Generative Inverse Design of Sequence Dynamics

A solvable benchmark, conditional sequence generation, and a multi-fidelity route toward RNA response design

Ouraman Hajizadeh | Bridging ML and SFT through generative and diffusion processes, ECT*

September 2026

RNA therapeutic development

Real RNA design problem
dynamical response could signal functional performance

$$\text{sequence} \longrightarrow \text{conformational dynamics / relaxation} \longrightarrow \text{functional performance.}$$

Which RNA sequences produce a desired dynamical behavior?

MD, high-quality coarse-grained simulations, or experiments computationally and experimentally expensive.

Role of the present study
We replace the realistic RNA system by a much cheaper, heterogeneous chain, with formally controlled dynamics
First step:

$$\text{sequence} \longrightarrow \text{exact response } (\eta_\infty, \tau_\infty) \longrightarrow \text{inverse sequence design.}$$

This gives a controlled benchmark for testing whether a **response-conditioned generative model** can concentrate proposals in a desired dynamical region

The inverse-design question

Given a desired dynamical or response phenotype, can we propose sequences likely to realize it *without exhaustively evaluating sequence space*?

$$s \xrightarrow{\text{forward model}} y(s)$$

$$y^* \xrightarrow{\text{inverse design}} s^*$$

Two coupled problems appear:

- **Modeling:** evaluate the response of a sequence.
- **Search:** find rare sequences with the requested response.

If evaluation is cheap, search is straightforward

A standard explore–exploit loop is enough when each candidate is inexpensive:

1. explore broadly in sequence or parameter space;
2. evaluate every proposal with the trusted model;
3. concentrate later proposals around promising regions;
4. repeat until the target or the evaluation budget is reached.

$$\underbrace{s_1, \ldots, s_P}_{\text{propose}} \longrightarrow \underbrace{y(s_1), \ldots, y(s_P)}_{\text{evaluate}}$$

new proposals $\leftarrow$ observed response

The hard case is when one trustworthy response evaluation is itself expensive.

For RNA dynamics, the forward evaluation can dominate the cost

- Atomistic RNA simulations include solvent, ions, base interactions, and many degrees of freedom.
- Fast molecular motions force small integration steps.
- Slow or rare conformational transitions demand long trajectories
- Correlation-based observables may require replicas and careful convergence checks.

candidate sequences $\times$ long simulations

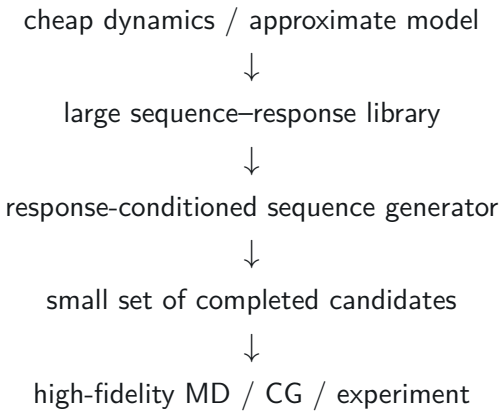
$\Downarrow$

nested computational cost

A forward surrogate is useful

only after enough trustworthy data exist to train and calibrate it.

Strategy: cheap proposal, expensive verification



The generator is not the physical oracle. Its role is to concentrate a finite evaluation budget into useful parts of sequence space.

The high-fidelity evaluator returns the data needed to improve:

- a prefix critic,
- a low-to-high-fidelity residual model,
- and eventually the inverse generator itself.

Active learning loop: one complete iteration

1. Generate a large cheap training library $\mathcal{D}_{\mathrm{LF}} = \{(s, y_{\mathrm{LF}})\}$.
2. Train a conditional generator $p_\theta(s \mid y_{\mathrm{LF}}, e)$.
3. For target y^*, generate many candidate sequences autoregressively, using the prefix critic to guide next-token sampling, preserving diversity.
4. For each completed candidate, compute the low-fidelity response and use the residual surrogate to predict its high-fidelity correction and uncertainty.
5. Rank the completed candidates using predicted target quality, uncertainty, and diversity, and select an acquisition batch A.
6. Evaluate only A with high-fidelity MD, CG, or experiment.
7. Add the new high-fidelity observations to D_{HF} and update the prefix critic, residual surrogate, and optionally the generator.
8. Repeat until hit rate, uncertainty, or the available high-fidelity budget stops improving.

