Narrative-Dynamical Systems (NDS): A Closed-Loop Architecture for Long-Horizon Autoregressive Decoding via Orthogonal Logit Projection and Dynamic Barriers

Faruk Alpay *

Posted Date: 15 January 2026

doi: 10.20944/preprints202601.1130.v1

Keywords: autoregressive decoding; closed-loop text generation; logit space control; decoding-time intervention; long-horizon degeneration; representation drift; entropy regularization; dynamic barriers; KL trust region; language model inference

 doi:10.20944/preprints202601.1130.v1

Article

Narrative-Dynamical Systems (NDS): A Closed-Loop Architecture for Long-Horizon Autoregressive Decoding via Orthogonal Logit Projection and Dynamic Barriers

Faruk Alpay

Department of Computer Engineering, Bahçeşehir University, Istanbul, Turkey; faruk.alpay@bahcesehir.edu.tr

Abstract

Standard autoregressive language models typically generate text in an open-loop fashion, ignoring the accumulation of errors over time. Consequently, despite their local fluency, these systems frequently suffer from long-horizon pathologies such as repetitive loops, diminished lexical diversity, and distributional collapse when relying on truncation-based sampling. To address this, we present **Narrative-Dynamical Systems (NDS)**, a closed-loop decoding architecture that couples a frozen generator with a frozen encoder through a modular pre-sampling logit processor. NDS actively monitors online statistics across three channels—representation drift, token-level redundancy, and distributional concentration—and intervenes only when these signals jointly indicate a transition into a degenerate regime (low-drift/high-redundancy). The control action is injected directly into the logit space as a combination of (i) an orthogonally projected ascent step derived from a quadratic KL trust-region surrogate, and (ii) a sparse dynamic barrier designed to suppress empirically identified attractor token sets. We provide explicit derivations for the KL approximation and projection steps, alongside a closed-form bound demonstrating the exponential attenuation of probability mass assigned to the attractor set.

Keywords: autoregressive decoding; closed-loop text generation; logit space control; decoding-time intervention; long-horizon degeneration; representation drift; entropy regularization; dynamic barriers; KL trust region; language model inference

---

1. Introduction

The ability of neural language models to sustain coherent open-ended generation is not solely a function of model scale or training data; it remains critically sensitive to the decoding algorithm employed at inference time. While models are capable of high-quality outputs, standard truncation heuristics and mode-seeking objectives can produce text that is locally grammatical yet globally degenerate, exhibiting high redundancy and diminished diversity over long horizons [1,2]. This phenomenon highlights a fundamental probability–quality tension, explaining why favoring high-probability continuations often results in dull or repetitive sequences in unconstrained settings [3].

To mitigate these issues, recent inference-time strategies have proposed alternative sampling rules that regulate local information content [4] or employ diversity-oriented decoding objectives, such as diversity-augmented beam search [5]. A complementary line of work introduces auxiliary predictors or experts to steer generation without updating the base generator parameters [6–8]. Furthermore, representation-aware decoding approaches based on contrastive objectives provide evidence that

 doi:10.20944/preprints202601.1130.v1

degeneration can be mediated by interventions that explicitly couple token selection to representation geometry [9].

In this work, we adopt a systems engineering perspective: we treat long-horizon decoding as a feedback process where the model constantly conditions on its own prior outputs. Within this framework, the induced closed-loop dynamics can drift into unstable regimes characterized by low representation drift, high token redundancy, and collapsing entropy. NDS targets this mechanism by specifying an explicit architecture and a logit-level control law that (i) ensures the model remains within a local trust region around the base distribution and (ii) actively suppresses empirically detected attractor sets. By treating the generator as a fixed inference component that exposes logits prior to sampling, we implement this control action as a modular, plug-and-play logit processor.

