DIRECT STATE FEEDBACK: FAST RNN TRAINING WITH STATIONARY BACKWARD DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Backpropagation through time (BPTT) trains recurrent neural networks by transporting errors through a sequence of time-varying state Jacobians, making temporal gradient propagation inherently sequential. Inspired by feedback alignment, we introduce Direct State Feedback (DSF), an alternative, model-agnostic training method for recurrent networks. DSF relies on a temporal feedback rule that replaces the recurrent Jacobians used during the backward pass with a fixed diagonal approximation, while preserving the original nonlinear forward dynamics. This turns temporal error transport into a stationary linear recurrence that can be efficiently evaluated with parallel scans. We further combine fixed feedback with exact gradients over short horizons, yielding T-DSF, which continuously interpolates between truncated BPTT and full temporal credit assignment. Despite its aggressive backward approximation, DSF retains much of the performance of BPTT across language modeling and memory tasks, while substantially outperforming purely local or truncated updates and reducing training cost at scales up to one billion parameters. Gradient diagnostics further show that DSF updates remain positively aligned with BPTT despite increasing approximation error at long horizons. These results suggest that recurrent networks can be trained with simple, stationary feedback dynamics in place of exact Jacobian transport, extending the feedback-alignment principle from depth to time.

1 INTRODUCTION

Recurrent neural networks (RNNs) provide a natural computational model for sequential data: they process inputs online, maintain a persistent hidden state, and can operate on sequences of arbitrary length (Hopfield, 1982; Jordan, 1997; Elman, 1990). Gated architectures such as long short-term memory networks (LSTMs) and gated recurrent units (GRUs) improve their ability to learn long-range dependencies and have historically been successful in applications ranging from handwriting recognition to machine translation (Hochreiter & Schmidhuber, 1997; Cho et al., 2014; Graves & Schmidhuber, 2008; Bahdanau et al., 2014). More recently, renewed interest in recurrent sequence models has highlighted their potential for efficient online inference and long-context processing (Orvieto et al., 2023; Beck et al., 2024).

Despite these advantages, training RNNs at scale remains challenging. Standard training relies on backpropagation through time (BPTT), which propagates error signals backward through the sequence by successively applying the state Jacobians of the recurrent transition. Since these Jacobians depend on the states and inputs encountered during the forward pass, temporal error transport is inherently sequential, thus preventing efficient, parallel implementations. Truncated BPTT, a popular approximation, reduces this sequential depth by restricting exact credit assignment to a finite horizon, but discards dependencies across segment boundaries (Williams & Peng, 1990). Parallel formulations of exact BPTT are also possible, but require the composition of time-dependent Jacobians and are prohibitively expensive for large hidden dimensions (Danieli et al., 2023; Lim et al., 2024). In this work, we therefore ask: *can the temporal gradient computed by BPTT be approximated in a form that is inexpensive and parallelizable, while retaining most of its usefulness for training?*

Our approach is motivated by Feedback Alignment (Lillicrap et al., 2016), which shows that useful parameter updates can be obtained without exactly reproducing the backward computations of

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

backpropagation. Direct Feedback Alignment (DFA), in particular, replaces layer-wise error propagation with fixed feedback projections, removing sequential dependencies across network depth (Nøkland, 2016; Launay et al., 2020). We apply the same principle across time. In BPTT, errors are propagated through a sequence of time-varying state Jacobians. We replace these Jacobians, during the backward pass only, with a fixed diagonal feedback operator. The resulting method, *Direct State Feedback* (DSF), leaves the nonlinear forward recurrence unchanged and approximates only the transport of error signals between recurrent states. Its backward recurrence is linear and time-invariant, allowing it to be evaluated with a parallel prefix scan (Hillis & Steele, 1986; Blelloch, 1990).

We additionally show DSF can be seamlessly combined with truncated BPTT, using exact Jacobians within short segments and stationary feedback for propagating longer-range gradients. This hybrid, which we call T-DSF, provides a continuous trade-off between the fidelity of temporal gradients and the sequential cost of computing them.

Our main contributions are as follows:

1. We introduce Direct State Feedback (DSF), extending Feedback Alignment to temporal credit assignment as an efficient approximation to BPTT for nonlinear RNNs.
2. We formulate DSF using a fixed diagonal feedback operator, yielding a linear time-invariant backward recurrence that can be evaluated with $\mathcal{O}(\log T)$ sequential depth using a parallel prefix scan.
3. We introduce T-DSF, which combines exact BPTT within short segments with DSF across segment boundaries.
4. We evaluate DSF and T-DSF on language modeling and controlled memory tasks, across recurrent architectures, sequence lengths, hidden dimensions, and model scales. DSF preserves much of the training quality of full BPTT while improving over fully truncated updates, and T-DSF improves over truncated BPTT at matched truncation horizons.

2 Temporal Credit Assignment and Approximate Feedback

2.1 Backpropagation Through Time (BPTT)

Let us consider a recurrent neural network defined by the recurrence relation:

$$\mathbf{h}_t = f_{\boldsymbol{\theta}}(\mathbf{h}_{t-1}, \mathbf{x}_t), \quad \mathbf{y}_t = g_{\boldsymbol{\phi}}(\mathbf{h}_t). \tag{1}$$

In this equation, $\mathbf{h}_t \in \mathbb{R}^d$ denotes the hidden state vector at time step t, which is produced by the recurrent cell $f_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$. Given the input sequence of observations $(\mathbf{x}_t)_{t=1}^T$, the recurrent network acts like a transducer and generates the output $\mathbf{y}_t$ using $g_{\boldsymbol{\phi}}$ at each time step. This prediction function maps hidden states to outputs. In a supervised setting, this model can be trained to minimize the loss function $\mathcal{L}_t = l(\mathbf{y}_t, \mathbf{y}_t^*)$ at every time step t. The total loss over the sequence is defined by $\mathcal{L} = \sum_{t=1}^T \mathcal{L}_t$.

In standard RNN training, gradient descent updates parameters by computing $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}$ as follows:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial \mathbf{h}_t} \frac{\partial f_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}(\mathbf{h}_{t-1}, \mathbf{x}_t). \tag{2}$$

In particular, the gradient $\mathbf{g}_t := \frac{\partial \mathcal{L}}{\partial \mathbf{h}_t}$ of the hidden state $\mathbf{h}_t$ involves recursively unfolding the recurrent state through time, as each hidden state depends on previous hidden states. Denoting the local errors $\mathbf{e}_t := \frac{\partial \mathcal{L}_t}{\partial \mathbf{h}_t}$ and the cells Jacobians $\mathbf{A}_t := \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}}$, we can express $\mathbf{g}_t$ through a simple linear recurrence:

$$\mathbf{g}_t = \mathbf{e}_t + \mathbf{g}_{t+1} \cdot \mathbf{A}_{t+1}. \tag{3}$$

This equation describes a linear dynamical system that evolves backwards with time, where the final gradient $\mathbf{g}_T$ plays the role of an initial state. It is usually solved by sequentially computing $\mathbf{g}_T$, then $\mathbf{g}_{T-1}$, and so on until $\mathbf{g}_1$. This sequential process prevents efficient parallel implementations on modern GPUs. We illustrate how BPTT operates in Figure 1.b.

