A SPECTRAL PERSPECTIVE ON CONTINUOUS DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

A spectral Fourier perspective provides a natural lens for understanding visual Diffusion Models. This view inspires a broad range of insights and practical methods. Unlike vision, the text domain does not naturally admit a spectral interpretation due to its discrete nature. However, the rise of Continuous Diffusion Language Models (DLMs), which perform diffusion in the representation space of pretrained text encoders, makes the spectral perspective more natural to investigate in the text domain. In this work, we study continuous DLMs through a Fourier lens. We show that the representation spaces of widely used text models, such as Qwen3.8, Gemma 4, and Nemotron-3, exhibit an approximate power-law spectral structure across sequence length, similar to that observed in images. Notably, this phenomenon persists across different model sizes, architectural families, data domains, training specializations, and instruction-tuning settings. We then analyze the information carried by different frequency bands and uncover a key distinction from vision: high-frequency components in text retain substantially more energy and encode important token-level, positional information, rather than merely fine-grained details. Motivated by these observations, we introduce a frequency-aware diffusion strategy that processes low- and high-frequency components separately across diffusion timesteps, enabling dedicated modeling of high frequencies. Across multiple tasks and representation spaces, this approach yields consistent improvements, with the gains tied to better preservation of positional relationships between tokens.

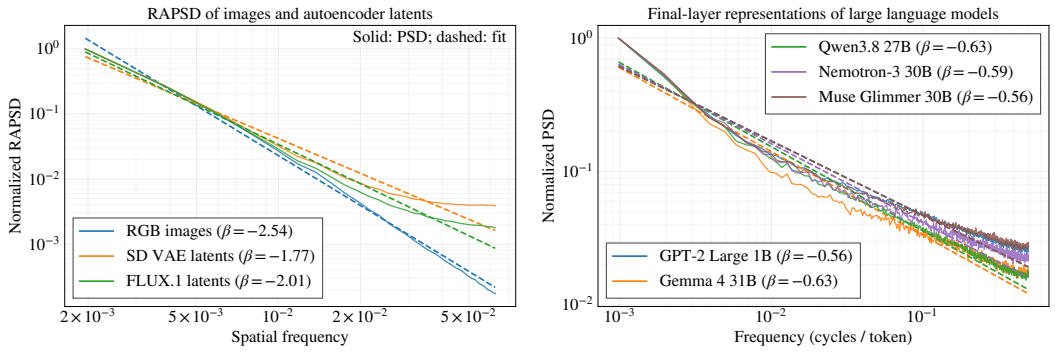
1 INTRODUCTION

An insightful framework for analyzing visual Diffusion Models (DMs) (Ho et al., 2020; Song & Ermon, 2019; Lipman et al., 2023) is through a *spectral perspective* (Dieleman, 2024; Falck et al., 2025): the Fourier power spectrum of natural images approximately follows a power law, with most of the signal energy concentrated at low spatial frequencies (van der Schaaf & van Hateren, 1996; Torralba & Oliva, 2003; Simoncelli & Olshausen, 2001). This implies the existence of a frequency hierarchy: high-frequency components are destroyed first during the noising process, followed by lower-frequency components, resulting in a coarse-to-fine reconstruction process during generation (Rissanen et al., 2023; Hoogeboom & Salimans, 2023; Phung et al., 2023). Beyond vision, similar power-law spectral structure also appears in audio and speech, motivating frequency-aware generative modeling in these domains (Khatami et al., 2018; Koizumi et al., 2022).

This perspective gives rise to a wide range of methods that improve different aspects of DMs, including generation quality (Pavasovic et al., 2026; Qian et al., 2024), latent space diffusability (Skorokhodov et al., 2025; Ning et al., 2026), and sampling speed (Starodubcev et al., 2026; Phung et al., 2023; Yang et al., 2023; Han et al., 2026). Notably, it applies across different image representations, including raw pixels (Li & He, 2026; Ma et al., 2026) and continuous latent spaces (Skorokhodov et al., 2025).

However, it remains unclear whether the *spectral perspective* is also applicable to Diffusion Language Models (DLMs). For discrete DLMs (Hoogeboom et al., 2021; Austin et al., 2021; Campbell et al., 2022; Gat et al., 2024), it is not obvious how to investigate such a perspective, since the discrete diffusion space does not admit a natural notion of frequency.

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.



(a) (b)

Figure 1: **Power spectra of image and text representation spaces.** (a) Radially averaged power spectral density of RGB images and latent representations from image autoencoders (Skorokhodov et al., 2025). (b) Power spectral density of final-layer representations from different text models. Solid lines show the empirical spectra and dashed lines their power-law fits. While both image and text representations exhibit an approximate power-law structure, text representations show substantially slower spectral decay, indicating relatively higher energy at high frequencies.

The rapid rise of DLMs brings ideas from vision DMs into the text domain. For example, a promising direction is to train continuous DLMs in the representation space of a pretrained text encoder. This direction is inspired by latent DMs in vision (Rombach et al., 2022; Esser et al., 2024; Kingma & Welling, 2013; Zheng et al., 2026), where an encoder maps an image into a compact continuous representation (Oquab et al., 2023), and the DM is then trained to model the distribution of these latent codes. For text, instead of applying diffusion directly to discrete tokens, these models first map text into continuous representations and then perform the diffusion process in the resulting continuous space (Hu et al., 2026; 2024; Li et al., 2022; Dieleman et al., 2022; Chen et al., 2026).

So, the move toward continuous representations makes it natural to ask whether the *spectral perspective* also extends to latent DLMs.

Contributions. In this paper, we analyze state-of-the-art continuous DLMs through a Fourier lens. We first show that the representation spaces of widely used text models exhibit an approximate power-law spectral structure across sequence length, as shown in Figure 1. Importantly, this phenomenon appears to be general across model sizes, architectural families, and data domains. We then characterize the information carried by different frequency components and identify a key distinction from images. In text, high-frequency components carry relatively more energy and preserve more useful information, whereas in images they primarily capture imperceptible details.

Based on these findings, we design a diffusion formulation that assigns separate denoising trajectories to low- and high-frequency components, allowing their distinct roles to be modeled explicitly. We show that it improves performance in most settings across different tasks and text representation spaces. Interestingly, these gains appear to stem from improved modeling of high-frequency components, leading to better preservation of positional relationships between tokens. This demonstrates the practical value of the *spectral perspective* for designing continuous DLMs.

To our knowledge, our findings establish the first *spectral perspective* on continuous DLMs operating in text encoder representation spaces. We believe that it opens the door to a broad range of frequency-aware approaches for improving the design and training of continuous DLMs.

2 Related Work

DLMs. DLMs (He et al., 2022; Zheng et al., 2023; Zhou et al., 2023; Lou et al., 2024; Sahoo et al., 2024; Shi et al., 2024; Ou et al., 2024) generate text by iteratively reversing a corruption process, enabling parallel updates across multiple tokens rather than imposing a fixed left-to-right factorization. Existing methods differ primarily in the space in which denoising is performed. Discrete DLMs operate directly on categorical token states, including early formulations such as D3PM (Austin et al.,

2021) and SEDD (Lou et al., 2024), masked models such as MDLM (Sahoo et al., 2024), and larger-scale models such as LLaDA (Nie et al., 2025) and Dream (Ye et al., 2025).

Continuous DLMs (Shabalin et al., 2025; Gao et al., 2024; Gulrajani & Hashimoto, 2023; Lin et al., 2023; Gong et al., 2023; Strudel et al., 2022) can be further distinguished by the continuous representation on which generation is performed. *Token-embedding models* replace each token with a dense vector from an embedding table and directly corrupt the resulting sequence of embeddings. This category includes Diffusion-LM (Li et al., 2022), SED (Strudel et al., 2022), CDCD (Dieleman et al., 2022), DiffuSeq (Gong et al., 2023), FlowSeq (Hu et al., 2024), and LangFlow (Chen et al., 2026). Other methods operate on continuous relaxations of the vocabulary space, FLM (Lee et al., 2026) evolves vocabulary-sized one-hot representations rather than dense text embeddings.

Encoder-based models, by contrast, first process the complete token sequence with a text encoder and then perform diffusion over its contextual hidden states. TEncDM (Shabalin et al., 2025) and ELF (Hu et al., 2026) operate on representations from pretrained language encoders, whereas LD4LG (Lovelace et al., 2023), LDLM (Meshchaninov et al., 2026), and CoDAR (Shen et al., 2026), and Cola DLM (Guo et al., 2026) additionally employ explicit latent encoders or decoders to mediate between continuous representations and discrete text. Our work belongs specifically to the encoder-based setting: following ELF, we apply diffusion to contextual representations produced by a pretrained text encoder and study the spectral structure of this representation space.

Spectral analysis of language. In vision, the spectral perspective is well established (Dieleman, 2024), whereas its application to language remains comparatively underexplored. Most prior work focuses on discriminative language models. Language Through Prism (Tamkin et al., 2020) applies spectral filtering to BERT representations and shows that different frequency bands capture linguistic information at different scales, ranging from local syntactic properties to document-level semantics. Funnel-Transformer (Dai et al., 2020) progressively compresses hidden sequences to remove redundancy and reduce computation, while Fourier Transformer (He et al., 2023) uses the DCT to identify and discard redundant components for efficient long-range modeling. FourierMoE (Jiang et al., 2026) applies Fourier parameterization to model adaptation, constructing frequency-specialized experts for parameter-efficient fine-tuning. Our work extends the spectral perspective beyond discriminative models and model efficiency to the broader setting of LLMs, studying spectral structure as a general property of the continuous representation spaces.

Most closely related is FourierSampler (He et al., 2026), which studies the spectral structure of hidden representations in discrete DLMs such as LLaDA (Nie et al., 2025). In contrast, we analyze the continuous training space produced by a text encoder, rather than the internal features of the DLM itself.

3 Spectral Analysis of LLM Representations

The continuous DLMs considered in this work map sequences of discrete tokens into sequences of continuous vectors using a pretrained text model, which we refer to as a *text encoder*.[1] These DLMs typically apply diffusion to the final-layer hidden states produced by this model. Accordingly, throughout this work, the *representation space* of a text encoder refers to the space of its final-layer hidden states, which serves as the diffusion space for training the DLM.

In this section, we first analyze the spectral properties of these representation spaces independently of the diffusion model. To examine whether these properties generalize across model families and scales, we consider smaller models—T5-small (Raffel et al., 2020), GPT-2 Base and GPT-2 Large (Radford et al., 2019), and Qwen3.5 0.8B (Qwen Team, 2026a)—as well as substantially larger models—Qwen3.8 27B (Qwen Team, 2026b), Nemotron 3 Nano 30B-A3B (NVIDIA, 2025), Muse Glimmer 30B (Meta Superintelligence Lab, 2026), and Gemma 4 31B (Gemma Team, 2026).