2. Related Work

Degeneration under open-ended decoding was characterized by Holtzman et al. [1], who show that truncation-based sampling changes the shape of the sampled distribution yet does not eliminate repetitive failure modes. Unlikelihood training penalizes repeated tokens at training time and provides a widely used point of comparison when parameter updates are permitted [2]. Meister et al. analyze the probability–quality paradox [3] and propose locally typical sampling as an inference-time correction [4]. Diversity-promoting decoding has been studied in the context of structured search procedures such as Diverse Beam Search [5]. Controlled generation methods such as PPLM and FUDGE modify token probabilities using auxiliary predictors while keeping the base generator frozen [6,7]; decoding-time expert mixtures provide another parameter-free control mechanism [8]. Contrastive frameworks connect degeneration to representation anisotropy and propose decoding rules that trade off plausibility and diversity in representation space [9]. Distributional evaluation metrics such as MAUVE quantify the gap between generated and reference text distributions under decoding interventions [10]; benchmarking toolkits such as Texygen support standardized evaluation of diverse generators and metrics [11].

3. System Architecture

NDS is defined as the composition of a frozen autoregressive generator, a frozen encoder, online statistics maintained over a sliding window, and a pre-sampling logit processor that injects control.

3.1. Components and Interfaces

Let G_θ be a generator (decoder-only LM) that maps a prefix h_t to logits z_t:

$$G_\theta : \; h_t \mapsto z_t = z_\theta(h_t) \in \mathbb{R}^{|\mathcal{V}|}, \qquad p_t(\cdot) = \mathrm{Softmax}(z_t).$$

Let E_ψ be an encoder that maps the same prefix to a compact state:

$$E_\psi : \; h_t \mapsto x_t = \phi(h_t) \in \mathbb{R}^d.$$

The control module computes $\delta_t \in \mathbb{R}^{|\mathcal{V}|}$ from (x_t), token and distribution statistics on a sliding window, and a one-step candidate set computed from p_t. The controlled distribution is

$$q_t(\cdot \mid h_t) = \mathrm{Softmax}(z_t + \delta_t), \qquad y_t \sim q_t(\cdot \mid h_t).$$

The generator parameters θ and encoder parameters ψ remain fixed throughout inference; all interventions are applied to logits prior to sampling.

 doi:10.20944/preprints202601.1130.v1

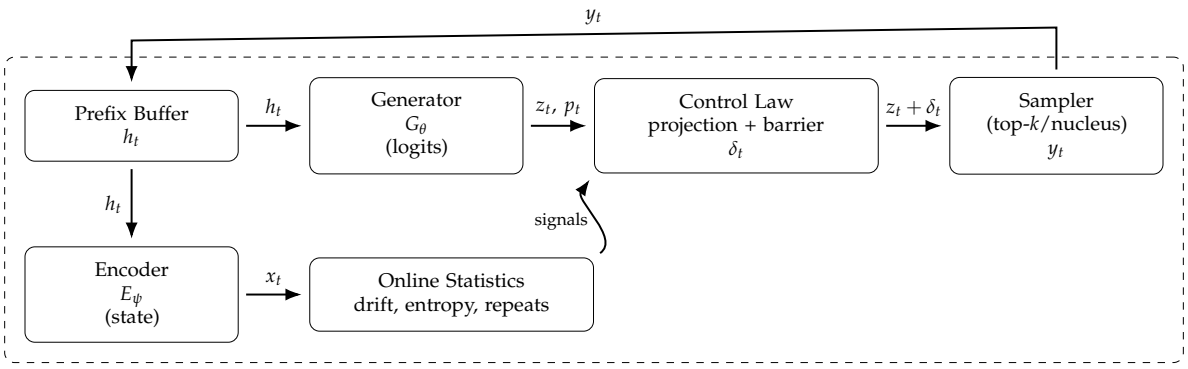

Figure 1. NDS architecture. Control is injected as a logit processor prior to sampling; the encoder and generator remain frozen.

4. Coupled Diagnostics for Long-Horizon Degeneration

NDS does not rely on a single proxy. Instead, it triggers control when evidence is consistent across (i) representation drift, (ii) token redundancy, and (iii) distributional concentration.

4.1. Representation Drift

Define the state and drift:

$$x_t = \phi(h_t), \qquad v_t = \|x_t - x_{t-1}\|.$$

Let $\bar{v}_t$ be a windowed drift average:

$$\bar{v}_t \triangleq \frac{1}{W} \sum_{i=t-W}^{t-1} v_i.$$

Low $\bar{v}_t$ indicates that the representation evolves slowly under the current decoding regime.