Conditional GPT: training and generation

Training: learn the inverse proposal

Generate sequence–response pairs:

$$s^{(n)} \xrightarrow{\text{physics}} y^{(n)} = (\eta_\infty^{(n)}, \tau_\infty^{(n)}).$$

Standardize the log-response:

$$c^{(n)} = \left(\frac{\log \eta_\infty^{(n)} - \mu_\eta}{\sigma_\eta}, \frac{\log \tau_\infty^{(n)} - \mu_\tau}{\sigma_\tau} \right).$$

Learn the conditional autoregressive distribution:

$$p_\theta(s \mid c) = \prod_{i=1}^{L} p_\theta(s_i \mid s_{<i}, c).$$

Generation: query a target response

Choose

$$y^* = (\eta^*, \tau^*)$$

and transform it to the same condition c^*. Generate the sequence autoregressively:

$$[\mathrm{BOS}] \to s_1 \to s_1 s_2 \to \cdots \to s_{1:L}.$$

At position i, the GPT predicts

$$p_\theta(s_i = b \mid s_{<i}, c^*), \qquad b = 0, \ldots, 7.$$

Sample using temperature and top-k:

$$p_i(b) \propto \exp\left(\frac{z_i(b)}{T} \right), \qquad T = 0.9, \quad k = 5.$$

The critic turns one expensive sequence into many prefix labels

Suppose a completed high-fidelity evaluation gives a scalar design reward

$$R_{\mathrm{HF}}(s; y^*)$$

for a full sequence $s = (s_1, \ldots, s_L)$. Every prefix of that sequence can inherit this final outcome as a training target:

$$(s_{1:i}, y^*) \longrightarrow R_{\mathrm{HF}}(s; y^*), \qquad i = 1, \ldots, L.$$

A simple value-model objective is

$$\mathcal{L}_V = \frac{1}{M} \sum_{n=1}^{M} \frac{1}{L} \sum_{i=1}^{L} \left[V_\psi(s_{1:i}^{(n)}, y_n^*) - R_{\mathrm{HF}}(s^{(n)}; y_n^*) \right]^2 .$$

One expensive completed sequence supplies supervision at every token position

How the critic can guide token-by-token generation

The cheap conditional GPT gives logits $z_i^{\mathrm{GPT}}(b)$ for the next token b. A prefix critic can modify them before sampling:

$$\widetilde{z}_i(b) = z_i^{\mathrm{GPT}}(b) + \beta\, V_\psi(s_{<i} \oplus b,\, y^*,\, e).$$

Then sample from

$$q_i(b) \propto \exp\left(\frac{\widetilde{z}_i(b)}{T}\right).$$

Interpretation:

- G_θ remembers broad sequence–response structure learned cheaply.
- V_ψ adds high-fidelity knowledge about whether the current prefix tends to end well.
- β controls how strongly high-fidelity guidance overrides the base proposal distribution.

Final completed candidates are still verified by the trusted high-fidelity evaluator.

A residual surrogate corrects systematic low-fidelity bias

A critic answers: "will this candidate score well?" A residual surrogate answers a different question: "how will the cheap response change at high fidelity?"

$$\Delta_\phi(s, e) \approx y_{\mathrm{HF}}(s, e) - y_{\mathrm{LF}}(s, e).$$

Then a corrected prediction is

$$\widehat{y}_{\mathrm{corr}}(s, e) = y_{\mathrm{LF}}(s, e) + \Delta_\phi(s, e).$$

If the residual model also produces uncertainty $\sigma_\Delta(s)$, an acquisition score can trade off exploitation, uncertainty, and diversity:

$$a(s) = \underbrace{\mu_R(s)}_{\text{promising}} + \kappa \underbrace{\sigma_R(s)}_{\text{uncertain}} + \lambda \underbrace{d(s, \mathcal{D}_{\mathrm{HF}})}_{\text{diverse}}.$$

This reduces the risk that active learning repeatedly evaluates only near-duplicate sequences already favored by the current surrogate.

Verify first step using a minimal model

The full multi-fidelity loop is expensive to develop directly on realistic RNA dynamics. For development we work with a nonlinear chain. Here, however, we first test the inverse-design step in the exactly solvable harmonic limit.