[1] We use the term *text encoder* functionally rather than architecturally: it denotes any text model used to encode an input sequence into continuous representations for DLM training, including models originally described as decoder-only models or LLMs.

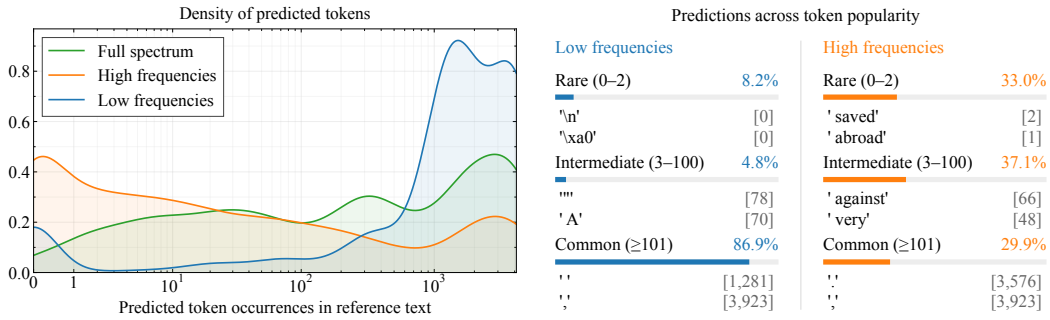


Figure 2: **Connection between spectral frequency bands and token frequency.** (a) Distribution of predicted tokens as a function of their occurrence frequency in the reference text corpus for the full spectrum, low-frequency components, and high-frequency components. (b) Examples of predictions across rare, intermediate, and common token groups. Low-frequency components strongly favor common tokens, whereas high-frequency components produce a substantially broader distribution that includes both common and rare tokens.

3.1 Text Encoder Representation Spaces Exhibit a Power Law

We compute the power spectrum for different text encoders on the natural text dataset (Penedo et al., 2024) by applying a discrete Fourier transform along the sequence dimension independently to each hidden dimension and averaging the squared coefficient magnitudes across dimensions. More details are provided in App. A.

We present the results for the larger models in Figure 1. Interestingly, we find that the representation spaces of all considered text encoders exhibit a clear frequency hierarchy: low-frequency components contain more energy, while the energy gradually decreases toward higher frequencies. In other words, the spectra of text encoders approximately follow a power law.

Importantly, the power-law exponent β is remarkably similar across different text encoders. Moreover, it remains systematically higher than that observed for images and autoencoder image latents. This indicates that text representations preserve substantially more relative energy at high frequencies compared to image representations. This suggests that high-frequency components may play a more important role in text representations than they do in images.

The observed power-law structure may help explain why DMs are effective in these representation spaces, as the noising process of DMs naturally induces a similar frequency hierarchy: high-frequency components are corrupted at low noise levels, while lower-frequency components are progressively destroyed as the noise level increases (Dieleman, 2024).

We provide more analyses in App. B. Notably, we find that the power-law structure is preserved across models of different sizes and architectural families. Moreover, this phenomenon persists across different data domains, domain-specific training, and instruction tuning. Interestingly, the power-law structure emerges already in relatively early layers and is then progressively refined, with deeper layers mainly steepening the spectral slope. Overall, these results suggest that the observed spectral structure is a robust property of text representation spaces rather than a specificity of a particular model, dataset, or training setup.

3.2 Properties of Frequency Bands

Connection to Token Distribution. A natural question is what different frequencies actually encode. For images, this question is relatively well understood: low frequencies primarily capture global structure and coarse semantic content, while high frequencies encode finer details, local variations, and textures. We next investigate whether a similar separation of information emerges in the embedding space of text models.

Configuration	Last layer				Average across layers			
	ListOps	Sentiment	Retrieval	Topic	ListOps	Sentiment	Retrieval	Topic
Random Classifier	10.0	50.0	50.0	5.0	10.0	50.0	50.0	5.0
Full	28.8 0.1	89.4 0.3	55.3 0.3	56.0 0.3	19.8 0.3	83.4 0.0	51.7 0.3	35.2 0.1
High Frequencies	32.4 0.8	78.2 0.5	50.5 0.9	30.0 0.1	**32.4** 0.2	73.5 0.0	49.9 0.6	17.2 0.1
Low Frequencies	32.4 0.8	**84.3** 0.2	**54.7** 1.0	**56.5** 0.4	31.5 0.2	**80.4** 0.1	**52.3** 0.3	**38.9** 0.1

Table 1: **Classification performance using different frequency components of Qwen representations.** We report results from the last layer and averaged across all layers on ListOps, Sentiment, Retrieval, and Topic classification tasks. Low-frequency representations outperform high-frequency representations on most tasks, while high-frequency components remain competitive.

The simplest intuition is to relate frequency components to the token frequency distribution. For example, one may expect common tokens to be associated with lower frequencies, while rarer tokens are represented more strongly by higher-frequency components.

To analyze this, we consider Qwen3.5-0.8B on the natural text dataset (Gokaslan & Cohen, 2019). Given a context sequence, we first perform next-token prediction and obtain the corresponding final-layer representation. We then apply the real FFT along the sequence dimension, retain either the low- or high-frequency band, and reconstruct the filtered representations using the inverse real FFT. The resulting frequency-filtered state is then passed through the Qwen decoder head, which maps it to logits over the vocabulary. In this way, we can analyze which tokens are associated with different frequency bands. More details and other models are in App. C.

After generating the sequences, we compute the empirical frequency of the predicted tokens and compare it with their occurrence frequency in the reference dataset. The results in Figure 2(a) show a clear distinction between high- and low-frequency components. When Qwen uses only the high-frequency components of its representations, the prediction distribution shifts toward tokens that are rarer in the reference corpus. In contrast, when only low-frequency components are retained, the model tends to generate tokens that occur frequently in the text.

Interestingly, the distribution of tokens generated using high-frequency components is more uniform than that obtained using low-frequency components. As shown in Figure 2(b), high-frequency representations still generate common tokens with substantial probability, while low-frequency representations produce a much more concentrated distribution dominated by a small set of frequent tokens. So, this suggests that high-frequencies retain a broader range of lexical information.

Low Frequencies. If low-frequency components primarily encode common tokens, one may hypothesize that they also capture more global information about the text. To test this, we consider a set of tasks in which global information is particularly important for achieving strong performance.

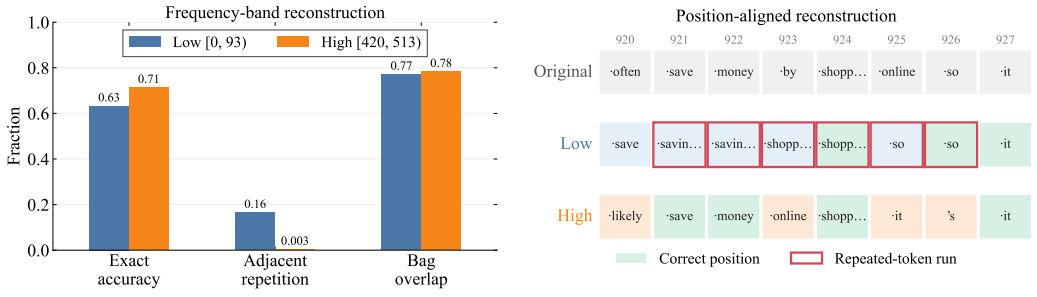
We consider four benchmarks: *Topic*, 20-way topic classification on 20 Newsgroups (Lang, 1995); *Text*, binary sentiment classification on IMDb (Maas et al., 2011); *ListOps*, prediction of the result of a nested expression (Nangia & Bowman, 2018; Tay et al., 2021); and *Retrieval*, classification of whether two documents from 20 Newsgroups belong to the same topic (Lang, 1995).

For each frozen Qwen3.5 0.8B layer, we retain either the full spectrum or equally sized low- and high-frequency bands along the token dimension. After inverse transformation, we train a linear classifier on the final reconstructed token. More details and other models are in App. C.

The results in Table 1 show that low-frequency components statistically outperform high-frequency components on three out of four benchmarks. This suggests that low frequencies primarily encode global, sequence-level information. However, high-frequency components remain competitive, suggesting that they retain useful information rather than merely representing noise.

Tamkin et al. (2020) report a similar observation for BERT representations. Our work extends this analysis to a broader range of language models.

High Frequencies. In the same spirit, we aim to better understand the properties of high-frequency components. We consider a reconstruction task in which we train a Transformer decoder to re-



(a) (b)

Figure 3: **Reconstruction from low- and high-frequency Qwen3.5 0.8B representations.** (a) High frequencies yield higher position-aligned reconstruction accuracy, while both bands achieve similar position-agnostic bag-of-tokens overlap. (b) Low frequencies often recover the correct token content but exhibit more repetitions and positional errors.

construct the original token sequence from either the high- or low-frequency components of the final-layer representations of Qwen3.5 0.8B. More details are in App. C.

We present the results in Figure 3(a). In this setting, the high-frequency band achieves better reconstruction performance than the low-frequency band (71% vs. 63%). Interestingly, this improvement mainly comes from a better recovery of token positions. When we ignore token positions and measure only bag overlap, the two bands perform almost identically (78% vs. 77%). Thus, both bands preserve comparable information about which tokens occur in the sequence, while the advantage of high frequencies primarily lies in recovering their correct positions.

We show a reconstruction example in Figure 3(b). It illustrates that low-frequency components often recover the correct words, but tend to produce adjacent repetitions and place them less accurately within the sequence.

To better understand this effect, we measure local cosine similarity between token representations obtained by applying the inverse Fourier transform to either low- or high-frequency coefficients. We present the results for different group sizes in Figure 12. We observe that low-frequency token representations are highly similar across nearby positions, making individual positions difficult to distinguish and limiting the encoding of position-specific information.

To sum up, low frequencies often know what tokens should occur, but struggle to place them. While high frequencies retain more position-specific information and are therefore better for exact text reconstruction.

To probe this more directly, we consider a more targeted task: password-order classification. Four distinct marker tokens are placed at fixed adjacent positions within a random context, and the objective is to identify their ordering among all possible permutations. We train linear probes on the low- and high-frequency representations extracted from each Qwen layer and present the results in Figure 13. The results show a consistent advantage of high-frequency representations across layers.

Key findings

1. General spectral structure. Power-law spectral structure emerges as a general property of text representation spaces, persisting across model scales, architectural families, data domains, training specializations, and instruction tuning.