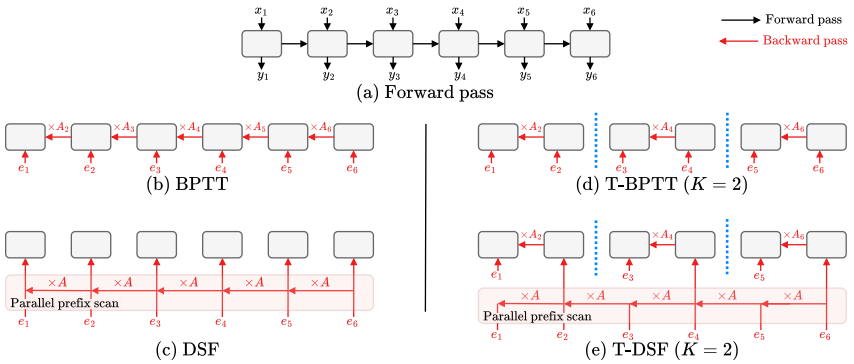


Figure 1: Illustration of an RNN and how the different algorithms operate. (a) The forward pass is the same for all methods. (b) BPTT backpropagates all errors sequentially through the RNN. (c) Our proposed DSF approximates the Jacobians $\mathbf{A}_t$ by a fixed matrix $\mathbf{A}$, allowing the use of fast prefix scan algorithms. (d) By splitting the sequence into segments, Truncated BPTT accelerates BPTT at the cost of accuracy. (e) DSF can be combined with T-BPTT to significantly improve training.

2.2 Truncated BPTT (T-BPTT)

The most popular way to accelerate BPTT is to split the sequences into multiple segments of length K during the backward pass (the forward pass remains untouched). BPTT is then run in parallel for each segment, thus improving computational efficiency. The segment size controls the accuracy/speed tradeoff: smaller segments are more computationally efficient, but prevent the model from learning long-range dependencies. The process is represented in Figure 1.d for $K = 2$.

The limiting case $K = 1$, which we refer to as Fully Truncated BPTT (FT-BPTT), entirely removes gradient propagation between recurrent states. This extreme approximation collapses both computational cost and sequential dependence to $\mathcal{O}(1)$ but discards temporal structure, typically degrading learning. FT-BPTT provides an important baseline for determining whether an approximate method transports useful information across time, rather than merely relying on local error signals.

2.3 Feedback Alignment

Feedback Alignment (FA) (Lillicrap et al., 2016) was introduced for deep feedforward networks, rather than recurrent models. Consider a network with hidden representations $\mathbf{h}^\ell = \phi^\ell(\mathbf{h}^{\ell-1}\mathbf{W}^\ell)$, and let $\boldsymbol{\delta}^\ell := \frac{\partial \mathcal{L}}{\partial \mathbf{h}^\ell}$ denote the error at layer ℓ. Standard backpropagation recursively transports errors through the transposes of the forward weight matrices:

$$\boldsymbol{\delta}^\ell = \left(\boldsymbol{\delta}^{\ell+1} \odot \phi^{\ell\prime}\right) \mathbf{W}^{\ell+1\top}. \tag{4}$$

FA replaces these transposed weights with fixed random feedback matrices $\mathbf{B}^{\ell+1}$:

$$\boldsymbol{\delta}^\ell_{\text{FA}} = \left(\boldsymbol{\delta}^{\ell+1}_{\text{FA}} \odot \phi^{\ell\prime}\right) \mathbf{B}^{\ell+1}. \tag{5}$$

While FA approximates the operators used to transport errors, it still requires sequential propagation across layers. Direct Feedback Alignment (DFA) (Nøkland, 2016) removes this dependency by projecting the output error $\boldsymbol{\delta}^L$ directly to each hidden layer through a fixed feedback matrix $\mathbf{B}^\ell$:

$$\boldsymbol{\delta}^\ell_{\text{DFA}} = \boldsymbol{\delta}^L \mathbf{B}^{\ell+1} \tag{6}$$

Despite their aggressive approximations, FA and DFA can provide useful learning signals and have been successfully applied to deep models at nontrivial scales (Launay et al., 2020). This suggests that fixed random feedback operators can retain enough information to support effective optimization.

We transfer this principle from depth to time. Since an RNN repeatedly applies the same recurrent cell, we replace its time-dependent state Jacobians with a shared feedback operator. DSF thus inherits its approximation principle from FA, while the resulting stationary backward dynamics enable parallel computation without removing the recurrent pathway for error propagation across time.

3 DIRECT STATE FEEDBACK (DSF)

Motivated by Feedback Alignment, we approximate the transport of errors across recurrent states by replacing the time-dependent Jacobians $\mathbf{A}_t$ with a shared feedback operator $\mathbf{A}$. Since an RNN repeatedly applies the same recurrent cell across time, using a single feedback operator provides a natural temporal counterpart to the fixed layer-wise operators used by FA. We then impose a diagonal structure on $\mathbf{A}$ to make the resulting backward recurrence computationally efficient. Importantly, both approximations apply only to backward-time gradient transport: the nonlinear forward dynamics and the local parameter derivatives remain unchanged.

3.1 STATIONARY APPROXIMATION

In standard BPTT, computing the gradient requires recursively multiplying the state Jacobians $\mathbf{A}_t = \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}}$. Although the recurrent cell shares parameters across time, each Jacobian is evaluated at different hidden states $\mathbf{h}_{t-1}$ and inputs $\mathbf{x}_t$. The Jacobians are therefore time-varying, making direct gradient transport sequential in T.

Following the approximation principle of FA, we replace the time-dependent Jacobians $\mathbf{A}_t$ with a single time-invariant feedback operator $\mathbf{A}$. This corresponds to a stationary approximation of the backward error-transport operator, while the forward dynamics $\mathbf{h}_t = f_{\boldsymbol{\theta}}(\mathbf{h}_{t-1}, \mathbf{x}_t)$ remain unchanged. Under this approximation, the BPTT recurrence in Eq. 3 becomes:

$$\tilde{\mathbf{g}}_t = \mathbf{e}_t + \tilde{\mathbf{g}}_{t+1}\mathbf{A}. \tag{7}$$

The matrix $\mathbf{A}$ acts as a stationary feedback operator, transporting error signals from future time steps toward earlier recurrent states.

Unlike the original BPTT recurrence, this is a linear time-invariant system. It can therefore be evaluated using parallel prefix-scan algorithms (Hillis & Steele, 1986; Blelloch, 1990), reducing its sequential depth from $\mathcal{O}(T)$ to $\mathcal{O}(\log T)$. We use a parallel scan in our implementation and further constrain the structure of $\mathbf{A}$ for efficiency, as discussed next. Initialization choices of $\mathbf{A}$ are discussed in section 3.3.

3.2 DIAGONAL APPROXIMATION

If $\mathbf{A}$ is an arbitrary $d \times d$ matrix, composing the feedback operators, as required by the parallel scan, involves costly $O(d^3)$ dense matrix operations. We therefore choose a diagonal feedback operator,

$$\mathbf{A} = \mathrm{diag}(a_1, \ldots, a_d), \tag{8}$$

so that its application and composition reduce to $\mathcal{O}(d)$ element-wise operations. Diagonal transition operators are also commonly used to make linear recurrences computationally efficient (Gu et al., 2022; Gupta et al., 2022).

The combination of stationary and diagonal feedback defines *Direct State Feedback* (DSF). Although it substantially simplifies the temporal gradient computed by BPTT, it retains a recurrent pathway through which error signals can propagate across the full sequence. We provide a conditional bound on the resulting gradient-transport error in Section 3.3, and show empirically in Section 4 that, akin to the Feedback Alignment methods, the approximation recovers a substantial portion of the training quality.

