how successful the optimizer is in finding this solution.

The network architecture is as depicted in fig. 1, with 20 neurons in the hidden layer. We train W^{in} over 100 epochs with 1000 training samples, both with feedback alignment and backpropagation. After each epoch, W^{out} is computed in closed form via least-squares minimization. To test the hypothesis that weight normalization can lead to faster convergence, we run the experiment with forced normalization, no normalization at all, as well as the augmented gradient given in eqs. (4) to (6). We measure the training error for each epoch, the average distance $\Delta\varphi$ between the preferred direction vectors, and the average length of $\vec{p}_i$.

The median results over 200 trials are shown in fig. 5. In general, backpropagation achieves a slightly smaller RMSE than feedback alignment. The system converges more quickly to a smaller error if synaptic weights are either normalized or the augmented gradient update is applied. In general, taking the augmented gradient into account, the weights stay approximately normalized, although this is not strictly enforced. As a result, training with artificially normalized weights and the augmented gradient results in exactly the same error. Independent of normalization, feedback alignment moves the encoding vectors more quickly towards the diagonals than back propagation.

In our second experiment we train a network with 200 hidden neurons on the MNIST dataset consisting of 60 000 handwritten digits from zero to nine as 28×28 pixel images. We train the network in batches of 5000 samples over 100 epochs. As before, feedback alignment and backpropagation are only used to train the input weights. The output weights are solved using least squares optimization.

Our results over 16 trials depicted in Figure 6 suggest that the presence or absence of weight normalization does not play an essential role in this higher-dimensional setup. As before, backpropagation achieves a smaller training and test error than feedback alignment.

Discussion

We presented a new geometric interpretation of feedback alignment in the context of neurobiologically plausible networks with neural tuning curves determined by a gain, bias, and preferred direction vector. In the context of computing multiplication, we have demonstrated that feedback alignment drives the preferred direction vectors more quickly towards the theoretically optimal configuration.

Furthermore, we discussed an augmented feedback alignment and backpropagation update rule that takes weight normalization into account. Somewhat surprisingly, this update rule itself ensures that the weight vectors are normalized. Correspondingly, it would be interesting to see whether a simplified version of eq. (6) under the assumption of $\|\vec{w}_i\|_2 = \alpha_i$ could be a feasible model for synaptic homeostasis.

In the future, we intend to use our interpretation to gain a better understanding of such failure-cases of feedback alignment. One simple failure case has been identified by Hunsberger (2018), who points out an XOR-like "three-banded" task, that can be trained with back-propagation, but not with feedback alignment; yet it remains unclear why exactly this is the case.

References

Eliasmith, C., & Anderson, C. H. (2003). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press.

Georgopoulos, A. P., Schwartz, A. B., & Kettner, R. E. (1986). Neuronal Population Coding of Movement Direction. *Science*, *233*(4771), 1416-1419.

Gosmann, J. (2015). *Precise multiplications with the NEF* (Tech. Rep.). Waterloo, ON: Centre for Theoretical Neuroscience.