4.2. Token Redundancy

Let $\mathcal{G}_n(h_t)$ be the multiset of n-grams in the last W tokens of h_t. Define the repeated n-gram participation rate:

$$r_n(t) \triangleq \frac{1}{W} \sum_{j=t-W}^{t-1} \mathbf{1}\Big[\exists\, g \in \mathcal{G}_n(h_t) : g \text{ occurs at least twice and } y_j \in g\Big].$$

This quantity is invariant to embedding choices and detects local redundancy in the generated token stream.

4.3. Distributional Concentration

Let $p_t = \mathrm{Softmax}(z_t)$. Define entropy:

$$\mathrm{H}(p_t) \triangleq -\sum_{i \in \mathcal{V}} p_t(i) \log p_t(i).$$

Low entropy indicates concentration of probability mass on a small subset of tokens and is frequently associated with repetitive continuations under long-horizon decoding [1,3].

4.4. Trigger Predicate

Definition 1 (Trigger predicate)**.** *For thresholds* $(\epsilon_v, \epsilon_r, \epsilon_H)$*,*

$$\texttt{trigger}_t \triangleq \mathbf{1}\Big[\bar{v}_t < \epsilon_v \ \wedge\ r_n(t) > \epsilon_r \ \wedge\ \mathrm{H}(p_t) < \epsilon_H\Big].$$

 doi:10.20944/preprints202601.1130.v1

Triggering is intentionally conservative: the controller activates only when all three channels indicate entry into a low-drift, high-redundancy, low-entropy regime.

5. Attractor Sets and Dynamic Barriers

NDS constructs an attractor set $\mathcal{A}_t \subset \mathcal{V}$ online from windowed statistics and imposes a sparse barrier in logit space.

5.1. Frequency and Repetition Criteria

Let $f_t(w)$ be the empirical frequency of token w in the last W steps:

$$f_t(w) \triangleq \frac{1}{W} \sum_{j=t-W}^{t-1} \mathbf{1}[y_j = w].$$

Fix thresholds (τ_f, τ_r). Define the attractor set:

$$\mathcal{A}_t \triangleq \{w \in \mathcal{V} : \ f_t(w) \geq \tau_f\} \ \cup \ \{w \in \mathcal{V} : \ w \text{ participates in repeated } n\text{-grams at rate } \geq \tau_r\}.$$

5.2. Barrier Definition

Let $\sigma(\cdot)$ be a smooth step function (e.g., logistic). Define the recent attractor density

$$\rho_t \triangleq \frac{1}{W} \sum_{j=t-W}^{t-1} \mathbf{1}[y_j \in \mathcal{A}_t].$$

The barrier is a sparse logit penalty:

$$(\delta_t^{\text{bar}})_i \triangleq -\beta \, \mathbf{1}[i \in \mathcal{A}_t] \cdot \sigma(\rho_t - \theta), \qquad \beta > 0.$$

Proposition 1 (Barrier suppresses attractor-set probability mass)**.** *Let $p = \text{Softmax}(z)$ and let $\mathcal{A} \subset \mathcal{V}$. Define $q = \text{Softmax}(z + \delta)$ where $\delta_i = -\beta$ for $i \in \mathcal{A}$ and 0 otherwise, with $\beta > 0$. Then*

$$q(\mathcal{A}) = \frac{e^{-\beta} p(\mathcal{A})}{e^{-\beta} p(\mathcal{A}) + (1 - p(\mathcal{A}))} \leq e^{-\beta} \, \frac{p(\mathcal{A})}{1 - p(\mathcal{A})}.$$

Proof. Since $q(i) \propto p(i) e^{\delta_i}$, we have

$$q(\mathcal{A}) = \frac{\sum_{i \in \mathcal{A}} p(i) e^{-\beta}}{\sum_{i \in \mathcal{A}} p(i) e^{-\beta} + \sum_{i \notin \mathcal{A}} p(i)} = \frac{e^{-\beta} p(\mathcal{A})}{e^{-\beta} p(\mathcal{A}) + (1 - p(\mathcal{A}))}.$$

The inequality follows by bounding the denominator below by $(1 - p(\mathcal{A}))$. □

6. Control Law: Orthogonal Projection under a KL Trust Region

When $\texttt{trigger}_t = 1$, NDS computes a projected step designed to increase a one-step diversification surrogate while remaining near the base distribution.