3.3 THEORETICAL BOUND ON GRADIENT APPROXIMATION ERROR

To quantify how well DSF approximates exact backpropagation through time, we derive a uniform upper bound on the discrepancy $\delta_t = \mathbf{g}_t - \tilde{\mathbf{g}}_t$ between the true BPTT gradient $\mathbf{g}_t$ and its DSF approximation $\tilde{\mathbf{g}}_t$. Under the contractivity assumptions ($\|\mathbf{A}_t\| \leq M < 1$ and $\|\mathbf{A}\| \leq M < 1$) and a bounded Jacobian deviation ($\|\mathbf{A}_t - \mathbf{A}\| \leq \varepsilon$), Proposition 1 shows that this error remains uniformly bounded for any sequence length T. This is a worst-case bound and can be loose when M is close to 1; nevertheless, Table 4 provides empirical evidence that DSF remains stable as the sequence length increases.

Proposition 1 (Uniform Gradient Approximation Error) *Let $\mathbf{g}_t$ be defined by Eq. 3 and $\tilde{\mathbf{g}}_t$ by Eq. 7. Assume that, for all $1 \leq t < T$ and $1 \leq s \leq T$:*

$$\textbf{(i)}\ \|\mathbf{A}_{t+1} - \mathbf{A}\| \leq \varepsilon, \qquad \textbf{(ii)}\ \|\mathbf{A}_{t+1}\| \leq M < 1 \text{ and } \|\mathbf{A}\| \leq M < 1, \qquad \textbf{(iii)}\ \|\mathbf{e}_s\| \leq E.$$

Then for any t, writing $n = T - t$, the gradient approximation error $\mathbf{g}_t - \tilde{\mathbf{g}}_t$ satisfies:

$$\|\mathbf{g}_t - \tilde{\mathbf{g}}_t\| \leq \frac{\varepsilon\, E}{(1-M)^2}\Big(1 - (n+1)M^n + nM^{n+1}\Big) \leq \frac{\varepsilon\, E}{(1-M)^2}. \tag{9}$$

Moreover, as $T - t \to \infty$, both $M^n \to 0$ and $nM^n \to 0$, yielding:

$$\lim_{T-t\to\infty} \|\mathbf{g}_t - \tilde{\mathbf{g}}_t\| \leq \frac{\varepsilon\, E}{(1-M)^2}. \tag{10}$$

Proof. See Appendix B.1 for the full derivation.

Choice of A. The above Proposition 1 provides insights about suitable choices for $\mathbf{A} = \mathrm{diag}(a_1, \ldots, a_d)$. It requires $\|\mathbf{A}\|_2 = \max_i |a_i| = \rho < 1$, meaning that the values of $\mathbf{A}$ must live strictly between -1 and 1. In our experiments we used a deterministic schedule with values uniformly spaced between 0.1 and 0.9 included (e.g. $\mathbf{A} = \mathrm{diag}(\texttt{torch.linspace}(0.1, 0.9, d))$). Ablations regarding other choices can be found in Appendix C.

3.4 T-DSF: Combining T-BPTT and DSF

Truncated BPTT computes exact gradients only within a horizon K, while DSF provides an efficient approximation to transport gradients beyond that horizon. We combine both in **T-DSF** by running *exact* BPTT on segments of length K and using the stationary DSF recursion to approximate longer-range gradients. Concretely, considering a segment from t to $t + K - 1$, the Jacobians $\mathbf{A}_t$ inside the segment are kept exact, as in BPTT, but the Jacobians of cells at positions $t' \geq t + K$ are approximated using DSF with a fixed diagonal matrix $\mathbf{A}$. In practice this is done by first running DSF to approximate the true gradients $\mathbf{g}_t$ by $\tilde{\mathbf{g}}_t$ at the end of each segment:

$$\tilde{\mathbf{g}}_t = \mathbf{e}_t + \tilde{\mathbf{g}}_{t+1}\mathbf{A}. \tag{11}$$

We then use these approximate gradients to initialize the truncated BPTT at the boundaries of each segment $[t; t + K - 1]$, and use regular BPTT inside them:

$$\bar{\mathbf{g}}_{t+K} = \tilde{\mathbf{g}}_{t+K}, \tag{12}$$

$$\bar{\mathbf{g}}_s = \mathbf{e}_s + \bar{\mathbf{g}}_{s+1}\mathbf{A}_{s+1}, \qquad s \in \{t, \ldots, t + K - 1\}. \tag{13}$$

Note that, just as DSF reduces to FT-BPTT when $\mathbf{A} = 0$, T-DSF reduces to T-BPTT under the same choice. T-DSF is thus a natural generalization of T-BPTT. See Figure 1.e for a visual representation.

3.5 Computational Complexity Analysis

We now analyze the computational complexity and sequentiality of gradient computations for the standard BPTT and our proposed DSF method. We summarize the complexity results discussed below in Table 1. This table illustrates how the proposed DSF approach offers substantial computa-

Table 1: Complexity and sequentiality of the methods, for the temporal error propagation step.

Method	**Complexity**	**Sequentiality**
BPTT	$\mathcal{O}(d^2 T)$	$\mathcal{O}(T)$
Fully Truncated BPTT (FT-BPTT)	$\mathcal{O}(1)$	$\mathcal{O}(1)$
DSF	$\mathcal{O}(d\, T \log T)$	$\mathcal{O}(\log T)$
Truncated BPTT (T-BPTT)	$\mathcal{O}(d^2\, T)$	$\mathcal{O}(K)$
T-DSF	$\mathcal{O}(d^2 T + d\, T \log T)$	$\mathcal{O}(K + \log(T))$

tional benefits compared to standard BPTT. The diagonal structure of $\mathbf{A}$ leads to a linear complexity in the dimension d, while the prefix scan algorithm is parallelizable with only $\log_2 T$ sequential steps. DSF provides a favorable balance between computational efficiency, memory usage, and sequential complexity, making it attractive for large-scale recurrent network training. We validate its computational efficiency empirically in section 4.4.

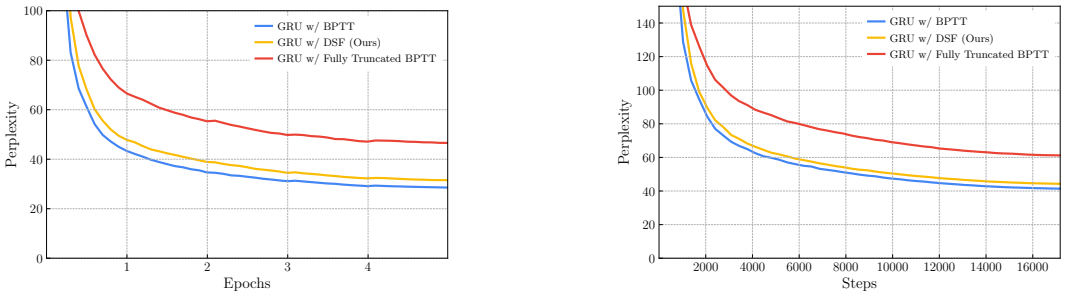
4 Experimental Results and Analysis

4.1 Experimental Setup

We evaluate DSF and T-DSF on language modeling, with the Wikitext-103 and OpenWebText datasets, as well as two synthetic tasks in Section 4.6. Our backbone follows GPT-2, with self-attention replaced by recurrent layers: Wikitext-103 uses 3-layer GRUs with hidden size 512 and context length 256, and the main OpenWebText setting uses 6-layer GRUs with hidden size 512 and context length 1024. We additionally experiment with other RNN types, including sLSTM (Beck et al., 2024), a modernized variant of LSTM. We compare against BPTT, FT-BPTT, and T-BPTT. Unless specified, DSF uses the fixed diagonal feedback matrix $\mathbf{A} = \mathrm{diag}(\mathrm{linspace}(0.1, 0.9, d))$, as motivated by Appendix C. Full experimental details are given in Appendix A.