2. Informative high frequencies. High-frequency components of text representations retain substantially richer information than their image-domain counterparts: like low-frequency components, they encode which tokens appear in a sequence, while additionally capturing fine-grained positional relationships between them.

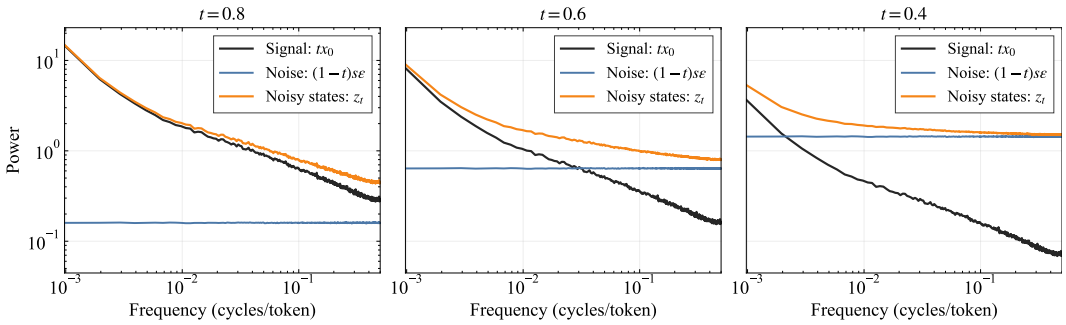


Figure 4: **Power spectra of noisy Qwen3.5 0.8B representations.** We show the power spectra of the signal, noise, and their mixture at different flow-matching timesteps, where $t = 1$ corresponds to clean data and $t = 0$ to noise. At noisier timesteps, only low-frequency components remain distinguishable from noise, while higher frequencies emerge progressively during denoising.

Snippet 1. Haar Delay, Training

```
# Split data and noise into Haar projections
x_lf, x_hf = haar(x)
e_lf, e_hf = haar(randn_like(x))

# Part 1: LF objective (coarse)
t = sample_t(0, 1)
z_lf = t * x_lf + (1 - t) * e_lf
z_hf = e_hf
x_hat = net(z_lf + z_hf, t)
xhat_lf, _ = haar(x_hat)
loss_coarse = l2_loss(xhat_lf - x_lf)

# Part 2: LF + HF objective (coarse+fine)
t = sample_t(t*, 1)
u = htime(t, t*)
z_lf = t * x_lf + (1 - t) * e_lf
z_hf = u * x_hf + (1 - u) * e_hf
x_hat = net(z_lf + z_hf, t)
xhat_lf, xhat_hf = haar(x_hat)
loss_fine = l2_loss(xhat_lf - x_lf)
          + l2_loss(xhat_hf - x_hf)
```

Snippet 2. Haar Delay, Inference

```
# Initialize
z_lf, z_hf = haar(randn())

# Phase 1: LF update
for t, t_next in steps(0, t*):
  x_hat = net(z_lf + z_hf, t)
  xhat_lf, _ = haar(x_hat)
  v_lf = (xhat_lf - z_lf) / (1 - t)
  z_lf += (t_next - t) * v_lf

# Phase 2: LF + HF refinement
for t, t_next in steps(t*, 1):
  u = htime(t, t*)
  u_next = htime(t_next, t*)
  x_hat = net(z_lf + z_hf, t)
  xhat_lf, xhat_hf = haar(x_hat)
  v_lf = (xhat_lf - z_lf) / (1 - t)
  v_hf = (xhat_hf - z_hf) / (1 - u)
  z_lf += (t_next - t) * v_lf
  z_hf += (u_next - u) * v_hf

return z_lf + z_hf
```

4 Exploiting Spectral Structure in DLMs

Motivation. Our analysis demonstrates that the representation spaces used to train continuous DLMs exhibit a spectral structure, with high-frequency components playing a particularly important role. This motivates the next question: how can these properties be explicitly exploited to improve the performance of continuous text DMs?

We first investigate which diffusion timesteps are most important for modeling high-frequency information. To this end, we apply the Fourier transform to noisy versions of the Qwen3.5 0.8B representation states. We then analyze how the spectral energy of different frequency components evolves across diffusion timesteps.

The results in Figure 4 reveal behavior similar to that observed in images. At noisy timesteps ($t = 0.4$), only the lowest-frequency components remain distinguishable from noise, while higher frequencies are largely suppressed. As denoising progresses, increasingly higher frequencies emerge above the noise level, with high-frequency information becoming distinguishable only during the later stages of generation. This suggests that modeling all frequencies uniformly across timesteps may be suboptimal and motivates emphasizing high-frequency components at later diffusion steps.

Haar Delay. We introduce *Haar Delay*, a simple frequency-aware training and sampling strategy inspired by WaiT (Pavasovic et al., 2026). Haar Delay applies a one-level Haar transform along the sequence dimension, decomposing text representations into low-frequency and high-frequency components. The Haar transform is an orthogonal and lossless multiresolution transform: for each

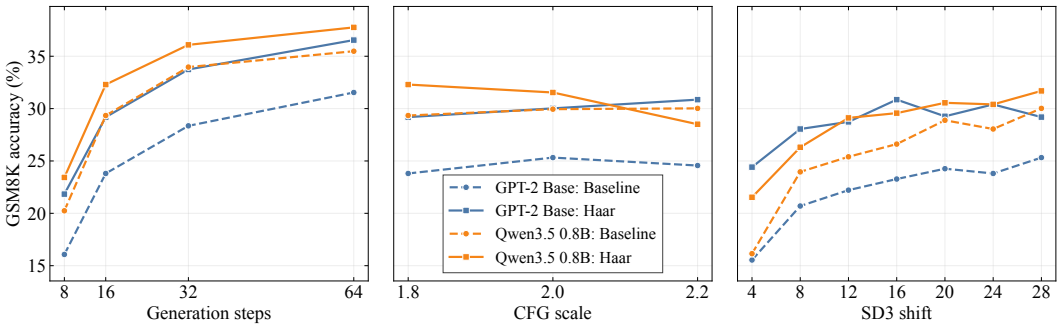
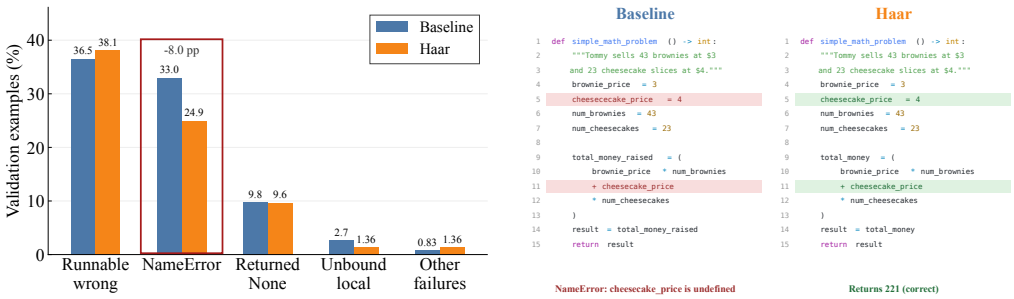


Figure 5: **GSM8K accuracy of ELF-B across sampling configurations.** We compare the baseline and Haar Delay while varying the number of generation steps, CFG scale, and SD3 sampling shift.



(a) (b)

Figure 6: **Error analysis on GSM8K for ELF-B.** (a) Errors for the baseline and Haar Delay. The main gain comes from reducing NameError cases. (b) Example of a NameError, where the baseline loses track of a previously introduced variable, while Haar Delay produces a consistent solution.

pair of neighboring tokens, it represents their shared local content through an average component and their local variation through a difference component. We use this decomposition to assign separate diffusion schedules to LF and HF information, delaying the denoising of high-frequency components. The complete training and inference procedures are shown in Snippets 1 and 2.

We introduce a hyperparameter `t*` that separates the diffusion trajectory into coarse and fine stages. Before `t*`, the model operates only on LF components. After `t*`, HF components are introduced and both frequency bands are refined jointly. We define the local HF time as

$$u(t) = \texttt{htime(t, t*)} = \max\left(0, \frac{t - \texttt{t*}}{1 - \texttt{t*}}\right).$$

This reparameterization is necessary because the LF and HF components follow separate denoising trajectories. When HF components are introduced at `t*`, they are still at their initial noise state, whereas LF components have already progressed to time `t*`. The function `htime` therefore maps the remaining interval $[\texttt{t*}, 1]$ onto the full HF trajectory $[0, 1]$, allowing HF components to catch up with LF components and reach the clean state simultaneously at $t = 1$. Ablations are in App. D.

Training. As shown in Snippet 1, training consists of two objectives. The coarse objective is applied over the full diffusion trajectory and reconstructs only the LF component. The fine objective is applied for $t > \texttt{t*}$ and jointly reconstructs LF and HF components. We use a simple, unweighted sum of the two reconstruction losses. This encourages the model to first learn the coarse sequence structure and then recover the local information encoded by high frequencies.

Inference. Inference follows the same coarse-to-fine process, as shown in Snippet 2. We initialize both Haar components from Gaussian noise. During the first stage, only LF is denoised, while HF remains fixed. During the second stage, LF and HF are updated jointly using the times t and $u(t)$, respectively. The method adds no parameters and uses only inexpensive Haar transforms.

	4 generation steps				32 generation steps				
	Reasoning			Text-to-SQL		Reasoning			Text-to-SQL
	L0-Trace EM	xLAM EM	Sudoku Solved	WikiSQL EM		L0-Trace EM	xLAM EM	Sudoku Solved	WikiSQL EM
GPT-2					**GPT-2**				
Baseline	96.3	**63.1**	98.0	47.7	Baseline	96.3	63.9	98.6	48.4
Haar	**97.1**	62.7	**98.6**	**49.2**	Haar	**96.9**	**65.0**	**98.8**	**48.6**
T5					**T5**				
Baseline	–	–	95.3	**50.4**	Baseline	–	–	96.9	49.6
Haar	–	–	**96.7**	50.0	Haar	–	–	**97.9**	**50.2**
Qwen3.5					**Qwen3.5**				
Baseline	93.2	28.1	98.8	**44.5**	Baseline	93.4	31.8	**99.4**	**45.5**
Haar	**95.7**	**29.1**	**99.2**	44.1	Haar	**95.9**	**36.5**	**99.4**	45.3

Table 2: **Results across benchmarks at 4 and 32 generation steps.** Haar Delay matches or improves upon the tuned baseline in 7 of 10 settings at 4 steps and 9 of 10 settings at 32 steps.