So the present work asks a narrower algorithmic question:

Can a response-conditioned sequence model learn a useful inverse proposal distribution

From a heterogeneous chain to an exactly solvable model

A generic overdamped heterogeneous chain can be written as

$$\mathbf{u}(t) \in \mathbb{R}^S, \qquad S = L + 1,$$

$$d\mathbf{u} = -\mu \nabla_{\mathbf{u}} H(\mathbf{u}; s)\, dt + \sqrt{2\mu k_B T}\, d\mathbf{W}_t, \qquad H(\mathbf{u}; s) = \sum_{i=1}^{L} V_{s_i}(u_{i+1} - u_i).$$

For general anharmonic (non-quadratic) V_{s_i}, response functions require finite trajectories and convergence analysis.

The benchmark:

$$V_{s_i}(\Delta u) = \frac{1}{2} k_i(s) (\Delta u)^2,$$

so the response can be evaluated by eigendecomposition and closed modal sums.

Define a deformation observable

Choose the spatially modulated deformation operator

$$B = D_1^{\mathsf{T}} \operatorname{diag}(b_i) D_1, \qquad b_i = \cos\left[\frac{2\pi(i+1/2)}{E}\right].$$

The quadratic observable is

$$q(u) = \frac{1}{2} u^{\mathsf{T}} B u = \frac{1}{2} \sum_{r,s} B_{rs} a_r a_s, \qquad B_{rs} = v_r^{\mathsf{T}} B v_s.$$

We study the connected equilibrium autocorrelation

$$C_q(t) = \langle q(0) q(t) \rangle - \langle q \rangle^2.$$

The operator is deliberately simple and synthetic. It creates a nonlocal response that depends on the sequence collective modes. inverse-design gets some of the global sequence sensitivity expected in more realistic molecular systems, keeping the dynamics exactly solvable.

What η_∞ and τ_∞ measure

By definition,

$$\eta_\infty = \frac{1}{k_B T V} \int_0^\infty C_q(t)dt, \qquad \tau_\infty = \frac{\int_0^\infty C_q(t)dt}{C_q(0)}.$$

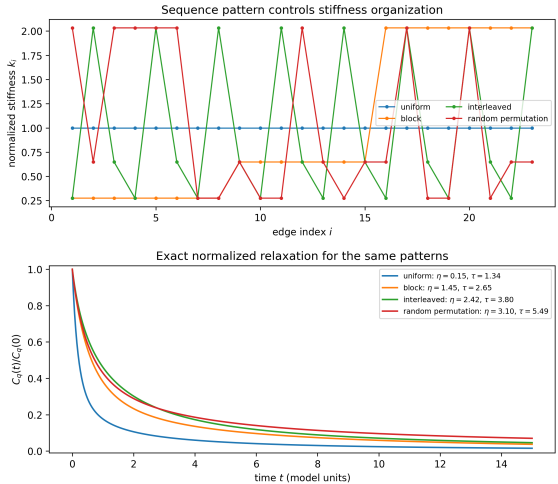
Eliminating the common integral immediately gives

$$\boxed{\eta_\infty = \frac{C_q(0)}{k_B T V}\, \tau_\infty.}$$

Therefore:

- $C_q(0)$ measures the equilibrium fluctuation amplitude of the deformation observable;
- τ_∞ measures the normalized persistence time of those fluctuations;
- η_∞ combines amplitude and persistence in one integrated response coefficient.

The same stiffness composition can produce different dynamics



Inverse design: find a sequence inside a response target

For L edges and eight tokens,

$$|\mathcal{S}| = 8^L.$$

At $L = 23$,

$$8^{23} \approx 5.90 \times 10^{20},$$

and the space grows to about 10^{65} at $L = 72$.

The target at $L = 23$ is centered at

$$(\eta_c, \tau_c) = (1.66069, 2.73170)$$

with symmetric log half-width $h = 0.12$.

For a positive response coordinate y,

$$m_y(y) = \min\{\log y - \log y_L, \log y_H - \log y\}.$$

The joint score is

$$m(s) = \min\{m_\eta, m_\tau\}.$$

Hence

$$m(s) \geq 0 \iff s \text{ is a target hit.}$$

Training data: forward physics first, inverse model second

Generate 15,000 random sequences from the uniform prior and evaluate each exactly:

$$s^{(n)} \xrightarrow{\text{harmonic solver}} (\eta_\infty^{(n)}, \tau_\infty^{(n)}).$$

The training set is

$$\mathcal{D} = \left\{ (s^{(n)}, \eta_\infty^{(n)}, \tau_\infty^{(n)}) \right\}_{n=1}^{15000}.$$

