Periodic Regulator Blocks for Attention-Free Language Modeling and Synthetic Memory Tasks

Damjan Žakelj

November 2025

Abstract

This work studies whether a small periodic modulation inserted into a transformer backbone can support non-trivial sequence processing without self-attention. We introduce a lightweight *Adaptive Frequency Block* (AFB), which applies a periodic, log-time-dependent modulation to hidden states, and a simple memory head (PVM) that accumulates information in a low-dimensional recurrent buffer.

We report two sets of experiments:

- **Phase I (positive result):** A decoder-only language model with all attention layers disabled and only two AFB blocks achieves an evaluation perplexity of approximately 47.0 on a standard text corpus after 6000 training steps, demonstrating non-trivial language modeling ability in a purely local, attention-free setting.
- **Phase II (negative result):** On a synthetic "needle-in-a-haystack" retrieval task with 512-token context and a 16-token target subsequence, the same family of models reaches at most $\approx$ 12.4% exact needle reconstruction and $\approx$ 24.4% top-5 success, despite strong internal localization signals from a diagnostic pointer head. The model learns partial memory but fails to reliably reconstruct all 16 tokens.

The main contribution of this paper is therefore twofold: (i) a concrete demonstration that periodic regulator blocks can support stable, attention-free language modeling with moderate perplexity; and (ii) a clear negative result on long-range retrieval, highlighting the gap between internal localization and actual sequence reconstruction in attention-free architectures.

1 Introduction

Self-attention is the dominant mechanism for long-range dependency modeling in modern language models. However, its quadratic cost in sequence length motivates the search for alternative mechanisms that are cheaper yet still expressive enough for both language modeling and memory-intensive tasks.

This paper investigates a very constrained setting: we disable all multi-head self-attention in a decoder-only transformer and insert a small number of *Adaptive Frequency Blocks* (AFB), which apply periodic, log-time-dependent modulation to the hidden states. In addition, we attach a compact memory head that aggregates information into a low-dimensional recurrent buffer. We then ask two concrete questions:

1. Can such a system, without any attention, learn non-trivial language statistics?
2. Can the same mechanism support explicit retrieval of a subsequence ("needle") embedded in a long context, given only local recurrence and a compact memory head?

To answer these questions we design two experimental phases:

- **Phase I – Language modeling (attention off):** We train a 33.6M-parameter decoder-only model with two AFB blocks and all self-attention disabled. The goal is to verify that the model can reach a perplexity substantially better than random or untrained baselines.
- **Phase II – Synthetic needle memory:** We train a smaller ≈ 9.5M-parameter model with one AFB block and a periodic memory head (PVM) on a synthetic needle-in-a-haystack task, with a 16-token needle embedded in a 512-token context. Here the goal is strong retrieval performance (ideally $\geq 60\%$ exact hits).

In Phase I we obtain a clear positive result: evaluation perplexity stabilizes around 47.0 without any attention. In Phase II, despite extensive tuning, diagnostic metrics, and internal pointer supervision, the model saturates at $\approx$ 12–13% exact hits and $\approx$ 2 correct tokens per needle on average, and does not cross this plateau.

The rest of the paper formalizes the architecture, describes the tasks, documents the experimental setup, and analyzes the failure modes observed in Phase II.

2 Model Architecture

2.1 Base decoder-only transformer

Our starting point is a standard decoder-only transformer at byte or subword level. Let x_t denote the token at time step t, and let $h_t \in \mathbb{R}^d$ denote the corresponding hidden state at layer input. In a conventional transformer, h_t is updated by a stack of blocks combining multi-head attention and feed-forward sublayers.

In this work, we *remove all multi-head self-attention layers* from the stack and retain only feed-forward components, augmented by the proposed periodic regulator and memory head. We keep the rest of the training pipeline—optimizer, data loader, tokenization—as close as possible to a standard implementation.