5 Experiments

We train ELF-B/M models (Hu et al., 2026) in representation spaces provided by GPT-2 Base, T5, and Qwen3.5-0.8B. The original ELF model was trained only with T5, for this setting, we closely follow the original configuration. For GPT-2 and Qwen3.5, we first identify the best-performing baseline configuration and then apply Haar Delay to this setting. Details are in App. D.

GSM8k. First, we consider conditional generation on GSM8K (Cobbe et al., 2021). Given a natural-language arithmetic word problem as context, the model generates an executable Python function that solves the problem. We measure accuracy by executing the generated code and comparing its output with the ground-truth answer. We provide ablations in App. D.

In Figure 5, we compare the baseline and Haar Delay for ELF-B across different numbers of sampling steps, CFG scales, and sampling-schedule shifts. We observe improvements across configurations, except at a CFG scale of 2.2. For ELF-M, gains are substantial across all settings (Figure 14).

A central question is what drives these improvements. In Figure 6(a), we analyze different types of errors and find that the boost mainly comes from reducing NameError cases. This type of error occurs when the model loses track of previously initialized variables, referring to them using incorrect names, as in Figure 6(b). Haar Delay reduces these errors, improving variable-name consistency.

Analysis in 3 links high-frequency components to fine-grained positional information. We therefore hypothesize that explicitly separating frequency bands encourages the model to better process high-frequency information and, consequently, to maintain more consistent positional dependencies. The reduction in NameError cases is consistent with this interpretation.

Other benchmarks. Next, we evaluate Haar Delay on a broader set of conditional-generation tasks: L0-Trace (Sun et al., 2025), xLAM (Liu et al., 2024), Sudoku (Rao, 2019), and WikiSQL (Zhong et al., 2017). Details of the training and evaluation are in App. D. Selected hyperparameters are in Table 7.

Table 2 reports ELF-B results at 4 and 32 generation steps. Haar Delay matches or outperforms the tuned baseline in 7 of 10 settings at 4 steps and 9 of 10 settings at 32 steps.

6 Conclusion

We introduce a spectral perspective on continuous DLMs and show that the representation spaces of diverse text encoders exhibit an approximate power-law structure, resembling natural images. Interestingly, text representations retain substantially more high-frequency energy, which preserves both token information and fine-grained positional relationships. Motivated by these findings, we propose Haar Delay, which separates low- and high-frequency trajectories and introduces high-frequency information later in generation. Across multiple encoders, model scales, and tasks, this simple modification improves performance in most setups. More broadly, our results establish the spectral perspective as a useful framework for analyzing and improving continuous DLMs.

AI USE STATEMENT

Generative AI assistants were used to support literature search, conducting the evaluation, result collection, language editing and formatting, and figure preparation. The authors take responsibility for the final content of the paper.

REFERENCES

Nasir Ahmed, T. Natarajan, and Kamisetty R Rao. Discrete cosine transform. *IEEE transactions on Computers*, C-23(1):90–93, 1974. doi: 10.1109/T-C.1974.223784.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2021. URL https://arxiv.org/abs/2107.03006.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models, 2022. URL https://arxiv.org/abs/2205.14987.

Yuxin Chen, Chumeng Liang, Hangke Sui, Ruihan Guo, Chaoran Cheng, Jiaxuan You, and Ge Liu. LangFlow: Continuous diffusion rivals discrete in language modeling, 2026. URL https://arxiv.org/abs/2604.11748.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Zihang Dai, Guokun Lai, Yiming Yang, and Quoc V. Le. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. In *Advances in Neural Information Processing Systems*, volume 33, pp. 4271–4282, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/2cd2915e69546904e4e5d4a2ac9e1652-Abstract.html.

Sander Dieleman. Diffusion is spectral autoregression, 2024. URL https://sander.ai/2024/09/02/spectral-autoregression.html. Blog post.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*, 2022.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Fabian Falck, Teodora Pandeva, Kiarash Zahirnia, Rachel Lawrence, Richard E. Turner, Edward Meeds, Javier Zazo, and Sushrut Karmalkar. A fourier space perspective on diffusion models, 2025. URL https://arxiv.org/abs/2505.11278.

Zhujin Gao, Junliang Guo, Xu Tan, Yongxin Zhu, Fang Zhang, Jiang Bian, and Linli Xu. Empowering diffusion models on the embedding space for text generation. In *Proceedings of NAACL*, 2024. URL https://arxiv.org/abs/2212.09412.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching, 2024. URL https://arxiv.org/abs/2407.15595.

Gemma Team. Gemma 4 Technical Report, 2026. URL https://arxiv.org/abs/2607.02770.

Gemma Team et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024. URL https://arxiv.org/abs/2408.00118.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. DiffuSeq: Sequence to sequence text generation with diffusion models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.08933.

Ishaan Gulrajani and Tatsunori B. Hashimoto. Likelihood-based diffusion language models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.18619.

Hongcan Guo, Qinyu Zhao, Yian Zhao, Shen Nie, Rui Zhu, Qiushan Guo, Feng Wang, Tao Yang, Hengshuang Zhao, Guoqiang Wei, and Yan Zeng. Continuous latent diffusion language model, 2026. URL https://arxiv.org/abs/2605.06548.

Jiaqi Han, Juntong Shi, Puheng Li, Haotian Ye, Qiushan Guo, and Stefano Ermon. Adaptive spectral feature forecasting for diffusion sampling acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43320–43330, 2026. URL https://arxiv.org/abs/2603.01623.

Siyang He, Qiqi Wang, Xiaoran Liu, Hongnan Ma, Yiwei Shi, Yuerong Song, Ying Zhu, Tianyi Liang, Zengfeng Huang, Ziwei He, and Xipeng Qiu. FourierSampler: Unlocking Non-Autoregressive Potential in Diffusion Language Models via Frequency-Guided Generation. *arXiv preprint arXiv:2601.23182*, 2026. URL https://arxiv.org/abs/2601.23182.

Zhengfu He, Tianxiang Sun, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. DiffusionBERT: Improving generative masked language models with diffusion models, 2022. URL https://arxiv.org/abs/2211.15029.

Ziwei He, Meng Yang, Minwei Feng, Jingcheng Yin, Xinbing Wang, Jingwen Leng, and Zhouhan Lin. Fourier transformer: Fast long range modeling by removing sequence redundancy with FFT operator. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8954–8966. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-acl.570. URL https://aclanthology.org/2023.findings-acl.570/.

Jonathan Ho, Ajay N. Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Emiel Hoogeboom and Tim Salimans. Blurring diffusion models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OjDkC57x5sz.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions, 2021. URL https://arxiv.org/abs/2102.05379.

Keya Hu, Linlu Qiu, Yiyang Lu, Hanhong Zhao, Tianhong Li, Yoon Kim, Jacob Andreas, and Kaiming He. Elf: Embedded language flows. *arXiv preprint arXiv:2605.10938*, 2026.

Vincent Hu, Di Wu, Yuki Asano, Pascal Mettes, Basura Fernando, Björn Ommer, and Cees Snoek. Flow matching for conditional text generation in a few sampling steps. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 380–392, 2024.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, et al. Qwen2.5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Albert Q. Jiang et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. URL https://arxiv.org/abs/2310.06825.

Juyong Jiang, Fan Wang, Hong Qi, Sunghun Kim, and Jing Tang. FourierMoE: Fourier mixture-of-experts adaptation of large language models, 2026. URL https://arxiv.org/abs/2604.01762.

Fatemeh Khatami, Markus Wöhr, Heather L. Read, and Monty A. Escabí. Origins of scale invariance in vocalization sequences and speech. *PLoS Computational Biology*, 14(4):e1005996, 2018. doi: 10.1371/journal.pcbi.1005996.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. The stack: 3 tb of permissively licensed source code. *arXiv preprint arXiv:2211.15533*, 2022.

Yuma Koizumi, Heiga Zen, Kohei Yatabe, Nanxin Chen, and Michiel Bacchiani. Specgrad: Diffusion probabilistic model based neural vocoder with adaptive noise spectral shaping. In *Interspeech*, pp. 803–807, 2022. doi: 10.21437/Interspeech.2022-301.

Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 331–339. Morgan Kaufmann, 1995. doi: 10.1016/B978-1-55860-377-6.50048-7.

Chanhyuk Lee, Jaehoon Yoo, Manan Agarwal, Sheel Shah, Jerry Huang, Aditi Raghunathan, Seunghoon Hong, Nicholas M Boffi, and Jinwoo Kim. Flow map language models: One-step language modeling via continuous denoising. *arXiv preprint arXiv:2602.16813*, 2026.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36115–36125, 2026.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-LM Improves Controllable Text Generation. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2205.14217.

Zhenghao Lin, Yeyun Gong, Yelong Shen, Tong Wu, Zhihao Fan, Chen Lin, Nan Duan, and Weizhu Chen. Text generation with diffusion language models: A pre-training approach with continuous paragraph denoise. In *International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2212.11685.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Zuxin Liu, Thai Hoang, Jianguo Zhang, Ming Zhu, Tian Lan, Shirley Kokane, Juntao Tan, Weiran Yao, Zhiwei Liu, Yihao Feng, Rithesh Murthy, Liangwei Yang, Silvio Savarese, Juan Carlos Niebles, Huan Wang, Shelby Heinecke, and Caiming Xiong. APIGen: Automated pipeline for generating verifiable and diverse function-calling datasets. In *Advances in Neural Information Processing Systems*, 2024.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution, 2024. URL https://arxiv.org/abs/2310.16834.

Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Shekhtman, and Kilian Q. Weinberger. Latent Diffusion for Language Generation. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2212.09462.

Zehong Ma, Longhui Wei, Shuai Wang, Shiliang Zhang, and Qi Tian. Deco: Frequency-decoupled pixel diffusion for end-to-end image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43600–43610, 2026.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150. Association for Computational Linguistics, 2011. URL https://aclanthology.org/P11-1015/.

Viacheslav Meshchaninov, Alexander Shabalin, Egor Chimbulatov, Nikita Gushchin, Ilya Koziev, Alexander Korotin, and Dmitry Vetrov. How to train your latent diffusion language model jointly with the latent space. *arXiv preprint arXiv:2605.07933*, 2026.

Meta Superintelligence Lab. Muse-Glimmer-30B. Hugging Face model repository, 2026. URL https://huggingface.co/meta-models/Muse-Glimmer-30B.

Nikita Nangia and Samuel R. Bowman. ListOps: A diagnostic dataset for latent tree learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 92–99. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-4013. URL https://aclanthology.org/N18-4013/.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large Language Diffusion Models. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2502.09992.