4.2 Approximating BPTT for Language Modeling

We report the validation perplexity during training on Wikitext-103 and OpenWebText in Figure 2. Across both datasets, DSF consistently outperforms FT-BPTT and narrows the gap to full BPTT. This shows that a fixed backward-time transport can preserve useful temporal credit assignment despite a strong simplification of the gradient recursion.



(a) Wikitext-103 – 28.2M model parameters.

(b) OpenWebText – 75.2M model parameters.

Figure 2: Validation perplexity during training for Wikitext-103 (left, 3 layers with hidden size 512) and OpenWebText (right, 6 layers with hidden size 512).

Larger scale. To test whether these trends persist beyond the $\sim$75M-parameter regime, we run larger-scale OpenWebText experiments with 12-layer GRU and sLSTM models of hidden size 768 and comparable parameter counts ($\sim$183M parameters). Table 2 compares the three learning rules across both architectures. On the GRU, DSF nearly matches full BPTT (37.39 vs. 36.20 PPL) while substantially outperforming FT-BPTT (46.52). The same qualitative pattern holds for the sLSTM: DSF reaches 35.12 PPL in 84.7 hours on a single H100, improving markedly over FT-BPTT (46.52 in 71.3 hours), although a larger gap to full BPTT remains (28.08 in 140 hours). These results show that fixed diagonal backward-time transport continues to provide useful long-range credit assignment at larger model scales and transfers beyond the GRU architecture.

Scaling beyond one billion parameters. We further test DSF at larger scale using a 36-layer GRU with hidden size 1280 ($>$1B parameters), trained for one epoch on OpenWebText under the same protocol for all learning rules. As shown in Table 3, DSF remains within approximately 3% of FT-BPTT runtime and achieves lower validation perplexity than BPTT in this run. We do not interpret this single one-epoch result as evidence that DSF generally outperforms BPTT, but rather that the approximation remains effective and computationally advantageous at this scale.

One possible explanation is that DSF attenuates long-range learning signals less strongly than exact BPTT in this deep recurrent regime. This is qualitatively consistent with Section 4.7, where DSF gradients have larger norms at long horizons while remaining positively aligned with BPTT. These diagnostics, however, do not establish vanishing gradients as the cause of the observed perplexity difference, and a systematic investigation of this effect is left to future work.

Table 2: ∼183M parameters OpenWebText experiments: 12-layer GRU and sLSTM models with hidden size 768.

	BPTT	DSF	FT-BPTT
GRU	36.20	37.39	46.52
sLSTM	28.08	35.12	46.52

Table 3: ∼1B parameters OpenWebText experiments: 36-layer GRU with hidden size 1280.

Method	PPL ↓	Time ↓
BPTT	28.68	35.70 h
DSF	22.54	29.28 h
FT-BPTT	29.28	28.47 h

4.3 Generality Across Recurrent Architectures and Configurations

4.3.1 Across Recurrent Cells

The DSF framework is agnostic of the type of recurrent cell, we thus investigate how DSF performs for different RNNs: vanilla RNNs, GRUs, LSTMs, and sLSTMs. Figure 3 summarizes the best validation perplexity of each RNN type.

DSF is effective across all recurrent types and maintains performance close to BPTT. For gated RNNs, this may reflect a synergy between gating and diagonal backward-time transport: the gates already organize memory along coordinate-wise channels, making diagonal feedback particularly natural. The small DSF–BPTT gap for vanilla RNNs is also noteworthy, and suggests that even simple recurrent dynamics may contain a substantial diagonal component in their memory mechanism.

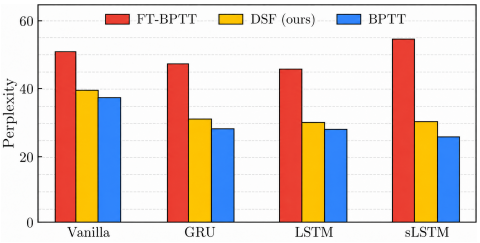


Figure 3: Perplexity across different RNN types on WikiText-103, using 3 layers with hidden size 512.

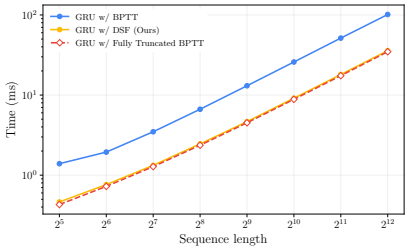


Figure 4: Computation times of different learning algorithms with respect to sequence length.

4.3.2 Across Model and Sequence Configurations

We study how DSF behaves as we increase (i) context length, (ii) hidden dimension, and (iii) number of layers. Table 4 shows that DSF is robust across sequence length, width, and depth, for the Wikitext-103 setting. Increasing the context length from 256 to 2048 does not cause degradation: DSF stays far above FT-BPTT and within a stable gap of BPTT, consistent with contractive stationary gradient transport. As width increases, all methods improve and the DSF–BPTT gap remains roughly stable, suggesting that diagonal feedback is not a bottleneck for scaling hidden dimension. With depth, DSF also improves and preserves the ordering BPTT $>$ DSF $\gg$ FT-BPTT, although the gap to BPTT grows moderately, as deeper models rely more on accurate temporal credit assignment.

Table 4: Validation perplexity on Wikitext-103 while varying either the context length, the hidden dimension, or the number of layers.

	Context length				Hidden size				Nb. of layers			
	256	**512**	**1024**	**2048**	**128**	**256**	**512**	**1024**	**1**	**3**	**6**	**12**
BPTT	28.32	27.14	26.75	26.52	51.17	36.67	28.32	24.19	35.59	28.32	25.44	23.73
DSF (ours)	31.54	30.17	30.64	30.22	56.14	40.39	31.54	27.16	37.42	31.54	29.62	28.99
FT-BPTT	46.60	45.95	46.75	46.52	82.91	59.61	46.60	38.29	56.61	46.60	41.35	36.59

4.4 Computation Time

The computational cost of the different layers is reported in Figure 4. We use a batch size of 128 and hidden dimension 512. As expected from the theoretical complexities in Table 1, the runtime grows approximately linearly with sequence length for all methods. With our optimized CUDA implementation, DSF closely matches the runtime of FT-BPTT across the entire range of sequence lengths, while remaining substantially faster than full BPTT. At long sequences, both DSF and FT-BPTT are roughly three times faster than BPTT. This shows that the stationary backward recurrence of DSF can be implemented with essentially the same practical cost as fully truncated BPTT, while retaining long-range backward credit transport.
For the hybrid method T-DSF, the relevant comparison is T-BPTT at matched horizon. Appendix A.8 reports that, on the 6-layer OpenWebText GRU setup with sequence length 1024, forward+backward time for one layer is 29.00 ± 0.09 ms for T-BPTT with $K = 64$ and 29.62 ± 0.06 ms for T-DSF with the same K. Thus, T-DSF should be viewed as a low-overhead extension of T-BPTT that improves the training signal, rather than as a strictly faster replacement at fixed truncation horizon.

4.5 Combining Truncated BPTT and DSF (T-DSF)

In this section we evaluate T-DSF (described in section 3.4), which naturally combines T-BPTT and DSF. It enhances T-BPTT by using DSF to approximate long-range gradients, while T-BPTT completely discards them. Exact BPTT is then performed within each segment of size K. Table 5 shows that T-DSF consistently improves over T-BPTT at matched K, especially for small horizons. As K grows, both methods approach BPTT, but T-DSF always reaches a better perplexity with minimal overhead (see Appendix A.8 for time measurements).

Table 5: Perplexity for T-BPTT vs. T-DSF for both Wikitext-103 and OpenWebText experiments.