2.2 Adaptive Frequency Block (AFB)

Each AFB acts as a lightweight modulation layer applied on top of a standard feed-forward transformation. For a given time step t (with $t \geq 0$ indexing tokens within a sequence), we compute a scalar phase

$$\theta_t = \omega \log(1 + t) + \phi, \tag{1}$$

where $\omega > 0$ is a frequency parameter and ϕ is a phase offset. The AFB then applies a two-dimensional rotation to a selected pair of channels (a_t, b_t) of the hidden state:

$$a'_t = a_t \cos\theta_t - b_t \sin\theta_t, \tag{2}$$

$$b'_t = a_t \sin\theta_t + b_t \cos\theta_t. \tag{3}$$

In practice we apply this rotation channel-wise to multiple disjoint pairs of hidden dimensions, and linearly mix the result back into the residual stream. We can write the overall AFB update schematically as

$$h_t^{\text{AFB}} = h_t + \alpha\, R_{\theta_t}(h_t), \tag{4}$$

where R_{θ_t} denotes the block-diagonal rotation operator and α is a small mixing coefficient.

In Phase I we use two AFB blocks ($n_{\text{AFB}} = 2$). In Phase II, to keep the parameter count small, we use a single AFB block ($n_{\text{AFB}} = 1$).

In the most general configuration the frequency ω can be treated as learnable and updated via gradient descent within a constrained interval $[\omega_{\min}, \omega_{\max}]$. In the experiments reported here we initialize $\omega \approx 6.0$ and allow a limited adaptation range in some runs of Phase I (Section 3.1).

2.3 Periodic Vector Memory (PVM)

To give the model access to a compact form of long-range information without attention, we add a small recurrent memory vector $m_t \in \mathbb{R}^k$ (with $k \ll d$), updated at each time step as

$$m_{t+1} = (1 - \beta_t)\, m_t + \beta_t\, u_t, \tag{5}$$

where u_t is a linear projection of the hidden state h_t,

$$u_t = W_u h_t + b_u, \tag{6}$$

and $\beta_t \in (0, 1)$ is a learned or parametrized update gate. In our implementation β_t is controlled by a small set of hyperparameters $(\alpha_{\text{mem}}, \beta_{\text{mem}}, g_{\text{init}})$ and a gating nonlinearity, but conceptually it acts as an exponential moving average with a learnable time constant.

The memory m_t is fed back into the main network via an additional projection:

$$\tilde{h}_t = h_t + V m_t, \tag{7}$$

where V is a learned matrix. During training we monitor the norm of m_t to confirm that the memory is effectively used rather than collapsing to zero.

2.4 Needle pointer head (Phase II)

For Phase II we attach an additional diagnostic head that predicts the position of a "needle" subsequence within the context. Given the sequence of hidden states $\{h_t\}$ and memory states $\{m_t\}$, the pointer head produces a score s_j for each position j in the context window. The pointer head is trained with a supervised loss to assign high scores to the true needle positions.

We summarize its performance by a pointer F1-score, precision, and recall computed over the top-K candidate positions (with $K = 64$ in our experiments). These metrics allow us to distinguish between two different failure modes:

- The model cannot localize the needle at all (low pointer recall).
- The model localizes the needle internally but fails to use this information when generating the output tokens.

In the Phase II results we consistently observe the second type: pointer recall is essentially 1.0, but exact reconstruction of the 16-token needle remains low.

3 Tasks and Datasets

3.1 Phase I: Language modeling (attention off)

In Phase I we evaluate the attention-free model on a standard text corpus with subword tokenization. We use a decoder-only model with

- approximately 33,599,564 trainable parameters,
- two AFB blocks ($n_{\text{AFB}} = 2$),
- the PVM memory enabled,
- an additional spatial memory head (PLM) enabled but with very small effective norm.

The key hyperparameters for the best run (as extracted from the training metadata) are:

- learning rate $\texttt{lr} = 3.5 \times 10^{-6}$,

- sequence length `seq` $= 256$,
- effective batch size `bs` $= 2$ with gradient accumulation `ga` $= 4$,
- number of training steps `steps` $= 6000$,
- two AFB blocks with initial frequency $\omega \approx 6.0$ and limited adaptive adjustment,
- all self-attention layers disabled (`attention_disabled = true`).

We report evaluation cross-entropy and perplexity on a held-out text set.