Mang Ning, Mingxiao Li, Le Zhang, Lanmiao Liu, Matthew B. Blaschko, Albert Ali Salah, and Itir Onal Ertugrul. Spectrum matching: A unified perspective for superior diffusability in latent diffusion, 2026. URL https://arxiv.org/abs/2603.14645.

NVIDIA. Nemotron 3 Nano: Open, efficient mixture-of-experts hybrid Mamba-Transformer model for Agentic reasoning, 2025. URL https://arxiv.org/abs/2512.20848. Technical report.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data, 2024. URL https://arxiv.org/abs/2406.03736.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. Openwebmath: An open dataset of high-quality mathematical web text. In *International Conference on Learning Representations*, 2024.

Krunoslav Lehman Pavasovic, Théophane Vallaeys, Stéphane Mallat, Giulio Biroli, Luke Zettlemoyer, Brian Karrer, and Jakob Verbeek. WaiT for the Signal: Simple Frequency-Aware Flow-Matching. *arXiv preprint arXiv:2607.28760*, 2026. URL https://arxiv.org/abs/2607.28760.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=n6SCkn2QaG.

Hao Phung, Quan Dao, and Anh Tran. Wavelet diffusion models are fast and scalable image generators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10199–10208, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Phung_Wavelet_Diffusion_Models_Are_Fast_and_Scalable_Image_Generators_CVPR_2023_paper.html.

Yurui Qian, Qi Cai, Yingwei Pan, Yehao Li, Ting Yao, Qibin Sun, and Tao Mei. Boosting diffusion models with moving average sampling in frequency domain. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8911–8920, 2024. URL https://arxiv.org/abs/2403.17870.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Rohan Rao. 9 million sudoku puzzles and solutions. Kaggle, 2019. URL https://www.kaggle.com/datasets/rohanrao/sudoku.

Severi Rissanen, Markus Heinonen, and Arno Solin. Generative modelling with inverse heat dissipation. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=4PJUBT9f2Ol.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models, 2024. URL https://arxiv.org/abs/2406.07524.

Alexander Shabalin, Viacheslav Meshchaninov, Egor Chimbulatov, Vladislav Lapikov, Roman Kim, Grigory Bartosh, Dmitry Molchanov, Sergey Markov, and Dmitry Vetrov. TEncDM: Understanding the Properties of the Diffusion Model in the Space of Language Model Encodings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25110–25118, 2025. doi: 10.1609/aaai.v39i23.34696. URL https://ojs.aaai.org/index.php/AAAI/article/view/34696.

Junzhe Shen, Jieru Zhao, Ziwei He, and Zhouhan Lin. CoDAR: Continuous diffusion language models are more powerful than you think, 2026. URL https://arxiv.org/abs/2603.02547.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data, 2024. URL https://arxiv.org/abs/2406.04329.

Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, 24(1):1193–1216, 2001.

Ivan Skorokhodov, Sharath Girish, Benran Hu, Willi Menapace, Yanyu Li, Rameen Abdal, Sergey Tulyakov, and Aliaksandr Siarohin. Improving the diffusability of autoencoders, 2025. URL https://arxiv.org/abs/2502.14831.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Nikita Starodubcev, Ilya Drobyshevskiy, Denis Kuznedelev, Artem Babenko, and Dmitry Baranchuk. Scale-wise Distillation of Diffusion Models. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Z06LNjqU1g.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, and Rémi Leblond. Self-conditioned embedding diffusion for text generation, 2022. URL https://arxiv.org/abs/2211.04236.

Simeng Sun, Cheng-Ping Hsieh, Faisal Ladhak, Erik Arakelyan, Santiago Akle Serano, and Boris Ginsburg. L0-reasoning bench: Evaluating procedural correctness in language models via simple program execution. *arXiv preprint arXiv:2503.22832*, 2025.

Alex Tamkin, Dan Jurafsky, and Noah Goodman. Language through a prism: A spectral approach for multiscale language representations. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3acb2a202ae4bea8840224e6fce16fd0-Abstract.html.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qVyeW-grC2k.

Antonio Torralba and Aude Oliva. Statistics of natural image categories. *Network: computation in neural systems*, 14(3):391–412, 2003.

A. van der Schaaf and J. H. van Hateren. Modelling the power spectra of natural images: statistics and information. *Vision research*, 36(17):2759–2770, 1996.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, et al. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22552–22562, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Yang_Diffusion_Probabilistic_Model_Made_Slim_CVPR_2023_paper.html.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7B: Diffusion Large Language Models. *arXiv preprint arXiv:2508.15487*, 2025. URL https://arxiv.org/abs/2508.15487.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=0u1LigJaab.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation, 2023. URL https://arxiv.org/abs/2302.05737.

Victor Zhong, Caiming Xiong, and Richard Socher. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017.

Kun Zhou, Yifan Li, Wayne Xin Zhao, and Ji-Rong Wen. Diffusion-NAT: Self-prompting discrete diffusion for non-autoregressive text generation, 2023. URL https://arxiv.org/abs/2305.04044.

A Background Details

DLM Training. The continuous DLMs considered in this paper (Hu et al., 2026) map a token sequence $\mathbf{s} = [s_1, \ldots, s_L]$ into continuous representations

$$\mathbf{x} = \mathbf{E}(\mathbf{s}) = [\mathbf{e}_1, \ldots, \mathbf{e}_L], \qquad \mathbf{e}_i \in \mathbb{R}^d, \tag{1}$$

where d is the text-encoder dimension. Following ELF (Hu et al., 2026), we train a Flow Matching model directly in this representation space. Training timesteps follow a logit-normal distribution, and noisy states are constructed by linear interpolation:

$$\begin{aligned} &\eta \sim \mathcal{N}(\mu_{\text{train}}, \sigma^2_{\text{train}}), \qquad t = \text{sigmoid}(\eta), \\ &\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad\qquad\quad \mathbf{z}_t = t\mathbf{x} + (1-t)\boldsymbol{\epsilon}. \end{aligned} \tag{2}$$

Thus, $t = 0$ corresponds to noise and $t = 1$ to clean representations. ELF uses an x-prediction parameterization, $\hat{\mathbf{x}}_\theta = f_\theta(\mathbf{z}_t, t)$, with

$$\mathbf{v} = \mathbf{x} - \boldsymbol{\epsilon}, \qquad \hat{\mathbf{v}}_\theta = \frac{\hat{\mathbf{x}}_\theta - \mathbf{z}_t}{1-t}. \tag{3}$$

The denoising objective is

$$\begin{aligned} \mathcal{L}_{\text{FM}} &= \mathbb{E}\left[\|\hat{\mathbf{v}}_\theta - \mathbf{v}\|_2^2\right] \\ &= \mathbb{E}\left[\frac{1}{(1-t)^2}\|\hat{\mathbf{x}}_\theta - \mathbf{x}\|_2^2\right]. \end{aligned} \tag{4}$$

At inference, the uniform N-step grid $t_i = i/N$ is transformed using the SD3 schedule shift:

$$\widetilde{t}_i = \frac{t_i}{s_{\text{infer}} - (s_{\text{infer}} - 1)t_i}, \qquad i = 0, \ldots, N, \tag{5}$$

where $s_{\text{infer}} = 1$ recovers the uniform schedule.

ELF also trains the same network to decode continuous representations into tokens. Corrupted clean representations $\tilde{\mathbf{z}}$ are processed in a separate *decode* mode at $t = 1$ and projected to vocabulary logits:

$$\mathcal{L}_{\text{CE}} = \mathbb{E}\left[\text{CE}\left(\mathbf{W} f_\theta(\tilde{\mathbf{z}}, 1), \mathbf{s}\right)\right]. \tag{6}$$

The denoising and decoding branches share all model parameters and are sampled with probabilities 0.8 and 0.2, respectively.

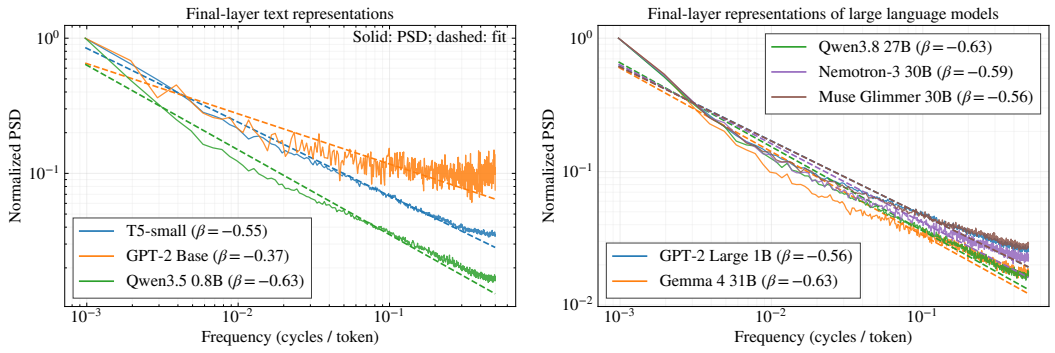
Discrete DCT. We apply spectral analysis to this continuous representation space $\mathbf{E}(\mathbf{s})$. Specifically, we use the discrete cosine transform (DCT) (Ahmed et al., 1974) along the sequence dimension, treating each hidden dimension as a one-dimensional signal. We apply the DCT to each hidden dimension independently and then average the resulting spectra across dimensions.

More formally, let $\mathbf{E}^j = [e_1^j, \ldots, e_L^j] \in \mathbb{R}^L$ denote the one-dimensional signal corresponding to the j-th hidden dimension, where $j \in \{1, \ldots, d\}$. The discrete Fourier transform (DFT), computed using the fast Fourier transform (FFT), represents this signal as $\mathbf{D}^j = [D_0^j, \ldots, D_{L-1}^j] \in \mathbb{R}^L$, where

$$D_f^j = \sum_{k=1}^{L} e_k^j \exp\left(-\frac{2\pi i}{L}(k-1)f\right), \qquad f = 0, \ldots, L-1. \tag{7}$$

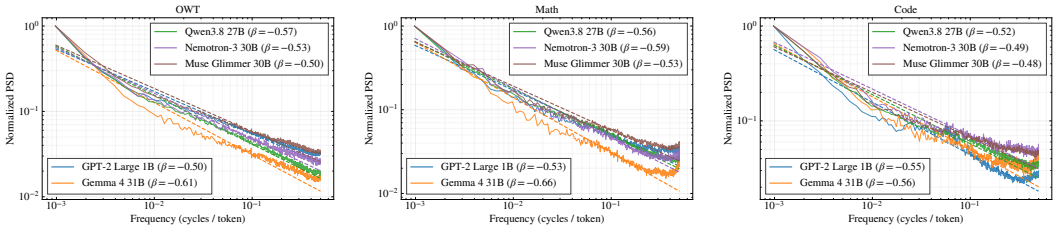
Here, f indexes the frequency components, with $f = 0$ corresponding to the zero-frequency (DC) component and larger values of f corresponding to higher frequencies. The final power spectrum $\mathbf{P} \in \mathbb{R}^L$ is obtained by averaging the power of $\mathbf{D}^j$ across all hidden dimensions:

$$P(f) = \frac{1}{d}\sum_{j=1}^{d}\left|D_f^j\right|^2. \tag{8}$$



(a) (b)

Figure 7: **Power spectra of representation spaces across text models of different scales.** (a) Final-layer representation spectra for smaller models, including T5-small, GPT-2 Base, and Qwen3.5 0.8B. (b) Final-layer representation spectra for larger models, including GPT-2 Large 1B, Qwen3.8 27B, Nemotron-3 30B, Muse Glimmer 30B, and Gemma 4 31B. Solid lines show the empirical power spectra, while dashed lines denote power-law fits. Across model families and scales, the spectra exhibit a similar approximately power-law decay.