Wikitext-103	$K=1$	2	4	8	16	32	64	BPTT
T-BPTT	46.60	38.88	34.28	31.66	30.05	29.22	28.82	28.55
T-DSF	31.54	30.75	29.77	29.30	29.03	28.95	28.65	

OpenWebText	$K=1$	4	16	64	BPTT
T-BPTT	61.21	50.69	44.28	42.22	41.42
T-DSF	44.25	43.29	42.31	41.89	

4.6 Selective and Long-Range Memory

To test whether DSF learns dependencies beyond short-range language-modeling signals, we use two memory benchmarks. MQAR (Arora et al., 2024) probes *selective* associative recall from key-value pairs, while the copy task isolates pure long-range credit assignment with delays uniformly sampled from 0 to 8192.
As shown in Table 6, DSF substantially improves over FT-BPTT on both tasks, indicating that purely local updates are insufficient for selective or long-range memory. On MQAR, DSF closes much of the gap to the BPTT baseline. On the copy task, DSF transports useful credit far beyond the truncation horizon of T-BPTT, and T-DSF further improves over DSF.

Table 6: Selective and long-range memory benchmarks. Left: MQAR exact-match accuracy (mean $\pm$ std over 5 runs). Right: copy task with delays sampled from 0 to 8192, reporting token-wise accuracy, exact-match accuracy, and equivalent solved length.

Method	**Exact** $\uparrow$
BPTT	**63.4**$\pm$**1.7**
FT-BPTT	15.3$\pm$0.4
DSF	55.7$\pm$0.9

Method	**Tok.** $\uparrow$	**Exact** $\uparrow$	**Len.** $\uparrow$
BPTT	**99.93**	**99.50**	**8151**
FT-BPTT	0.56	0.00	0
T-BPTT, $K = 16$	44.72	6.64	544
DSF	82.84	26.87	2201
T-DSF, $K = 16$	85.87	37.69	3088

4.7 Alignment with exact BPTT

To better understand the learning signal produced by DSF, we compare DSF and BPTT gradients on identical forward passes, keeping model parameters, hidden states, outputs, and losses fixed and changing only the temporal gradient-transport rule. As sequence length increases from 16 to 4096, cosine similarity with the BPTT gradient decreases from 0.763 to 0.114 for recurrent parameters and from 0.741 to 0.151 for input parameters. Despite this increasing angular error, coordinate-wise sign agreement remains approximately 85–89% throughout the tested range. At the same time, relative error and gradient-norm ratios increase with sequence length.

These results show that DSF should not be interpreted as reconstructing the exact BPTT gradient at long horizons. However, it preserves a positively aligned and strongly sign-consistent first-order learning signal, even as its magnitude and angular accuracy deteriorate with sequence length. Full diagnostic results are reported in Appendix A.9.

5 Discussion and Future Directions

Our results show that DSF offers a practical trade-off between efficiency and accuracy for training any recurrent networks. By replacing time-varying BPTT Jacobians with a fixed diagonal feedback operator, DSF reduces training cost while retaining useful long-range credit assignment. Across architectures, context lengths, and model scales, DSF consistently improves over truncated BPTT and remains reasonably close to full BPTT. The gap stays roughly stable with width and increases only moderately with depth, suggesting that stationary backward transport remains effective as model capacity grows. Several directions could further refine DSF:

- **Hybridization and adaptive horizons.** T-DSF already provides a smooth accuracy-sequentiality trade-off by mixing exact BPTT over the last K steps with DSF transport beyond that horizon. A natural extension is to choose K adaptively based on compute budget or optimization signals.
- **Richer structured feedback.** Beyond diagonal feedback, low-rank+diagonal or other structured operators could improve gradient transport at modest additional cost, potentially narrowing the DSF–BPTT gap while preserving parallelizable implementations.
- **Initialization and stability.** Our diagonal schedule (with $\|\mathbf{A}\|_2 = \rho < 1$) enforces contractivity and yields stable training. Exploring alternative schedules or data-driven calibrations of ρ may improve performance, while keeping the backward operator stable.
- **Theory beyond worst-case bounds.** Our uniform bound guarantees that the approximation error stays controlled under contractivity and bounded deviation assumptions, but it is conservative when M is close to 1. Sharper analyses (e.g., average-case bounds or task-dependent characterizations) could better explain when stationary feedback is sufficient.

Overall, DSF suggests that carefully designed *backward-time* structure can substantially reduce the sequential bottleneck of BPTT without eliminating the ability of recurrent models to learn long-range dependencies.

6 Conclusion

We introduced Direct State Feedback (DSF), an efficient approximation to BPTT for recurrent neural networks. Inspired by Feedback Alignment, DSF replaces the time-varying state Jacobians used for temporal error transport with a fixed diagonal feedback operator, while leaving the nonlinear forward recurrence unchanged. Under contractivity and bounded-deviation assumptions, we provide a uniform bound on the resulting gradient approximation error.

Across language modeling and controlled memory tasks, DSF preserves much of the training quality of exact BPTT while improving over fully truncated updates and reducing backward sequentiality. We further introduced T-DSF, which combines exact BPTT within short segments with stationary feedback across segment boundaries, consistently improving over T-BPTT at comparable cost.

Overall, our results show that exact temporal Jacobian transport is not always necessary for effective recurrent learning. Fixed stationary feedback can preserve useful long-range credit assignment while enabling more efficient gradient computation. Future work includes richer structured feedback (e.g., low-rank+diagonal), adaptive horizons or feedback schedules, and sharper theoretical characterizations beyond worst-case bounds.

AI USE STATEMENT

Generative AI tools were used for language editing and debugging support. All AI-assisted text was reviewed by the authors. The authors take full responsibility for the paper's claims, results, citations, and final content.

REPRODUCIBILITY STATEMENT

We provide the information needed to reproduce our theoretical and empirical results throughout the paper and supplementary material. The assumptions and complete proofs of the theoretical results are given in the appendix. The experimental setup, model architectures, datasets, preprocessing, optimization hyperparameters, and evaluation protocols are documented in the experimental and appendix sections. Code and configuration files will be released upon publication.

REFERENCES

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Re. Zoology: Measuring and improving recall in efficient language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=LY3ukUANko.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2014.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xlstm: Extended long short-term memory. 2024.

Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, 11, 07 2020. doi: 10.1038/s41467-020-17236-y.

Guy E. Blelloch. Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University, November 1990.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. 2014.

Federico Danieli, Miguel Sarabia, Xavier Suau Cuadros, Pau Rodriguez, and Luca Zappella. Deeppcr: Parallelizing sequential operations in neural networks. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 47598–47625. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/948d8ba4e30c8c3a800cf436b31f376e-Paper-Conference.pdf.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Alex Graves and Jürgen Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. *Advances in neural information processing systems*, 21, 2008.

Albert Gu, Ankit Gupta, Karan Goel, and Christopher Ré. On the parameterization and initialization of diagonal state space models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, 2022.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. *Advances in Neural Information Processing Systems*, 35:22982–22994, 2022.

William D. Hillis and Guy L. Steele. Data parallel algorithms. *Commun. ACM*, 29:1170–1183, 1986. URL https://api.semanticscholar.org/CorpusID:2315965.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Michael I Jordan. Serial order: A parallel distributed processing approach. In *Advances in psychology*, volume 121, pp. 471–495. Elsevier, 1997.

Julien Launay, Iacopo Poli, François Boniface, and Florent Krzakala. Direct feedback alignment scales to modern deep learning tasks and architectures. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Timothy P. Lillicrap, Daniel Cownden, Douglas Blair Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7, 2016. URL https://api.semanticscholar.org/CorpusID:10050777.