The important direction reversal is

$$\underbrace{s \to (\eta, \tau)}_{\text{create labels}} \qquad \Longrightarrow \qquad \underbrace{(\eta, \tau) \to \text{likely } s}_{\text{learn inverse proposals}}.$$

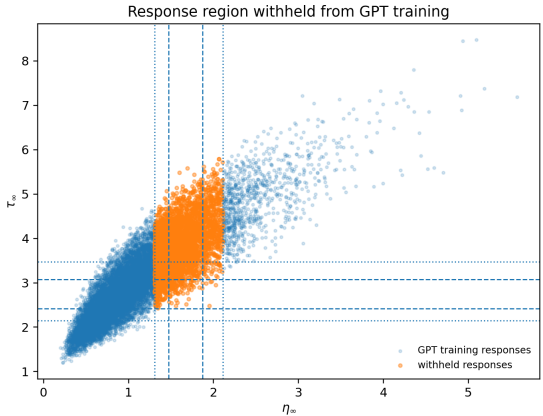
Interpolation: remove the requested response neighborhood from training

For the held-out experiment, remove every calibration example with

$$\eta_\infty \in [1.30635, 2.11115],$$

including all target-box hits. This leaves 11,384 training pairs.

Generation is still requested at the target center inside the missing band.



GPT conditioning: turn the physical response into a model input

The response associated with each training sequence is standardized:

$$c = \begin{pmatrix} (\log \eta_\infty - \mu_\eta)/\sigma_\eta \\ (\log \tau_\infty - \mu_\tau)/\sigma_\tau \end{pmatrix}.$$

The causal sequence model factorizes

$$\boxed{p_\theta(s \mid c) = \prod_{i=1}^{L} p_\theta(s_i \mid s_{<i}, c).}$$

At every sequence position the hidden input is approximately

$$h_i = e_{\text{token}}(x_i) + e_{\text{pos}}(i) + e_{\text{cond}}(c).$$

The same condition embedding is supplied at every position.

GPT architecture

For the $L = 23$ benchmark:

- one causal Transformer layer;
- model width $d_{\text{model}} = 32$;
- four attention heads;
- feed-forward width 64;
- learned token and positional embeddings;
- eight output logits, one per stiffness token;
- no dropout.

Small GPT by design: the benchmark physics and dataset are simple, so extra model complexity is unnecessary and would only increase training cost.

$$p(s_i \mid s_{<i}, c).$$

Training steps

For a true training sequence

$$s = (s_1, s_2, \ldots, s_L),$$

feed the shifted input

$$x = (\mathrm{BOS}, s_1, s_2, \ldots, s_{L-1})$$

predict:

$$(s_1, s_2, \ldots, s_L).$$

At position i the model is trained on

$$p_\theta(s_i \mid s_1, \ldots, s_{i-1}, c).$$

The loss is token-level cross-entropy:

$$\boxed{\mathcal{L}_{\mathrm{GPT}} = -\frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{L} \log p_\theta(s_i^{(n)} \mid s_{<i}^{(n)}, c^{(n)}).}$$

Training uses AdamW, batch size 256, for four epochs.

Generation I: fix the requested response and grow the sequence

At design time choose the target center (η_c, τ_c) and standardize it to c^* using the same training statistics.

Every candidate is generated with this fixed condition:

$$[\mathrm{BOS}] \to s_1 \to s_1 s_2 \to \cdots \to s_1 \cdots s_L.$$

At step i, the network produces eight next-token logits

$$z_i(0), \ldots, z_i(7),$$

corresponding to

$$p_\theta(s_i = b \mid s_{<i}, c^*), \qquad b = 0, \ldots, 7.$$

The model is solving an amortized inverse problem: many target values can be queried after one training run.

Generation II: temperature and top-5 sampling preserve diversity

The logits are divided by the sampling temperature $T = 0.9$:

$$p_i(b) \propto \exp\left(\frac{z_i(b)}{0.9}\right).$$

Only the five highest-scoring tokens are kept; their probabilities are renormalized, and one token is sampled.

Why sample instead of choosing the maximum at every position?

- Greedy decoding tends to collapse onto a small set of similar sequences.
- The conditional GPT is only an approximate proposal model.
- A slightly lower-probability token can lead to a sequence with a better exact physical response.