(a) (b) (c)

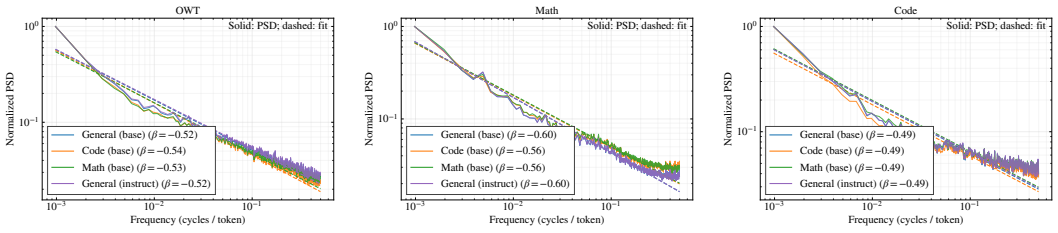
Figure 8: **Power spectra of text-model representation spaces across different data domains.** We report the normalized PSD of final-layer representations for large text models on (a) OWT, (b) Math, and (c) Code datasets. Across domains, the representation spaces exhibit a similar approximate power-law decay, suggesting that the observed spectral structure is largely preserved across different types of text.

B Additional Spectral Analysis

We compute $\mathbf{P}$ in Eq. equation 8 from the final-layer hidden representations of each text encoder using the FineWeb corpus (Penedo et al., 2024). The same documents are tokenized separately by each model and packed into 32 sequences of length 1,024 without padding. For each sequence, we compute the spectral power independently for every hidden channel, average the power across channels, and then aggregate the spectra across sequences using a geometric mean. We omit the zero-frequency component and normalize each spectrum by its maximum nonzero-frequency power.

The resulting spectra for smaller and larger models are presented in Figure 7. Across model families and scales, we observe a remarkably similar, approximately power-law spectral decay. Notably, this behavior persists across substantially different model architectures, including encoder- and decoder-based models, as well as models with different positional encoding schemes. This consistency suggests that the observed spectral structure is not tied to a particular architecture, but may instead reflect a more general property of text representations.

In addition, in Figure 8, we verify that the observed power-law structure persists across datasets of substantially different nature, including general-domain text from OpenWebText (OWT) (Gokaslan & Cohen, 2019), mathematical text from OpenWebMath (Paster et al., 2024), and code (Kocetkov et al., 2022). Interestingly, for Math and Code, the spectra exhibit a mild plateau at high frequencies.



(a) (b) (c)

Figure 9: **Effect of text-encoder training data on spectral structure.** We compare architecture-matched Qwen2.5-1.5B checkpoints trained or adapted for different domains: a general base model, code- and mathematics-specialized base models, and a general instruction-tuned model. We report the normalized PSD of their final-layer representations on (a) OWT, (b) Math, and (c) Code data. Despite differences in pretraining specialization and post-training, all variants exhibit similar approximately power-law spectra, indicating that the spectral structure is robust to both domain-specific training and instruction tuning.

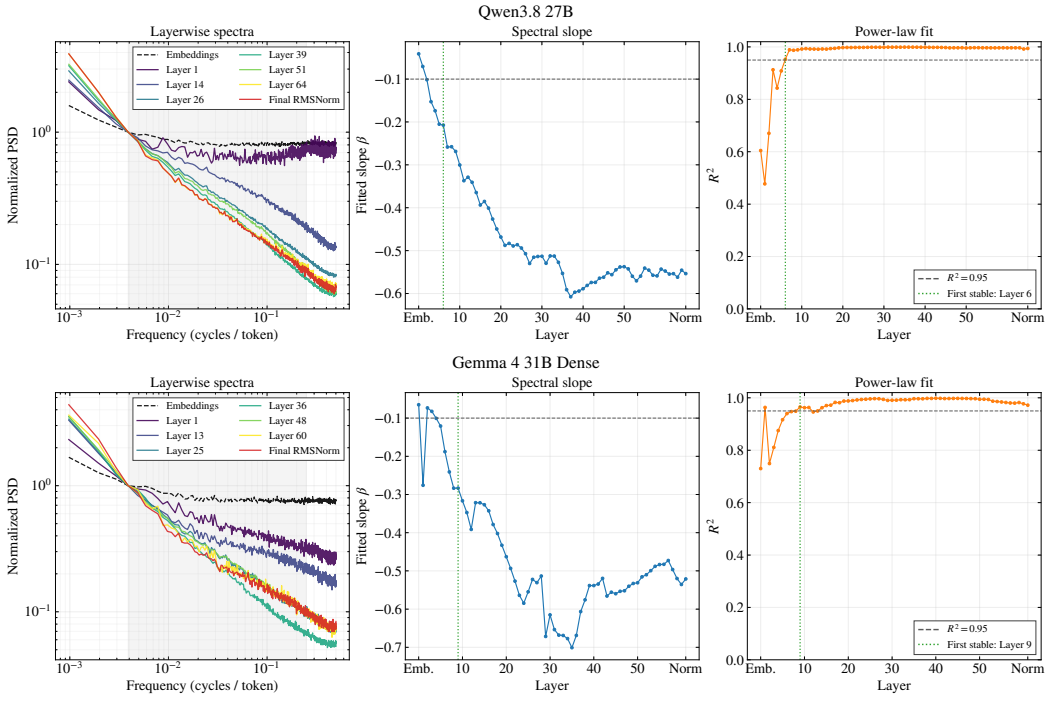


Figure 10: **Layerwise emergence of power-law spectral structure in Qwen3.8 and Gemma 4.** The input embeddings do not exhibit a clear power law. The structure emerges in the early transformer layers—at Layer 6 for Qwen3.8 and Layer 9 for Gemma 4—and remains stable thereafter. Subsequent layers primarily adjust the spectral slope while preserving a strong power-law fit.

Next, we consider the effect of the data used to train the text encoder. We compare four architecture-matched Qwen2.5-1.5B checkpoints: the general base model, code- (Hui et al., 2024) and mathematics- (Yang et al., 2024) specialized base models, and the general instruction-tuned model (Qwen et al., 2025). As shown in Figure 9, all variants exhibit similar power-law spectra despite their different training specializations and post-training procedures. These results indicate that the observed spectral structure is robust to both domain-specific training and instruction tuning.

Finally, we consider the PSD across different layers of Qwen3.8 and Gemma 4 in Figure 10. First, the input embeddings do not yet exhibit a clear power-law spectrum. Second, the power-law structure forms relatively early: at Layer 6 for Qwen3.8 and Layer 9 for Gemma 4. This is confirmed by

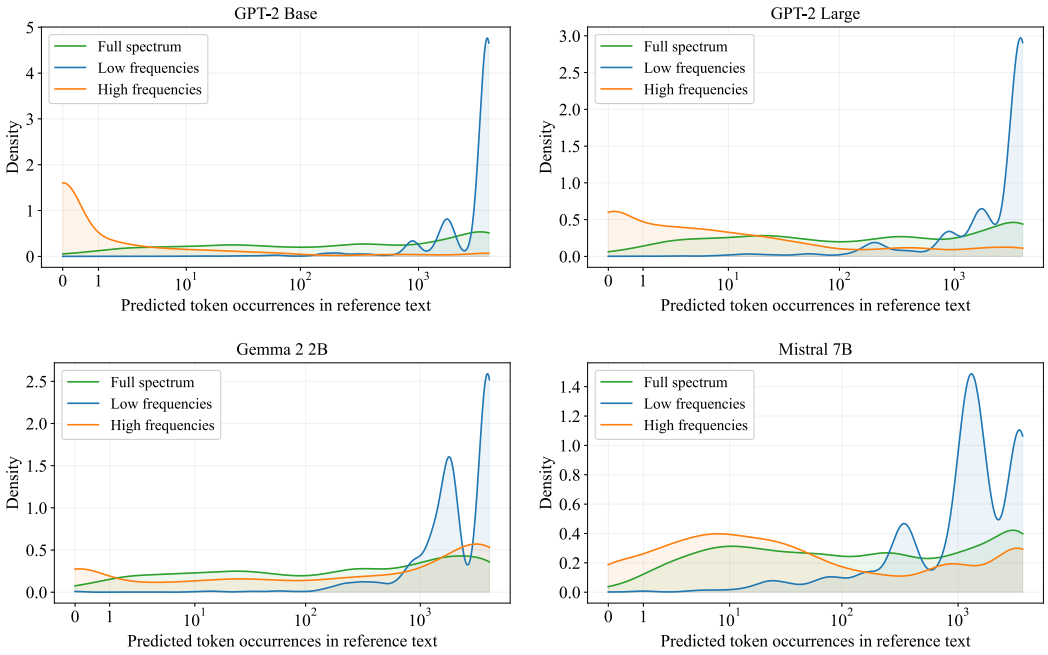


Figure 11: **Distribution of predicted-token frequencies across text models.** We compare predictions obtained using the full spectrum and the isolated low- and high-frequency components of GPT-2 Base, GPT-2 Large, Gemma 2 2B, and Mistral 7B representations. Across all models, low-frequency components strongly favor common tokens, whereas high-frequency components shift the predictions toward rarer tokens and produce a broader lexical distribution.

the high R^2 values, which indicate a strong log–log linear fit from these layers onward. Finally, the later layers mainly refine the spectral decay, making the slope more negative while preserving the overall power-law shape.

C Properties of Frequency Bands

Connection to Token Distribution. We use Qwen3.5-0.8B-Base and OpenWebText (Gokaslan & Cohen, 2019). We tokenize and concatenate documents, inserting an EOS token between consecutive documents. We construct 100 non-overlapping examples without padding. Each example contains an initial context of 1,024 tokens followed by 32 target tokens, resulting in 3,200 next-token predictions.