3.2 Phase II: Synthetic needle-in-a-haystack task

3.2.1 Task definition

The needle task is a synthetic sequence-to-sequence problem designed to test long-range retrieval in a controlled setting. Each sample consists of:

- a **context** of 512 tokens, organized into 4 chunks of 128 tokens each,
- a **needle** subsequence of 16 tokens, inserted at a random position within the 512-token context,
- a **query segment** near the end of the sequence, which instructs the model to return the needle (e.g. a textual pattern such as `<SEP> QUERY: RETURN NEEDLE`),
- a **target segment** of 16 tokens, which is a verbatim copy of the embedded needle.

The total sequence length is therefore approximately 534 tokens:

$$512 \text{ (context)} + \text{query} + 16 \text{ (target)}.$$

The dataset contains on the order of 1000 training examples and 100 validation examples. All tokens are drawn from the same vocabulary as in Phase I.

3.2.2 Evaluation metrics

We compute several metrics for each validation batch:

- **Exact needle hit rate**: fraction of examples where the model correctly generates all 16 tokens of the needle in order.
- **Top-5 needle hit rate**: fraction of examples where the correct needle appears in the top-5 ranked candidate sequences (when available).
- **Average correct tokens per needle**: the average number of positions (out of 16) where the generated token matches the ground truth.
- **Pointer F1, recall, precision**: diagnostic metrics produced by the pointer head, considering the top-K candidate positions ($K = 64$).

The loss function combines the standard cross-entropy loss for the generated tokens with additional terms for the pointer head and, in some runs, an explicit retrieval loss.

3.3 Phase II: Training configuration

In Phase II we use a smaller model with approximately 9.46M trainable parameters, one AFB block, and the PVM memory enabled. Key hyperparameters for the best-performing configuration are:

- learning rate `lr` $= 1 \times 10^{-4}$,
- sequence length `seq` $= 512$ (context) plus query and target,
- batch size `bs` $= 1$ with gradient accumulation `ga` $= 4$,
- number of training steps `steps` $= 2000$ in the main runs, with shorter runs for ablations,
- one AFB block ($n_{\mathrm{AFB}} = 1$) with fixed $\omega \approx 6.0$,
- PVM memory enabled with $(\alpha_{\mathrm{mem}}, \beta_{\mathrm{mem}})$ chosen such that the memory norm remains in a moderate range,
- self-attention disabled (`attention_disabled = true`).

We additionally perform a series of "micro" runs with modified memory settings to investigate sensitivity to the memory decay and update rate.

4 Results

4.1 Phase I: Attention-free language modeling

Table 1 summarizes the main metrics for the best attention-free language model run. All numbers are taken from the training metadata associated with the final checkpoint.

Metric	Value
Evaluation cross-entropy (CE)	3.85
Evaluation perplexity (PPL)	47.02
Training steps	6000
Trainable parameters	33,599,564
AFB blocks	2
Self-attention	disabled

Table 1: Phase I – attention-free language modeling with two Adaptive Frequency Blocks (AFB) and PVM enabled.

Perplexity 47.0 is significantly worse than what a full attention-based model of similar size would achieve on the same corpus, but it is orders of magnitude better than the trivial baseline (where perplexity would be effectively at the vocabulary size). The evaluation loss remains stable over the last training epochs and does not exhibit gradient explosions or numerical instability.

We interpret this as a **positive result**: even with all self-attention removed, the combination of local feed-forward transformations and periodic modulation in AFB blocks is sufficient to learn a non-trivial language model. This establishes that the periodic regulator is a viable building block for attention-free sequence modeling, at least in the moderate-perplexity regime.

4.2 Phase II: Synthetic needle memory

4.2.1 Baseline and partial memory runs

Initial baseline runs with the needle task and attention disabled yield near-zero retrieval performance. As expected, without any explicit memory head or pointer mechanism, the model behaves essentially as a local auto-regressive predictor and fails to reproduce the 16-token needle.

After enabling the PVM memory and the pointer head and tuning the corresponding hyperparameters, we observe a clear but limited improvement. A representative run achieves:

- `needle_hit_rate` ≈ 0.105 (10.5% exact hits),
- `needle_hit_rate_topk` ≈ 0.186 (18.6% top-5 hits),
- `avg_correct_tokens_per_needle` ≈ 1.68,
- pointer F1-score ≈ 0.40, pointer recall ≈ 1.0, pointer precision ≈ 0.25.