Generation III: make a pool, then hand control back to physics

Generate a pool of $P = 64$ completed sequences

$$s^{(1)}, \ldots, s^{(64)} \sim p_\theta(s \mid c^*).$$

Then evaluate every complete sequence with the exact harmonic model:

$$s^{(j)} \xrightarrow{\text{exact physics}} (\eta_\infty^{(j)}, \tau_\infty^{(j)}).$$

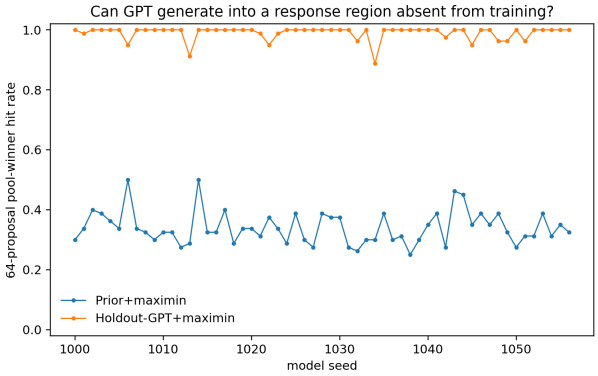
For each coordinate,

$$m_\eta = \min\left\{\log \frac{\eta}{\eta_L}, \log \frac{\eta_H}{\eta}\right\}, \quad m_\tau = \min\left\{\log \frac{\tau}{\tau_L}, \log \frac{\tau_H}{\tau}\right\},$$

and select

$$\boxed{s^* = \arg\max_j \min\{m_\eta(s^{(j)}), m_\tau(s^{(j)})\}.}$$

Held-out response interpolation at $L = 23$

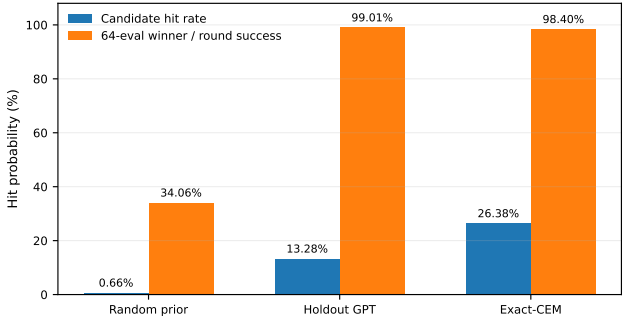


	Prior	GPT
candidate hit	0.00659	0.1328
pool-winner hit	0.3406	0.9901

57 independent model/proposal seeds; 80 pools per seed; 64 completed candidates per pool.

Candidate hit rate is measured before maximin selection and therefore measures the proposal distribution itself.

A stronger baseline: exact-model search vs. GPT



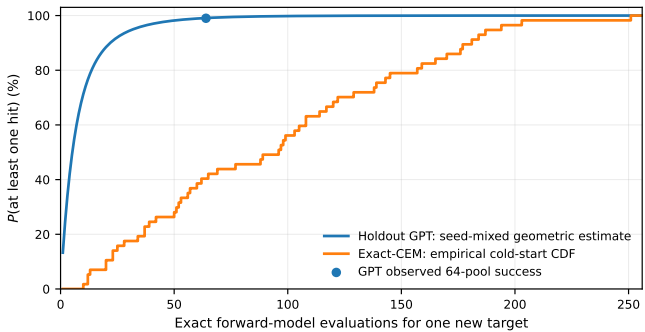
	GPT	CEM
candidate hit	13.28%	**26.38%**
64-eval success	**99.01%**	98.40%

Exact-CEM begins from a uniform distribution, queries the exact model, and adapts after each 64-candidate round.

The strong classical baseline therefore roughly doubles the trajectory-average candidate hit rate relative to GPT.

CEM candidate hit rate is averaged over all 80 adaptive rounds; GPT is a fixed post-training proposal distribution. The random prior is shown only as the legacy reference.

Query-time probability tells a different story

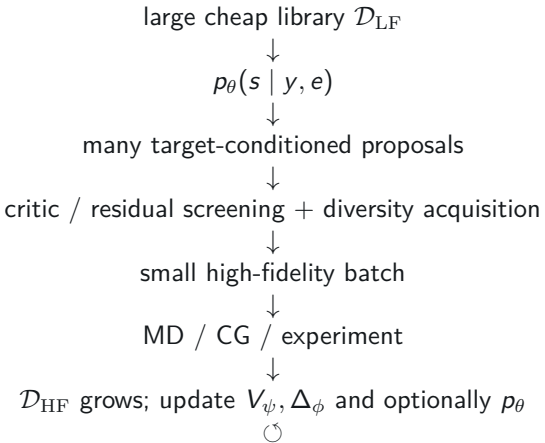


GPT has already paid its learning cost before a new target arrives.