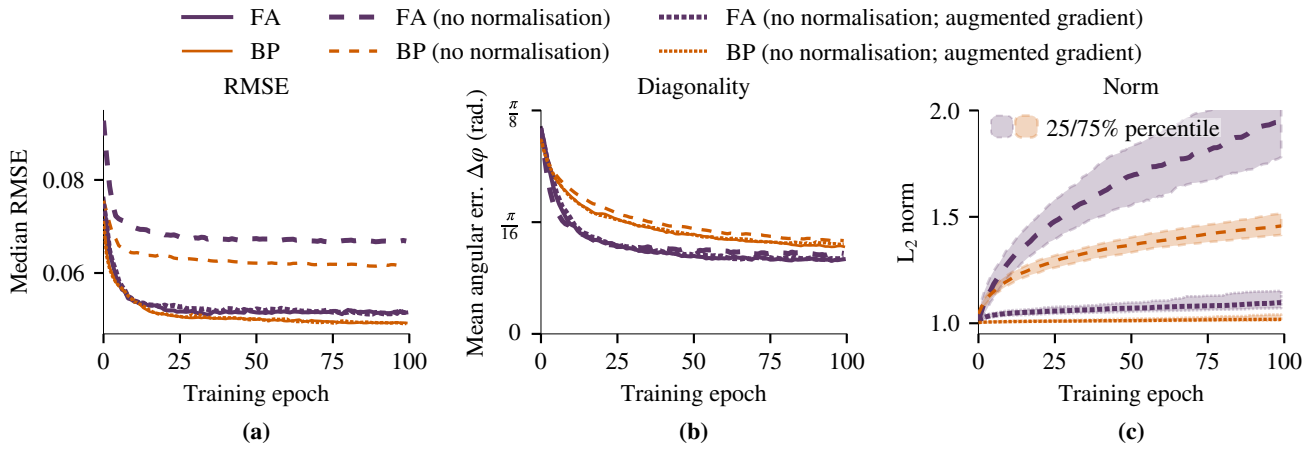

Figure 5: Effects of weight normalization. Comparison between backpropagation and feedback alignment when training the preferred direction vectors to compute multiplication with and without normalization. All quantities correspond to the median over 200 trials with 20 hidden LIF neurons. **(a)** Median RMSE. **(b)** Median average distance of the preferred vectors from the diagonals. **(c)** Median average norm of the preferred vectors with and without the augmented gradient update.

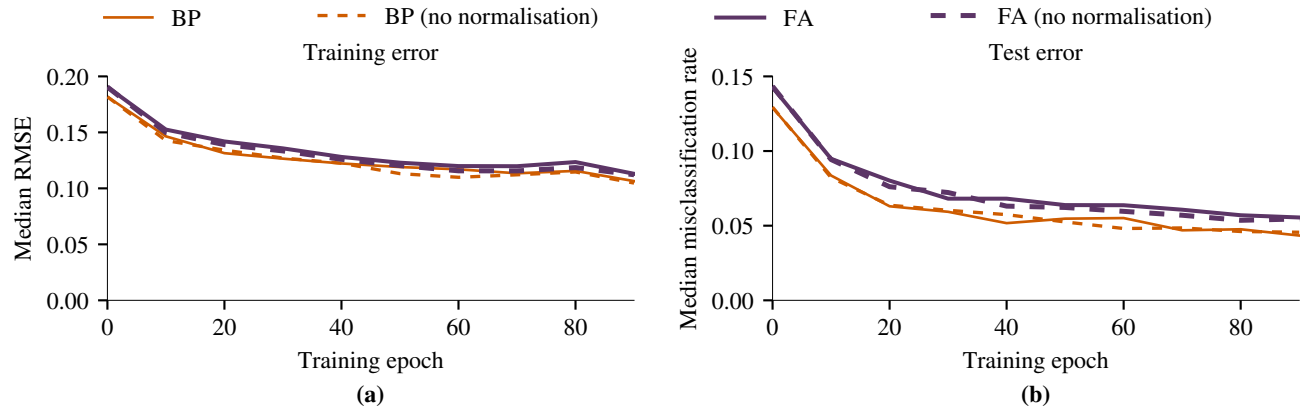

Figure 6: MNIST trained with backpropagation and feedback alignment. Reported quantities are the median over 16 trials. **(a)** RMSE between the one-hot coded output $\vec{y}$ and the target $\vec{t}$. **(b)** Classification error on the test data with 10000 samples.

Guerguiev, J., Lillicrap, T. P., & Richards, B. A. (2017). Towards deep learning with segregated dendrites. *ELife*, *6*, e22901.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, *4*(2), 251-257.

Hunsberger, E. (2018). *Spiking deep neural networks: Engineered and biological approaches to object recognition*. Phd thesis, University of Waterloo.

Keck, T., Toyoizumi, T., Chen, L., Doiron, B., Feldman, D. E., Fox, K., ... van Rossum, M. C. (2017). Integrating Hebbian and homeostatic plasticity: The current state of the field and future research directions. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *372*(1715), 20160158.

LeCun, Y., Bengio, Y., & Hinton, G. (2015, May). Deep learning. *Nature*, *521*, 436.

Lillicrap, T. P., Cownden, D., Tweed, D. B., & Akerman, C. J. (2016, November). Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, *7*, 13276.

Mesnard, T., & Richards, B. (2018). Activation alignment: exploring the use of approximate activity gradients in multilayer networks. In *2018 Conference on Cognitive Computational Neuroscience.* Philadelphia, Pennsylvania.

Salimans, T., & Kingma, D. P. (2016). Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 29* (p. 901-909). Curran Associates, Inc.

Stork. (1989). Is backpropagation biologically plausible? In *International 1989 Joint Conference on Neural Networks* (p. 241-246 vol.2).

Widrow, B., & Hoff, M. E. (1960). *Adaptive switching circuits* (Tech. Rep.). Stanford University, CA, Stanford Electronics Labs.