6.1. One-Step Surrogate and Score-Function Direction

Define the window centroid in state space:

$$\bar{x}_t = \frac{1}{W} \sum_{i=t-W}^{t-1} x_i, \qquad \mathcal{J}(x) = \frac{1}{2} \|x - \bar{x}_t\|^2.$$

 doi:10.20944/preprints202601.1130.v1

Construct a candidate set $\mathcal{C}_t$ from p_t (top-k or nucleus). Define one-step scores

$$\widehat{\mathcal{J}}(y; h_t) = \mathcal{J}\big(\phi(h_t \oplus y)\big).$$

Using $\nabla_z \log p_t(y) = \mathbf{e}_y - p_t$, define a finite-sum score-function direction:

$$g_t \triangleq \sum_{y \in \mathcal{C}_t} w_t(y)\,\big(\widehat{\mathcal{J}}(y; h_t) - b_t\big)\,(\mathbf{e}_y - p_t),$$

where w_t are normalized base probabilities restricted to $\mathcal{C}_t$ and b_t is a baseline (e.g., the weighted mean of $\widehat{\mathcal{J}}$ over $\mathcal{C}_t$).

6.2. Second-Order KL Approximation

Let $p = \mathrm{Softmax}(z)$ and $q = \mathrm{Softmax}(z + \delta)$. For sufficiently small δ, a second-order approximation holds.

Lemma 1 (Second-order KL approximation in logit space)**.** *Let $p = \mathrm{Softmax}(z)$ and $q = \mathrm{Softmax}(z + \delta)$. Then*

$$\mathrm{KL}(q \| p) = \frac{1}{2}\,\delta^\top H\,\delta + O(\|\delta\|^3), \qquad H = \mathrm{diag}(p) - pp^\top.$$

Proof. A Taylor expansion of $\mathrm{KL}(\mathrm{Softmax}(z + \delta) \| \mathrm{Softmax}(z))$ around $\delta = 0$ yields a zero first-order term and a Hessian equal to the Fisher information of the categorical family parameterized by logits, namely $\mathrm{diag}(p) - pp^\top$. □

6.3. Orthogonal Projection and Minimum-Norm Characterization

Let c_t be a constraint direction. Define the orthogonal projection:

$$g_t^\perp \triangleq g_t - \frac{\langle g_t, c_t \rangle}{\|c_t\|^2 + \epsilon_c}\, c_t.$$

Lemma 2 (Minimum-norm projection)**.** *For $c \neq 0$, $g^\perp$ is the unique solution of*

$$\min_u \|u - g\|^2 \quad \text{s.t.} \quad \langle c, u \rangle = 0.$$

Proof. Let $L(u, \lambda) = \|u - g\|^2 + 2\lambda \langle c, u \rangle$. Stationarity gives $u = g - \lambda c$. Enforcing $\langle c, u \rangle = 0$ yields $\lambda = \langle c, g \rangle / \|c\|^2$, and thus $u = g - (\langle c, g \rangle / \|c\|^2)c$. □

6.4. Trust-Region Scaling

A practical scaled step is obtained by using the quadratic KL surrogate:

$$\delta_t^{\mathrm{proj}} \triangleq \sqrt{\frac{2\kappa}{(g_t^\perp)^\top H^{-1} g_t^\perp}}\, H^{-1} g_t^\perp \quad \text{when } g_t^\perp \neq 0.$$

6.5. Composite Control

The final control is

$$\delta_t \triangleq \mathtt{trigger}_t \cdot \delta_t^{\mathrm{proj}} + \delta_t^{\mathrm{bar}}, \qquad q_t = \mathrm{Softmax}(z_t + \delta_t).$$

 doi:10.20944/preprints202601.1130.v1

7. Algorithm

Algorithm 1 NDS closed-loop decoding (single step)