These numbers indicate that the model has learned a weak but real notion of long-range memory: roughly two positions of the needle are correct on average, and a non-trivial fraction of examples achieve perfect reconstruction.

4.2.2 Best needle run

The best-performing configuration in our experiments (with optimized memory parameters and readout settings) reaches:

- `needle_hit_rate` ≈ 0.12375 (12.4% exact hits),
- `needle_hit_rate_topk` ≈ 0.24375 (24.4% top-5 hits),
- `avg_correct_tokens_per_needle` ≈ 1.98,
- pointer F1-score ≈ 0.40 with recall ≈ 1.0 and precision ≈ 0.25.

We summarize these results in Table 2.

Metric	Value
Exact needle hit rate	0.12375
Top-5 needle hit rate	0.24375
Average correct tokens per needle	1.98
Pointer F1-score (diagnostic)	0.40
Pointer recall (top-64)	1.0
Pointer precision (top-64)	0.25
Context length	512
Needle length	16
Trainable parameters	$\approx$ 9.46M
Self-attention	disabled

Table 2: Phase II – best synthetic needle memory run with one AFB block and PVM enabled.

The exact hit rate around 12–13% is clearly above random baseline and is reproducible across seeds and minor hyperparameter variations, but it remains far below the target range ($\geq 60\%$) that would indicate a robust memory mechanism.

4.2.3 Joint training with pointer supervision

We also experimented with a "joint" training regime, where the model is trained simultaneously on language modeling, pointer localization, and explicit retrieval loss terms. In this configuration the pointer head continues to perform well (F1 ≈ 0.4, recall ≈ 1.0), but the generation of the needle tokens does not improve. In some joint runs, the perplexity on the needle segment remains extremely high and the exact hit rate stays at 0%, despite clear gradients reaching the pointer and retrieval heads.

This suggests that the main bottleneck is not the ability to localize the needle within the context, but the ability to *use* that localization information to produce the correct sequence of tokens at the output.

5 Analysis and Failure Modes

5.1 Plateau around 12–18% exact hits

A central empirical observation is that, once the PVM memory and pointer mechanisms are properly tuned, the model's performance on the needle task stabilizes in a narrow band around 10–13% exact hits and ≈ 2 correct tokens per needle on average. Further tuning of hyperparameters (α, γ) in the AFB, PVM update rates, learning rate, or loss weights produces only small fluctuations within this band and does not produce a clear breakthrough to significantly higher retrieval accuracy.

There are several reasons to believe that this plateau reflects a structural limitation rather than suboptimal training:

- Runs with different seeds consistently converge to the same range of metrics.
- Larger or smaller values of the main hyperparameters degrade performance, but do not produce outlier runs with, for example, 30–40% exact hits.
- The pointer head exhibits strong recall from early in training, indicating that the core network can detect the needle location, yet the generation head fails to align with this signal.

5.2 Local recurrence vs. random access

The needle task is, by design, a random-access retrieval problem: the model must reconstruct a 16-token subsequence placed at an arbitrary position in a 512-token context. Standard self-attention is well suited to this type of problem, as it can compare each query position with all positions in the context and directly attend to the most relevant ones.

The architecture studied in this paper, by contrast, is essentially a locally recurrent system with a low-dimensional memory. Information about earlier tokens must be propagated forward through many steps of the AFB and PVM dynamics. While this is sufficient to carry coarse or aggregated information (such as averages or global statistics), it is much harder to maintain the exact identity and ordering of a 16-token subsequence over hundreds of steps.

The pointer diagnostics confirm this picture:

- The model learns a strong internal localization signal: pointer recall is ≈ 1.0 at $K = 64$, meaning the needle positions are consistently among the top-64 candidates.
- The precision remains low (≈ 0.25), and the main language modeling head does not reliably reconstruct the actual token sequence even when the pointer is correct.

In other words, the architecture learns "*where*" the needle is, but does not fully learn "*what*" it is in token space.