Yi Heng Lim, Qi Zhu, Joshua Selfridge, and Muhammad Firmansyah Kasim. Parallelizing non-linear sequential models over the sequence length. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=E34AlVLN0v.

Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. *Advances in neural information processing systems*, 29, 2016.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In *International Conference on Machine Learning*. PMLR, 2023.

Ronald J. Williams and Jing Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501, 1990. doi: 10.1162/neco.1990.2.4.490.

Nicolas Zucchet, Robert Meier, Simon Schug, Asier Mujika, and João Sacramento. Online learning of long-range dependencies. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

A Experimental Setup

A.1 Datasets

Wikitext-103 The Wikitext-103 dataset is a larger version of the Wikitext-2 dataset, and is also a widely used benchmark for language modeling. It consists of over 100 million tokens from Wikipedia articles. We use a vocabulary size of 16k, and a default context length of 256.

OpenWebText The OpenWebText dataset is an open-source recreation of OpenAI's WebText corpus, comprising text scraped from URLs shared on Reddit with at least three upvotes. It contains approximately 38 GB of raw text—on the order of 200 million tokens. We process it using the GPT-2 tokenizer, with vocabulary size 50,257, and a context length of 1024 as in the original GPT-2 paper.

A.2 Models

A.2.1 RNNs

- **Vanilla RNN.** The simplest form of RNN, where the hidden state is updated as
$$\mathbf{h}_t = \sigma(W_1\mathbf{x}_t + W_2\mathbf{h}_{t-1} + b).$$
- **GRU** (Cho et al., 2014). A gated RNN, with the following update equations:
$$\begin{aligned}\mathbf{z}_t &= \sigma\big(W_z x_t + U_z h_{t-1} + b_z\big),\\ \mathbf{r}_t &= \sigma\big(W_r x_t + U_r h_{t-1} + b_r\big),\\ h_t &= (1-\mathbf{z}_t)\odot h_{t-1} + \mathbf{z}_t \odot \tanh\big(W_h x_t + U_h(\,\mathbf{r}_t \odot h_{t-1}\,) + b_h\big).\end{aligned}$$
- **LSTM** (Hochreiter & Schmidhuber, 1997). A gated RNN with a cell state $\mathbf{c}_t$ and hidden state $\mathbf{h}_t$. The update equations are:
$$\begin{aligned}\mathbf{i}_t &= \sigma(W_i\mathbf{x}_t + U_i\mathbf{h}_{t-1} + b_i),\\ \mathbf{f}_t &= \sigma(W_f\mathbf{x}_t + U_f\mathbf{h}_{t-1} + b_f),\\ \mathbf{o}_t &= \sigma(W_o\mathbf{x}_t + U_o\mathbf{h}_{t-1} + b_o),\\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(W_c\mathbf{x}_t + U_c\mathbf{h}_{t-1} + b_c),\\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t).\end{aligned}$$
While an LSTM usually outputs only the hidden state $\mathbf{h}_t$, we instead use the concatenation of both $\mathbf{c}_t$ and $\mathbf{h}_t$ to improve the performance of the model.
- **sLSTM** (Beck et al., 2024) We additionally evaluate the sLSTM cell of xLSTM, a modern gated recurrent architecture. We use the same outer residual language-model architecture as for the other recurrent cells. For exact BPTT, we use the CUDA implementation of the sLSTM recurrence. For DSF and FT-BPTT, we keep the same nonlinear sLSTM forward dynamics and modify only the temporal gradient transport: DSF uses the fixed diagonal feedback operator described in Section 3, while FT-BPTT removes recurrent-state gradient propagation. The WikiText-103 experiments use 3 layers with hidden size 512 and 4 heads. The larger OpenWebText experiments use 12 layers with hidden size 768.

A.2.2 Linearized GRU Control

We also consider a linearized recurrent control designed to isolate the role of the nonlinear forward recurrence. The model uses the transition
$$h_t = \mathbf{A}h_{t-1} + (I - \mathbf{A})\tanh(W_{ih}x_t),$$
where $\mathbf{A}$ is diagonal. This model is informative because, when $\mathbf{A}$ is fixed as in DSF, its exact hidden-state backward recursion is governed by the same stationary operator used by DSF. We evaluate two variants:

- a fixed-$\mathbf{A}$ variant, where $\mathbf{A}$ is chosen using the same diagonal schedule as DSF;
- a learned-$\mathbf{A}$ variant, initialized from the DSF schedule and then optimized during training.

This control distinguishes exact training of a forward-linear recurrence from approximate DSF training of a nonlinear recurrent model.

A.2.3 Gated Diagonal Feedback Variant

For GRUs, we also evaluate a stronger time-varying diagonal feedback rule. In this variant, the fixed DSF operator $\mathbf{A}$ is replaced by a time-dependent diagonal operator derived from the GRU update gate:

$$\mathbf{A}_t = I - \mathrm{diag}(\mathbf{z}_t).$$

This variant is more faithful to the local recurrent dynamics than fixed DSF, but it is also model-specific and no longer stationary. We therefore treat it as a practical extension and diagnostic comparison rather than as the main DSF algorithm.

A.3 Training

A.3.1 Setup

All our experiments are run on A100 and H100 GPUs. We use the AdamW optimizer, with linear warmup and cosine learning-rate scheduling. We use TF32 precision for the Wikitext-103 models, and FP16 for OpenWebText, to speed up training without performance degradation. The sLSTM experiments use BF16 on H100 GPUs.

Unless stated otherwise, OpenWebText experiments with 6 layers use the same training setup as the main 6-layer GRU model. The larger-scale OpenWebText experiment uses a 12-layer GRU with hidden size 768. The linearized-GRU controls, gated diagonal feedback variant, and feedback-matrix ablations are all evaluated in the 6-layer OpenWebText setting.

A.3.2 Hyperparameters

We report the hyperparameters used for training the main language models in Table 7.

Table 7: Hyperparameters for the models used in the experiments, unless stated otherwise.

Dataset	**Wikitext-103**	**OpenWebText**		
Layers	3	6	12	36
Hidden dimension	512	512	768	1280
Vocabulary size	16k	50,257	50,257	50,257
Context length	256	1024	1024	2048
Training steps	5 epochs	17.2k steps	51.6k steps	17.2k steps
Batch size	128	64	128	256
Learning rate	10^{-3}	10^{-3}	$3 \cdot 10^{-4}$	$2 \cdot 10^{-4}$
Adam β_2	0.999	0.98	0.98	0.95
Warmup steps	10%	10%	10%	2%

A.4 Multi-Query Associative Recall

We evaluate selective memory using the Multi-Query Associative Recall (MQAR) benchmark (Arora et al., 2024). MQAR requires the model to store several key-value pairs and answer multiple queries referring to different keys. This task probes whether a model can learn content-based retrieval rather than only short-range smoothing or local pattern matching.

We report exact-match accuracy averaged over 5 runs. Unless stated otherwise, all compared models use the same architecture size and optimization protocol within the MQAR experiment. The baselines are full BPTT, FT-BPTT, and DSF.

A.5 Long-Range Copy Task

To isolate long-range credit assignment from the complexities of language modeling, we also evaluate a controlled copy task. Delays are sampled uniformly from the interval $[0, 8192]$. The model must remember a symbol over a sampled delay and reproduce it at the appropriate output position.

We report three metrics:

- **Token-wise accuracy**: the fraction of copied tokens predicted correctly.
- **Exact-match accuracy**: the fraction of examples for which the entire copied sequence is predicted correctly.
- **Equivalent length**: an estimate of the effective delay length solved by the model, reported by the evaluation protocol.