Require: prefix h_t, logits z_t, window size W

$x_t \leftarrow E_\psi(h_t)$; update $\bar{v}_t$, $r_n(t)$, and $\mathrm{H}(p_t)$

$\texttt{trigger}_t \leftarrow \mathbf{1}[\bar{v}_t < \epsilon_v \wedge r_n(t) > \epsilon_r \wedge \mathrm{H}(p_t) < \epsilon_H]$

$p_t \leftarrow \mathrm{Softmax}(z_t)$; build candidate set $\mathcal{C}_t$ from p_t (top-k/nucleus)

Compute one-step scores $\widehat{\mathcal{J}}(y; h_t)$ for $y \in \mathcal{C}_t$

Compute g_t; project to $g_t^{\perp}$; scale to δ_t^{proj} under the KL surrogate

Build $\mathcal{A}_t$ from window counts and repeated n-grams; compute sparse barrier δ_t^{bar}

$\delta_t \leftarrow \texttt{trigger}_t \cdot \delta_t^{\mathrm{proj}} + \delta_t^{\mathrm{bar}}$

Sample $y_t \sim \mathrm{Softmax}(z_t + \delta_t)$; update $h_{t+1} \leftarrow h_t \oplus y_t$

8. Implementation Considerations

NDS requires two operational interfaces: (i) an encoder that maps the evolving prefix to a fixed-dimensional representation, and (ii) a generator runtime that exposes logits prior to sampling so that a logit processor can inject δ_t. The fragments below illustrate a deterministic windowed-state implementation and sparse logit-bias injection using the `llama.cpp` library [13]; they are representative of a standard inference stack where pre-sampling hooks are available.

8.1. Windowed Statistics and Attractor-Set Construction

A ring buffer maintains the last W tokens and W embeddings, along with a hash map for token frequencies. Repeated n-gram participation is computed on the token ring buffer; in a production setting the repeated n-gram detector is implemented with rolling hashes to avoid quadratic scans in W.

Listing 1. C++: windowed statistics and frequency-based attractor set.

```cpp
struct WindowStats {
  int W;
  std::vector<llama_token> tokens;   // ring buffer
  std::vector<std::vector<float>> X; // ring buffer of embeddings
  int head = 0, size = 0;

  std::unordered_map<llama_token,int> freq;

  explicit WindowStats(int W_, int d): W(W_) {
    tokens.resize(W);
    X.assign(W, std::vector<float>(d, 0.0f));
  }

  void push(llama_token t, const std::vector<float>& x) {
    if (size == W) {
      auto old = tokens[head];
      auto it = freq.find(old);
      if (it != freq.end() && --it->second == 0) freq.erase(it);
    } else size++;

    tokens[head] = t;
    X[head] = x;
    freq[t]++;

    head = (head + 1) % W;
  }

```

 doi:10.20944/preprints202601.1130.v1


```
  float drift() const {
    if (size < 2) return 1.0f;
    int i1 = (head - 1 + W) % W;
    int i0 = (head - 2 + W) % W;
    float s = 0.0f;
    for (size_t j = 0; j < X[i1].size(); ++j) {
      float d = X[i1][j] - X[i0][j];
      s += d * d;
    }
    return std::sqrt(s);
  }

  std::vector<llama_token> attractor_by_freq(float tau_f) const {
    std::vector<llama_token> A;
    if (size == 0) return A;
    for (auto& kv : freq) {
      float ft = float(kv.second) / float(size);
      if (ft >= tau_f) A.push_back(kv.first);
    }
    return A;
  }
};
```

8.2. Sparse Logit Injection

Projection is computed on candidate-supported vectors; the barrier is applied only to the sparse attractor set $\mathcal{A}_t$. Both enter as additive bias terms prior to the sampling decision.