5.3 Bottleneck at the readout interface

Another important factor is the interface between the memory/pointer subsystem and the main language modeling head. In the current implementation, the PVM state and pointer scores influence the logits through relatively small projections and scaling factors. If these projections are too weak, the model may learn to rely primarily on local context and ignore the memory signals, which would explain why improvements in pointer quality do not translate into higher needle hit rates.

Indeed, some of the diagnostics indicate that:

- The norm of the PVM memory remains in a moderate range (so the memory is active).
- Scaling up the pointer signal at evaluation time can artificially increase the influence of the pointer, but this does not always lead to better exact reconstruction.

This suggests that a more powerful or more tightly integrated interface between the memory and the token-level logits may be necessary.

5.4 Model size and capacity limitations

Finally, the model used in Phase II has only ≈ 9.46M trainable parameters and a single AFB block. It is plausible that the model simply lacks sufficient capacity to jointly learn:

- a robust language representation,
- a compact memory encoding for the needle subsequence,
- a reliable mapping from the memory and pointer signals back to the output token sequence.

Preliminary tests with larger models and more AFB blocks suggest that perplexity on language tasks can indeed be reduced, but systematic experiments on the needle task with larger configurations remain for future work.

6 Discussion

The experiments reported here illustrate a clear separation between two abilities:

1. **Stable attention-free language modeling:** Phase I shows that periodic regulator blocks (AFB) can support non-trivial language modeling without self-attention, with evaluation perplexity around 47.0 on a standard text corpus. The model is small and far from state-of-the-art, but the result is stable and reproducible.
2. **Exact long-range retrieval:** Phase II shows that, under the current design, the same family of architectures struggles to perform exact retrieval on a synthetic needle-in-a-haystack task, saturating around 12–13% exact hits despite strong internal localization signals.

These findings are consistent with the intuition that local recurrence plus a low-dimensional memory can approximate *some* aspects of long-range dependency (e.g. summary statistics or coarse information), but that exact reconstruction of arbitrarily positioned subsequences is a qualitatively harder problem.

Importantly, the Phase II results should be seen as a **negative but useful** outcome:

- They provide a concrete quantitative limit for the current design.

- They highlight specific bottlenecks (readout interface, model capacity, objective design) that can be targeted in future work.
- They separate the notion of "internal localization" (where the pointer head performs well) from "usable memory for generation" (where the main language head still fails).

7 Limitations and Future Work

This study has several limitations:

- **Model size:** Both Phase I and Phase II use relatively small models (tens of millions of parameters), which may not be sufficient to fully explore the capabilities of the architecture.
- **Task diversity:** The synthetic needle task is intentionally challenging and may not be representative of all realistic memory-intensive tasks. Intermediate tasks (e.g. shorter needles, structured queries, or repeated patterns) could provide a more nuanced view of the memory capabilities.
- **Interface design:** The current interface between the memory and the logits is relatively simple. More expressive readout mechanisms or explicitly gated mixture-of-experts structures might be needed to fully exploit the internal localization signals.
- **Training objectives:** Joint training of language modeling, pointer localization, and retrieval may require more careful balancing or curriculum strategies to avoid interference between tasks.

Future work should therefore explore:

1. Scaling the model to larger sizes and more AFB blocks while keeping attention disabled.
2. Systematic ablations on the memory interface, including stronger readout gates and direct copying mechanisms.
3. Variants of the needle task with different needle lengths and context structures, to map out the scaling behavior of retrieval accuracy.
4. Hybrid architectures that combine a small amount of attention with periodic regulator blocks and memory heads, to better understand the trade-offs between complexity and retrieval performance.

8 Conclusion

We have presented an experimental study of periodic regulator blocks and compact memory heads as building blocks for attention-free sequence modeling. In the language modeling regime (Phase I), the architecture achieves stable, non-trivial perplexity without any self-attention, demonstrating that periodic modulation can serve as a viable alternative backbone. In the synthetic needle retrieval regime (Phase II), the same architecture learns partial memory and strong internal localization signals, but fails to reach high exact retrieval rates, plateauing around 12–13%.

These results draw a clear line between what this class of attention-free models can currently do and where they fall short. They provide both a positive proof-of-concept for attention-free language modeling and a well-defined negative result for long-range retrieval, which can serve as a benchmark for future improvements.