The task is designed to test whether the learning rule can propagate useful credit over distances that are far beyond a small truncation horizon. In particular, we compare BPTT, FT-BPTT, T-BPTT with $K = 16$, DSF, T-DSF with $K = 16$.

A.6 Additional OpenWebText Comparisons

Table 8 reports the linearized-GRU and gated-DSF controls. The linearized-GRU results show that exact training of a forward-linear recurrence performs worse than approximate DSF training of a nonlinear GRU. The gated-DSF variant shows that time-varying diagonal feedback can further narrow the gap to BPTT, at the cost of losing the stationary, architecture-agnostic form of DSF.

Table 8: Linearized-GRU and gated-feedback controls on OpenWebText.

Method	Validation PPL ↓
GRU w/ BPTT	**40.78**
GRU w/ DSF	44.60
GRU w/ gated DSF	42.37
Linear GRU, fixed $\mathbf{A}$	47.92
Linear GRU, learned $\mathbf{A}$	47.19

A.7 Comparison with an e-prop-style Variant

DSF is related to the literature on approximate online and local learning for recurrent networks, including e-prop-style methods. To clarify this connection, we evaluate a variant in which only local gradients are backpropagated across layers, while the temporal approximation is used to update the recurrent weights of each layer independently. This is closer in spirit to e-prop/RTRL-style approximations (Bellec et al., 2020; Zucchet et al., 2023), but differs from DSF, which remains an offline approximation to BPTT and preserves standard spatial backpropagation across layers.

Table 9 shows that this e-prop-style variant performs worse than DSF, although it remains substantially better than FT-BPTT. This suggests that propagating the approximate temporal error through the stack of recurrent layers is useful in the offline setting considered in this paper.

Table 9: OpenWebText comparison with an e-prop-style DSF variant.

Method	Validation PPL ↓
BPTT	**40.78**
FT-BPTT	61.21
DSF	44.60
DSF, e-prop style	47.91

A.8 Matched-Horizon Timing for T-DSF

For the hybrid T-DSF method, the most relevant practical comparison is T-BPTT at matched truncation horizon K. Table 10 reports forward+backward wall-clock time for one recurrent layer in the 6-layer OpenWebText GRU setup with sequence length 1024.

The timings show that T-DSF is close to T-BPTT in wall-clock cost at matched K, with a small overhead due to the DSF-based boundary-gradient initialization. Thus, the benefit of T-DSF is not that it is strictly faster than T-BPTT at the same horizon, but that it improves the training signal and perplexity for a small additional cost.

Table 10: Matched-K wall-clock comparison. Times correspond to forward and backward passes for one layer in the 6-layer GRU OpenWebText setup with sequence length 1024.

Method	**Time (ms)** $\downarrow$
BPTT	32.57 ± 0.10
T-BPTT, $K = 64$	29.00 ± 0.09
T-DSF, $K = 64$	29.62 ± 0.06

A.9 Gradient diagnostics against exact BPTT

The theoretical analysis in Section 3.3 provides a worst-case bound on the difference between exact BPTT and DSF under contractivity and bounded-deviation assumptions. These assumptions are deliberately sufficient rather than intended to exactly characterize the dynamics of trained nonlinear recurrent networks. We therefore complement the theoretical analysis with a direct empirical comparison of the gradients produced by DSF and exact BPTT.

Protocol. We compare both learning rules on exactly the same forward pass. Model parameters, inputs, hidden states, predictions, and loss are identical in the two branches; the only difference is the temporal gradient-transport rule used during the backward pass. Thus, any discrepancy reported below is caused by replacing the sequence of exact recurrent Jacobians with the fixed DSF feedback operator.

We separately analyze two parameter families: the recurrent parameters and the input parameters of each recurrent layer. For each family, all corresponding gradient tensors are flattened and concatenated into a single vector. We denote the resulting exact and approximate gradients by g_{BPTT} and g_{DSF}, respectively.

For each parameter family, we report four complementary diagnostics:

$$\text{Cosine} = \frac{\langle g_{\mathrm{BPTT}}, g_{\mathrm{DSF}} \rangle}{\|g_{\mathrm{BPTT}}\|_2 \|g_{\mathrm{DSF}}\|_2}, \tag{14}$$

$$\text{Relative } \ell_2 = \frac{\|g_{\mathrm{DSF}} - g_{\mathrm{BPTT}}\|_2}{\|g_{\mathrm{BPTT}}\|_2}, \tag{15}$$

$$\text{Norm ratio} = \frac{\|g_{\mathrm{DSF}}\|_2}{\|g_{\mathrm{BPTT}}\|_2}, \tag{16}$$

$$\text{Sign agreement} = \frac{100}{N} \sum_{i=1}^{N} \mathbf{1}\left[\operatorname{sign}(g_{\mathrm{DSF},i}) = \operatorname{sign}(g_{\mathrm{BPTT},i})\right]. \tag{17}$$

We vary the sequence length from 16 to 4096 tokens. Reported values are the mean $\pm$ standard deviation across five analyzed recurrent layers. These standard deviations measure variation across layers and should not be interpreted as independent-seed variability.

Alignment degrades smoothly with sequence length. Table 11 shows both the usefulness and the limitations of the DSF approximation. For recurrent parameters, cosine similarity decreases from 0.763 at $L = 16$ to 0.114 at $L = 4096$; for input parameters, it decreases from 0.741 to 0.151. At the same time, the relative ℓ_2 error increases substantially with sequence length. DSF should therefore not be interpreted as a numerically accurate reconstruction of the full BPTT gradient at long horizons.
Nevertheless, the cosine similarity remains positive for every tested sequence length and for both parameter families:

$$\langle g_{\mathrm{BPTT}}, g_{\mathrm{DSF}} \rangle > 0.$$

Thus, despite the increasing approximation error, the DSF update remains positively correlated with the exact BPTT direction across the entire tested range.

Sign structure is remarkably stable. The coordinate-wise sign agreement remains approximately 85–89% across all sequence lengths. In particular, it does not exhibit the strong monotonic degrada-

Table 11: Gradient diagnostics comparing DSF with exact BPTT on identical forward passes as sequence length L increases. Cosine similarity measures angular alignment, relative ℓ_2 error measures the discrepancy with respect to the BPTT gradient, norm ratio compares gradient magnitudes, and sign agreement is the percentage of coordinates having the same sign. Values are mean $\pm$ standard deviation across five analyzed layers.

	Recurrent parameters				**Input parameters**			
L	Cosine	Relative ℓ_2	Norm ratio	Sign (%)	Cosine	Relative ℓ_2	Norm ratio	Sign (%)
16	0.763 ± 0.007	0.896 ± 0.026	1.383 ± 0.022	87.72 ± 0.09	0.741 ± 0.005	0.910 ± 0.017	1.355 ± 0.016	86.37 ± 0.10
32	0.671 ± 0.008	1.198 ± 0.043	1.612 ± 0.041	86.76 ± 0.06	0.665 ± 0.007	1.130 ± 0.030	1.513 ± 0.028	85.43 ± 0.07
64	0.601 ± 0.019	1.473 ± 0.096	1.838 ± 0.087	86.46 ± 0.08	0.608 ± 0.019	1.333 ± 0.076	1.678 ± 0.066	85.05 ± 0.09
128	0.528 ± 0.026	1.825 ± 0.176	2.141 ± 0.164	86.64 ± 0.09	0.560 ± 0.017	1.521 ± 0.088	1.834 ± 0.081	85.11 ± 0.11
256	0.440 ± 0.048	2.406 ± 0.408	2.669 ± 0.384	87.11 ± 0.08	0.502 ± 0.039	1.794 ± 0.221	2.072 ± 0.200	85.46 ± 0.08
512	0.356 ± 0.051	3.239 ± 0.782	3.452 ± 0.758	87.07 ± 0.06	0.431 ± 0.059	2.247 ± 0.485	2.483 ± 0.455	85.42 ± 0.07
1024	0.258 ± 0.075	4.982 ± 1.482	5.138 ± 1.441	87.78 ± 0.15	0.334 ± 0.087	3.151 ± 0.853	3.332 ± 0.806	85.97 ± 0.02
2048	0.192 ± 0.080	7.721 ± 3.543	7.838 ± 3.500	87.77 ± 0.14	0.254 ± 0.105	4.692 ± 2.019	4.830 ± 1.966	85.97 ± 0.02
4096	0.114 ± 0.037	13.680 ± 5.460	13.753 ± 5.440	88.68 ± 0.12	0.151 ± 0.053	8.114 ± 3.110	8.197 ± 3.083	86.79 ± 0.04