Listing 2. C++: sparse barrier insertion in a sampler chain.

```
void apply_sparse_barrier(llama_sampler* smpl,
                          const std::vector<llama_token>& A,
                          float bias = -80.0f) {
  std::vector<llama_logit_bias> biases;
  biases.reserve(A.size());
  for (auto t : A) biases.push_back({t, bias});

  llama_sampler_chain_add(
    smpl,
    llama_sampler_init_logit_bias(
      llama_vocab_n_tokens(llama_model_get_vocab(model_main)),
      (int)biases.size(),
      biases.data()
    )
  );
}
```

9. Evaluation Methodology

This section states an experimental design aligned with prior decoding studies.

9.1. Baselines

Baselines are implemented as alternative pre-sampling interventions operating on the same generator logits. We include: (i) nucleus sampling [1], (ii) locally typical sampling [4], (iii) diversity-oriented search procedures [5], (iv) auxiliary-predictor steering [6,7], and (v) decoding-time expert mixtures [8]. Where applicable, representation-aware decoding rules are included for comparison [9]. All methods share identical tokenization, stopping criteria, and prompt suite.

 doi:10.20944/preprints202601.1130.v1

9.2. Prompt Suite

A fixed prompt suite is used with stratified lengths and styles to probe long-horizon behavior. For each prompt, generation is run for a fixed token budget and evaluated under identical termination rules. The prompt suite is held constant across methods to isolate the effect of decoding-time interventions.

9.3. Metrics

We report complementary metrics capturing redundancy, diversity, and distributional shift: (i) *Distinct-n* (ratio of unique n-grams to total n-grams) [11]; (ii) repetition rate via repeated n-gram participation $r_n(t)$ aggregated over the generation; (iii) *Self-BLEU* as a corpus-level indicator of mode concentration [11]; and (iv) *MAUVE* as a distributional metric comparing generated and reference text [10]. Metrics are interpreted jointly due to known trade-offs between diversity and local coherence under decoding interventions.

10. Conclusion

NDS formalizes long-horizon decoding as a feedback process and proposes a closed-loop architecture in which a fixed generator is coupled to a fixed encoder through a pre-sampling logit processor. Control is activated only under conservative, multi-signal triggering and is implemented as the sum of an orthogonally projected trust-region step and a sparse dynamic barrier that exponentially suppresses probability mass on empirically identified attractor sets. The analysis provides closed-form derivations for the KL surrogate and the projection operator, together with an explicit attenuation bound for attractor-set mass, yielding a principled decoding-time intervention that does not require updating generator parameters.

References

1. A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2020.
2. S. Welleck, I. Kulikov, S. Roller, E. Dinan, K. Cho, and J. Weston. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*, 2019.
3. C. Meister, G. Wiher, T. Pimentel, and R. Cotterell. On the probability–quality paradox in language generation. *arXiv preprint arXiv:2203.17217*, 2022.
4. C. Meister, T. Pimentel, G. Wiher, and R. Cotterell. Locally typical sampling. *arXiv preprint arXiv:2202.00666*, 2022.
5. A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. Crandall, and D. Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2018.
6. S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2020.
7. K. Yang and D. Klein. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, 2021.
8. A. Liu, M. Sap, X. Lu, S. Swayamdipta, C. Bhagavatula, N. A. Smith, and Y. Choi. DExperts: Decoding-time controlled text generation with experts and anti-experts. *arXiv preprint arXiv:2105.03023*, 2021.
9. Y. Su, T. Lan, Y. Wang, D. Yogatama, L. Kong, and N. Collier. A contrastive framework for neural text generation. *arXiv preprint arXiv:2202.06417*, 2022.
10. K. Pillutla, S. Swayamdipta, R. Zellers, J. Thickstun, S. Welleck, Y. Choi, and Z. Harchaoui. MAUVE: Measuring the gap between neural text and human text using divergence frontiers. *arXiv preprint arXiv:2102.01454*, 2021.
11. Y. Zhu, S. Lu, L. Zheng, J. Guo, W. Zhang, J. Wang, and Y. Yu. Texygen: A benchmarking platform for text generation models. *arXiv preprint arXiv:1802.01886*, 2018.

 doi:10.20944/preprints202601.1130.v1

12. N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.
13. G. Gerganov. llama.cpp: Port of Facebook's LLaMA model in C/C++. *GitHub repository*, https://github.com/ggerganov/llama.cpp, 2023.