For each target position, we run Qwen causally and extract the final-layer hidden states of the available prefix. Thus, every prediction uses only the tokens preceding its target, and no future information is included. We apply the real FFT along the sequence dimension independently for each hidden channel. We retain the same number of coefficients for the low- and high-frequency bands and set all remaining coefficients to zero. At a sequence length of 1,024, the low-frequency band contains coefficients $[0, 93)$, including the DC component, while the high-frequency band contains coefficients $[420, 513)$. We then apply the inverse FFT to reconstruct the frequency-filtered hidden-state sequence.

Because frequency filtering changes the magnitude of the representations, we match the RMS norm of the final filtered state to that of the original final-layer state. The resulting vector is passed directly through Qwen's pretrained language-model head, without additional training, to obtain logits over the vocabulary. We compare predictions from the full spectrum, low-frequency band, and high-frequency band.

To measure token popularity, we count the occurrences of every token ID in the same packed OpenWebText sample used for evaluation. Each predicted token is assigned its corresponding reference

	Last layer				Average across layers			
Configuration	ListOps	Sentiment	Retrieval	Topic	ListOps	Sentiment	Retrieval	Topic
Random Classifier	10.0	50.0	50.0	5.0	10.0	50.0	50.0	5.0
Full	23.1 1.6	93.0 0.2	53.7 0.5	61.6 0.2	16.9 0.3	89.4 0.1	55.1 0.1	48.8 0.1
High Frequencies	31.6 0.6	87.8 0.1	50.6 0.3	33.3 0.3	**31.8 0.0**	81.2 0.1	50.9 0.1	22.2 0.1
Low Frequencies	**33.5 0.4**	**91.1 0.3**	**52.8 0.6**	**62.8 0.3**	31.4 0.1	**87.1 0.1**	**54.2 0.1**	**50.8 0.0**

Table 3: **Spectral probing results for Mistral 7B.**

	Last layer				Average across layers			
Configuration	ListOps	Sentiment	Retrieval	Topic	ListOps	Sentiment	Retrieval	Topic
Random Classifier	10.0	50.0	50.0	5.0	10.0	50.0	50.0	5.0
Full	19.2 0.5	84.4 0.4	51.4 0.4	47.6 0.2	15.1 0.1	81.1 0.1	51.0 0.1	30.0 0.1
High Frequencies	30.9 0.9	73.9 0.4	50.4 0.8	27.4 0.3	**31.8 0.3**	67.2 0.0	50.4 0.6	15.0 0.0
Low Frequencies	**31.6 0.2**	**82.6 0.2**	**51.8 0.6**	**49.2 0.8**	31.3 0.1	**77.9 0.1**	**50.9 0.3**	**34.1 0.0**

Table 4: **Spectral probing results for Gemma 2 2B.**

	Last layer				Average across layers			
Configuration	ListOps	Sentiment	Retrieval	Topic	ListOps	Sentiment	Retrieval	Topic
Random Classifier	10.0	50.0	50.0	5.0	10.0	50.0	50.0	5.0
Full	30.1 1.2	92.0 0.1	52.1 0.5	54.2 0.2	17.8 0.3	84.5 0.1	51.0 0.2	38.9 0.2
High Frequencies	31.8 0.8	83.8 0.1	50.6 0.4	26.9 0.5	32.7 0.7	72.7 0.3	50.6 0.5	16.5 0.1
Low Frequencies	**34.6 0.9**	**88.6 0.4**	**51.1 0.4**	**58.4 0.4**	**33.5 0.1**	**81.7 0.2**	**51.3 0.1**	**43.4 0.0**

Table 5: **Spectral probing results for GPT-2 Large.**

	Last layer				Average across layers			
Configuration	ListOps	Sentiment	Retrieval	Topic	ListOps	Sentiment	Retrieval	Topic
Random Classifier	10.0	50.0	50.0	5.0	10.0	50.0	50.0	5.0
Full	26.3 0.3	86.6 0.1	51.7 0.4	51.9 0.1	15.6 0.3	78.8 0.2	50.2 0.4	31.8 0.1
High Frequencies	32.8 0.1	73.8 0.0	49.4 1.0	25.7 0.2	32.3 0.4	66.1 0.3	49.7 0.2	14.6 0.2
Low Frequencies	**33.6 0.7**	**84.1 0.4**	**50.9 0.7**	**54.6 0.1**	**33.4 0.1**	**76.4 0.1**	**50.7 0.5**	**36.5 0.1**

Table 6: **Spectral probing results for GPT-2 Base.**

count. We then plot the density of the log-transformed reference counts over all 3,200 predictions. Small values correspond to tokens that are rare or absent in the reference text, whereas large values correspond to common tokens.

In Figure 11, we consider more text encoders, including GPT-2 Base and GPT-2 Large (Radford et al., 2019), Mistral 7B (Jiang et al., 2023), and Gemma 2 2B (Gemma Team et al., 2024). We consistently observe the same pattern across model families and scales: low-frequency components concentrate predictions on a small set of frequent tokens, whereas high-frequency components produce a broader distribution shifted toward rarer tokens. In particular, the low-frequency bands produce only 55–149 unique tokens, compared with 963–1,888 for the high-frequency bands. Moreover, almost none of the low-frequency predictions correspond to rare tokens, while their proportion reaches 18–74% for high frequencies. These results confirm that the observed frequency separation is not specific to Qwen model.

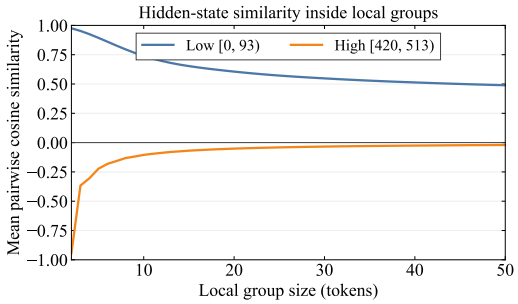


Figure 12: **Local similarity of low- and high-frequency Qwen representations.** Mean pairwise cosine similarity between token representations within local groups of different sizes. Low-frequency representations remain more similar across nearby positions, while high-frequency representations are more position-specific.

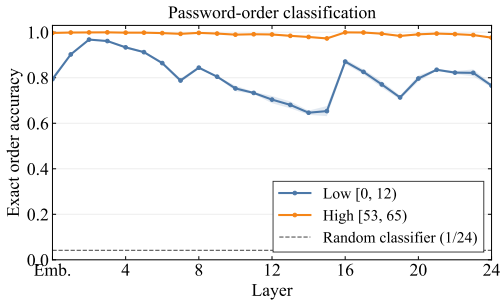


Figure 13: **Exact token-order prediction from low- and high-frequency Qwen representations.** High-frequency components consistently encode token order more accurately across layers, achieving near-perfect performance, while low-frequency components retain substantially less position-specific information.

Low Frequencies. For each frozen Qwen3.5 0.8B layer, including the embedding layer, we apply an orthonormal real-valued Fourier transform along the unpadded token dimension. We retain either the full spectrum, the lowest-frequency band, or an equally sized highest-frequency band, corresponding to approximately 18% of all frequency components. The remaining coefficients are set to zero, and the sequence is reconstructed using the inverse Fourier transform. We use the final valid token vector as the sequence representation.

For every layer and frequency condition, we train an independent linear classifier on standardized features while keeping the language model frozen. We use the same training configuration in all cases and report results averaged over three seeds. These benchmarks serve as an analysis tool for identifying where task-relevant information is concentrated in the spectrum, rather than as a method for improving task performance.

We present results for additional models in Tables 3, 4, 5, and 6. We observe the same pattern across model families and scales: low-frequency components consistently perform better, while high-frequency components also achieve strong results, indicating that they contain meaningful task-relevant information rather than merely noise.

High Frequencies. We use the normalized final-layer representations of Qwen3.5 0.8B for packed OpenWebText sequences of 1,024 tokens. We apply the real-valued Fourier transform along the token dimension and retain either the first 93 low-frequency bins or the last 93 high-frequency bins out of 513 total bins. All remaining coefficients are set to zero before applying the inverse transform. For each band, we fine-tune a separate Transformer decoder initialized from the pretrained ELF-B checkpoint. The decoder receives the filtered representations at $t = 1$ and is trained with same-position cross-entropy over all 1,024 target tokens. We train for 10,000 updates with a global batch size of 16 and report token accuracy on a fixed validation set.

We find that low-frequency components often recover the correct words, but tend to produce adjacent repetitions and place them less accurately within the sequence. To understand this, we use 64 held-out OpenWebText sequences, each containing 1,024 tokens without padding. We extract the normalized final-layer Qwen representations and apply the Fourier transform along the token dimension independently for each hidden channel. The low-frequency representation retains the first 93 of 513 frequency bins, while the equally sized high-frequency representation retains the last 93 bins. All other coefficients are set to zero before applying the inverse Fourier transform.

For local group sizes from 2 to 50 tokens, we compute the cosine similarity between every pair of reconstructed token representations within the group and average across positions and sequences. No decoder is used in this experiment: similarity is measured directly between the inverse-Fourier-transformed Qwen representations. The results in Figure 12 show that low-frequency representations

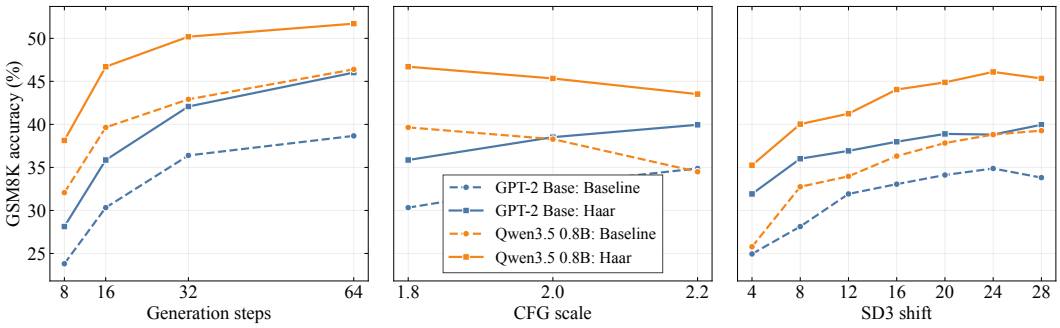


Figure 14: **GSM8K accuracy of ELF-M across sampling configurations.** We compare the baseline and Haar Delay while varying the number of generation steps, CFG scale, and SD3 sampling shift.