tion observed for cosine similarity. This indicates that a large fraction of gradient coordinates retain the same orientation under DSF even when the exact relative magnitudes become increasingly distorted. Together with the positive cosine similarity, this provides evidence that DSF produces a structured first-order learning signal rather than an unrelated update direction.

Gradient magnitudes diverge at long horizons. The norm ratio increases systematically with sequence length. For recurrent parameters it grows from 1.38 at $L = 16$ to 13.75 at $L = 4096$, while for input parameters it grows from 1.36 to 8.20. DSF therefore increasingly produces larger gradient norms than exact BPTT as the temporal horizon grows. This observation is qualitatively consistent with the possibility that the fixed DSF dynamics attenuate long-range learning signals less strongly than the sequence of exact recurrent Jacobians.

Relation to the theoretical analysis. These diagnostics do not directly validate the assumptions of Proposition 1. In particular, positive gradient alignment does not imply a convergence guarantee, and the increasing relative error shows that DSF can substantially deviate from exact BPTT at long horizons. Although its angular and magnitude accuracy deteriorate with sequence length, DSF preserves positively aligned and strongly sign-consistent first-order information over a broad range of temporal horizons. This provides an empirical complement to the worst-case analysis and helps explain how a fixed backward-time operator can remain useful for optimization despite being a coarse approximation of exact BPTT.

B Proofs of the Mathematical Derivations

B.1 Proof of Proposition: Uniform Gradient Approximation Error

Define the approximation error

$$\delta_t = g_t - \tilde{g}_t,$$

where the exact and DSF gradients satisfy

$$g_T = e_T, \quad g_t = e_t + g_{t+1}A_{t+1}, \qquad \tilde{g}_T = e_T, \quad \tilde{g}_t = e_t + \tilde{g}_{t+1}A.$$

Here A_{t+1} denotes the exact backward-time Jacobian transport operator from time $t+1$ to time t, while A is the fixed DSF operator.

Under the assumptions

$$\|A_{t+1} - A\| \le \varepsilon, \qquad \|A_{t+1}\| \le M < 1, \qquad \|A\| \le M < 1, \qquad \|e_t\| \le E,$$

we first write a recurrence relation for δ_t. Subtracting the two recursions gives

$$\delta_t = (e_t + g_{t+1}A_{t+1}) - (e_t + \tilde{g}_{t+1}A) = \delta_{t+1}A_{t+1} + \tilde{g}_{t+1}(A_{t+1} - A).$$

Taking norms and using submultiplicativity,

$$\|\delta_t\| \le \|\delta_{t+1}\|\,\|A_{t+1}\| + \|\tilde{g}_{t+1}\|\,\|A_{t+1} - A\| \le M\|\delta_{t+1}\| + \varepsilon\,\|\tilde{g}_{t+1}\|. \tag{18}$$

Applying Eq. 18 recursively for $i = 1, \ldots, T-t$, and writing $n = T - t$, we obtain by induction:

$$\|\delta_t\| \leq \varepsilon \sum_{i=1}^{n} M^{i-1} \|\tilde{g}_{t+i}\|. \tag{19}$$

From the DSF recursion,

$$\tilde{g}_{t+i} = \sum_{k=t+i}^{T} e_k A^{k-(t+i)}.$$

Therefore,

$$\begin{aligned}\|\tilde{g}_{t+i}\| &\leq \sum_{k=t+i}^{T} \|e_k\| \, \|A\|^{k-(t+i)} \\ &\leq E \sum_{m=0}^{T-(t+i)} M^m = E \frac{1 - M^{n-i+1}}{1 - M}.\end{aligned}$$

Substituting this bound into Eq. 19, we get

$$\|\delta_t\| \leq \frac{\varepsilon E}{1 - M} \sum_{i=1}^{n} M^{i-1} \left(1 - M^{n-i+1}\right).$$

We now split the sum:

$$\sum_{i=1}^{n} M^{i-1} = \frac{1 - M^n}{1 - M},$$

and

$$\sum_{i=1}^{n} M^{i-1} M^{n-i+1} = \sum_{i=1}^{n} M^n = nM^n.$$

Hence,

$$\begin{aligned}\|\delta_t\| &\leq \frac{\varepsilon E}{1 - M} \left(\frac{1 - M^n}{1 - M} - nM^n \right) \\ &= \frac{\varepsilon E}{(1 - M)^2} \left(1 - M^n - n(1 - M)M^n\right) \\ &= \frac{\varepsilon E}{(1 - M)^2} \left(1 - (n+1)M^n + nM^{n+1}\right).\end{aligned}$$

Since $0 < M < 1$, the term in parentheses is bounded above by 1, and thus

$$\|\delta_t\| \leq \frac{\varepsilon E}{(1 - M)^2}.$$

Moreover, as $n \to \infty$, $M^n \to 0$ and $nM^n \to 0$, so

$$\lim_{T \to \infty} \|\delta_t\| \leq \frac{\varepsilon E}{(1 - M)^2}.$$

This concludes the proof.

B.2 Contractivity of the Feedback Matrix

In our experiments, unless stated otherwise, we use

$$\mathbf{A} = \mathrm{diag}(\mathrm{linspace}(0.1, 0.9, d)).$$

Since $\mathbf{A}$ is diagonal and all its entries lie in $[0.1, 0.9]$, its spectral norm is

$$\|\mathbf{A}\|_2 = \max_i |A_{ii}| = 0.9 < 1.$$

Therefore, $\mathbf{A}$ satisfies the contractivity condition used in Proposition 1. This choice also avoids the numerically loose regime induced by schedules whose maximum entry is extremely close to one.

C Ablations Regarding the Choice of A

Table 12 reports an ablation over several fixed diagonal schedules for $\mathbf{A}$ on OpenWebText with a 6-layer GRU. The results show that the choice of the feedback spectrum is important. Schedules concentrated too close to one can be unstable, while overly simple schedules such as a constant scalar matrix are weaker than using a range of diagonal values. The best fixed schedule among those tested is $\mathrm{linspace}(0.1, 0.9)$, which provides a spread of contractive timescales.

Table 12: Ablations regarding the choice of $\mathbf{A}$. We report validation perplexity of the 6-layer GRU model trained on OpenWebText. Here $\rho = 1 - 10^{-9}$.

Choice of A	**Validation perplexity** $\downarrow$
BPTT	**40.78**
$\mathrm{linspace}(0, \rho)$	45.59
$\mathrm{linspace}(-\rho, \rho)$	46.68
$\mathrm{linspace}(0.9, \rho)$	201.58
$\mathrm{logspace}(0.1, \rho)$	58.59
$0.5I$	49.14
$\mathrm{linspace}(0.1, 0.9)$	**44.60**
$\mathrm{linspace}(0.2, 0.8)$	46.22