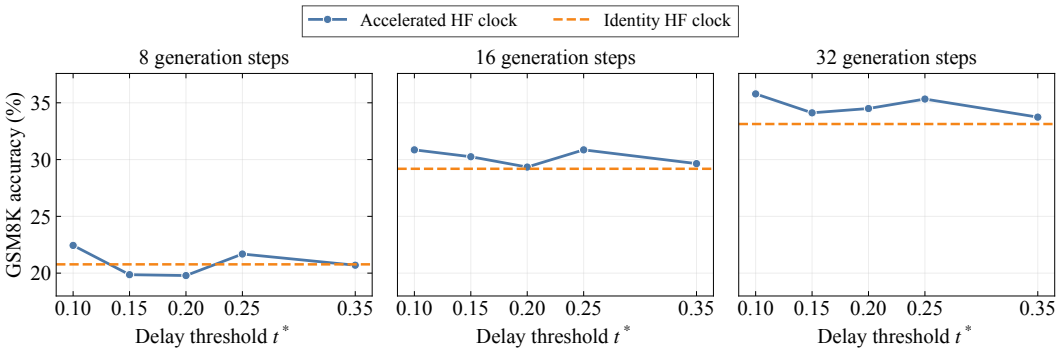


Figure 15: **Ablation of the delay threshold and HF clock on GSM8K.** We evaluate ELF-B with GPT-2 Base at different generation-step budgets, selecting the best CFG scale and SD3 sampling shift for each configuration. Smaller delay thresholds perform better. The accelerated HF clock outperforms the identity clock.

are substantially more similar across nearby positions, whereas high-frequency representations remain more position-specific and distinguishable.

To validate this effect in practice, we introduce a synthetic *password-order* probe. Each input is a 128-token sequence containing four adjacent marker tokens—*red*, *blue*, *green*, and *yellow*—at fixed positions. The prediction target is their ordering, yielding $4! = 24$ balanced classes. For every random filler context, we generate all 24 permutations, consequently, all classes contain exactly the same tokens and differ only in their order. We extract representations from each frozen Qwen layer and retain either 12 low-frequency or 12 high-frequency Fourier bins. Then, an identical linear classifier is trained to predict the exact permutation. Since token identity and counts are identical across classes, successful classification requires recovering positional order. As shown in Figure 13, high-frequency components enable nearly perfect order prediction across all layers and consistently outperform low-frequency components.

D Experiments

Training details. We train ELF-B (105M) and ELF-M (342M) models (Hu et al., 2026), containing 12 and 24 Transformer blocks, respectively. For each task, text is tokenized with the corresponding encoder tokenizer and mapped to the encoder's final hidden states. The text encoder remains frozen throughout training. We use the original latent scaling for T5 and statistics estimated on the training representations for GPT-2 Base and Qwen3.5-0.8B. All matched Baseline and Haar Delay runs use the same data, architecture, optimization settings, training budget, and auxiliary decoding objective.

			Baseline			Haar Delay			
Encoder	Task	Steps	μ_{train}	σ_{train}	s_{infer}	μ_{train}	σ_{train}	t^*	s_{infer}
GPT-2									
	GSM8K	32	−1.75	0.80	24	−1.75	0.80	0.10	32
	L0-Trace	4	−2.00	0.80	32	−1.50	0.80	0.05	32
		32	−2.00	0.80	32	−1.50	0.80	0.05	32
	xLAM	4	−2.00	0.80	32	−2.00	0.80	0.03	32
		32	−2.00	0.80	32	−2.00	0.80	0.01	32
	Sudoku	4	−3.00	0.80	32	−2.50	0.80	0.10	32
		32	−3.00	0.80	32	−2.50	0.80	0.10	32
	WikiSQL	4	−1.50	0.80	32	−1.50	0.80	0.01	32
		32	−1.50	0.80	32	−1.50	0.80	0.01	32
T5									
	L0-Trace	4	–	–	–	–	–	–	–
		32	–	–	–	–	–	–	–
	xLAM	4	–	–	–	–	–	–	–
		32	–	–	–	–	–	–	–
	Sudoku	4	−2.50	0.80	32	−2.50	0.80	0.15	32
		32	−2.50	0.80	32	−2.50	0.80	0.15	32
	WikiSQL	4	−1.50	0.80	32	−1.50	0.80	0.03	32
		32	−1.50	0.80	32	−1.50	0.80	0.03	32
Qwen3.5									
	GSM8K	32	−1.75	0.80	32	−1.75	0.80	0.15	24
	L0-Trace	4	−2.25	0.80	32	−2.00	0.80	0.05	32
		32	−2.25	0.80	32	−2.00	0.80	0.05	32
	xLAM	4	−1.50	0.80	32	−1.50	0.80	0.03	32
		32	−1.50	0.80	32	−1.50	0.80	0.05	32
	Sudoku	4	−3.00	0.80	32	−3.00	0.80	0.18	32
		32	−3.00	0.80	32	−3.00	0.80	0.18	32
	WikiSQL	4	−1.75	0.80	32	−1.50	0.80	0.03	32
		32	−1.75	0.80	32	−1.50	0.80	0.03	32

Table 7: Selected hyperparameters for the results in Table 2. Training timesteps are sampled from a logit-normal distribution with mean μ_{train} and standard deviation σ_{train}. At inference, s_{infer} denotes the SD3 sampling-schedule shift. The Haar threshold t^* is used only by Haar Delay. Baseline and Haar Delay are tuned independently for every encoder, task, and sampling budget. A dash denotes an unavailable experiment. For GSM8K, we report the best SD3 shift at 32 generation steps; the associated CFG scales are 2.2 for GPT-2 and 1.8 for Qwen3.5.

For GPT-2, T5 and Qwen3.5, we first tune the baseline training-time schedule and then evaluate Haar Delay around the best-performing region. The standard deviation of the logit-normal training schedule is fixed to 0.8, while its mean is selected on validation data. For Haar Delay, we additionally tune the delay threshold t^*. At inference, we sweep the number of generation steps and the SD3 schedule shift, as well as the CFG scale for conditional tasks. Baseline and Haar Delay configurations are selected independently using the task-specific validation metric.

Importantly, we align the inference budgets of Haar Delay and the baseline by matching the number of denoiser evaluations. When a sampling interval crosses t^*, Haar Delay splits that interval into two updates and calls the denoiser once more. We account for this extra call in the baseline schedule, so improvements cannot be attributed simply to more denoiser evaluations.

GSM8K. We additionally evaluate Haar Delay with the larger ELF-M model (342M parameters) on GSM8K (Cobbe et al., 2021). Given an arithmetic word problem, the model generates an executable Python function, and accuracy is measured by executing the generated code and comparing its output with the ground-truth answer. Figure 14 compares the baseline and Haar Delay across different numbers of generation steps, CFG scales, and SD3 sampling shifts for GPT-2 Base and Qwen3.5-0.8B representation spaces. Haar Delay consistently improves accuracy for both encoders across the three sweeps. Performance generally increases with additional generation steps, while the optimal CFG scale and schedule shift depend on the representation space. Under the best configuration, Haar Delay improves accuracy from 42.4% to 47.8% with GPT-2 Base and from 46.4% to 51.7% with Qwen3.5-0.8B.

Ablations on GSM8k. Haar Delay introduces a delay threshold, `t*`, and an accelerated clock that allows the HF trajectory to catch up with LF after the delay. In Figure 15, we ablate both components

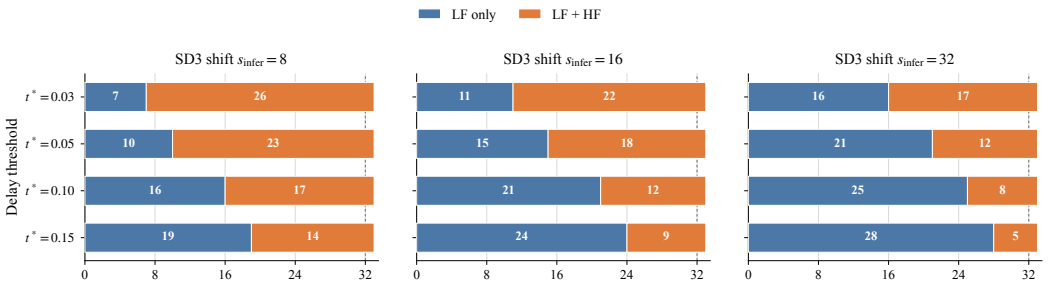


Figure 16: **Allocation of denoising updates between frequency bands.** Before t^*, only LF components are updated, after LF and HF components are updated jointly. Larger delay thresholds and SD3 sampling shifts leave fewer updates for HF. Numbers inside the bars indicate the corresponding number of denoising updates.

for ELF-B with GPT-2 Base at 8, 16, and 32 generation steps. For each configuration, we report the best result across CFG scales and SD3 sampling shifts. We find that an early threshold of $t^* = 0.10$ performs best at 8 and 32 steps and ties with $t^* = 0.25$ at 16 steps. The optimal value of `t*` can be difficult to select universally, although smaller thresholds generally perform better.

We additionally replace the accelerated HF clock with an identity clock. The accelerated clock improves accuracy by 1.7, 1.7, and 2.7 percentage points at 8, 16, and 32 steps, respectively, demonstrating the importance of allowing the delayed HF trajectory to catch up with LF.

Other benchmarks. We evaluate Haar Delay on a broader set of conditional-generation tasks: L0-Trace (Sun et al., 2025), xLAM (Liu et al., 2024), Sudoku (Rao, 2019), and WikiSQL (Zhong et al., 2017). These benchmarks cover both natural-language and structured generation. In our L0-Trace adaptation, the model generates the complete execution trace of a synthetic scalar-loop program, evaluated by exact match. xLAM conditions on a user query and the available tool schemas and generates the corresponding JSON function calls, also evaluated by exact match. For Sudoku, the model completes a partially filled 9×9 grid, and we report the percentage of valid solutions that preserve the given clues. Finally, WikiSQL requires generating an SQL query from a natural-language question and the corresponding table schema, evaluated by exact match.

For each encoder, task, and sampling budget, we first tune the shift coefficient of the logit-normal time distribution for the baseline and report the best result. We then tune `t*` for Haar Delay and report its best-performing configuration. The final hyperparameter configurations are in Table 7.

Other details. Figure 16 shows how the denoising updates are distributed between the two frequency bands. Before t^*, only the LF state is updated; after t^*, LF and HF are updated jointly. Consequently, LF follows the full sampling trajectory, whereas the number of HF updates depends on both t^* and the SD3 sampling shift. Larger values of t^* or s_{infer} leave fewer updates for HF. When t^* falls between two sampling-grid points, we split the corresponding interval at t^*.