- Expected exact verifications to first hit: **9.76** for GPT versus **94.5** for cold-start CEM.
- CEM median first hit: **96** evaluations.
- At 64 evaluations: GPT observed **99.01%** pool success; cold-start CEM has hit in only **40.35%** of runs.

GPT curve: seed-average of $1 - (1 - p_s)^N$ using the 57 measured candidate hit rates; the dot is the observed 64-pool result. CEM curve: empirical first-hit CDF across 57 independent uniform cold starts.

A practical multi-fidelity inverse-design loop



Conclusion

- small GPT conditioned on response, the first exploratory results: 0.13 candidate hit rate for the targets range so aroun 7 trials on average you will hit the target.
- traditional search beats it by a factor of 2. but parameters of conditional generation can be updated with every new full evaluation.
- the pros and cons of these two, could be studied. you can also update GPT it with any source of data.

Outlook

- train the conditioned GPT on real RNA data
- the active learning loop for anharmonic chain
- compare classical and generative inverse design

Acknowledgements

Research Center Pharmaceutical Engineering GmbH (RCPE) is funded within the framework of COMET - Competence Centers for Excellent Technologies by BMIMI, BMWET, Land Steiermark, and SFG. The COMET program is managed by the FFG.

Backup Slides

Additional details, derivations

Known Rouse theory vs. our response construction

Established theory

Rouse, *JCP* 21, 1272 (1953)
Homogeneous overdamped bead–spring chain; normal modes, relaxation times, linear viscoelasticity.

Broderix et al., *PRE* 63, 011510 (2000)
Crosslinked generalized Rouse model; exact static-viscosity/resistor-network relation and spectral stress relaxation.

Irzhak, Kuzub & Irzhak, *JEPTh* 76, 610–618 (2003)
Inhomogeneous links with different local relaxation/friction parameters; link placement changes stress/strain spectra.

Cohen, Banerjee & Pappu, *JCP* 161, 095103 (2024)
Graph-Laplacian generalized Rouse model for sequence-specific condensate storage/loss moduli.

New here: the response construction

The linear Rouse/OU dynamics and spectral solution are inherited. We introduce

$$q(u) = \frac{1}{2} u^{\mathsf{T}} B u, \;\; (D_1 \mathbf{u})_i = u_{i+1} - u_i$$

$$B = D_1^{\mathsf{T}} \mathrm{diag}(b_i) D_1, \qquad b_i = \cos \frac{2\pi(i+1/2)}{L},$$

Response summary:

$$C_q(t) \longrightarrow (\eta_\infty, \tau_\infty).$$

Why not standard Rouse viscosity?

- no literal 3-D shear stress in a scalar 1-D chain;
- order-sensitive nonlocal response at fixed mean/composition;
- inverse-design benchmark, not bulk rheology.

Harmonic model: sequence controls a weighted operator

A design is a sequence of $E = 23$ discrete stiffness labels,

$$s = (s_1, \dots, s_E), \qquad s_i \in \{0, \dots, 7\}.$$

After normalization to fixed mean stiffness,

$$k_i(s) = \bar{k}\, \frac{\ell_{s_i}}{E^{-1} \sum_{j=1}^{E} \ell_{s_j}}, \qquad \bar{k} = 1.$$

The energy is

$$H(u; s) = \frac{1}{2} u^{\mathsf{T}} L(s) u, \qquad L(s) = D_1^{\mathsf{T}} K(s) D_1.$$

The weighted path Laplacian has one translation zero mode. On the internal subspace,

$$L v_r = \lambda_r v_r, \qquad \lambda_r > 0.$$

$$u = \sum_r a_r v_r.$$

Sequence order changes both eigenvalues and eigenvectors, not merely a mean stiffness.

The internal modes are independent Ornstein–Uhlenbeck processes

Because

$$H = \frac{1}{2} \sum_r \lambda_r a_r^2,$$

the overdamped Langevin equation separates into

$$da_r = -\mu \lambda_r a_r \, dt + \sqrt{2\mu k_B T} \, dW_r.$$

The stationary density of one mode is Gaussian,

$$p_{\mathrm{eq}}(a_r) \propto \exp\left[-\frac{\lambda_r a_r^2}{2 k_B T} \right],$$

therefore

$$\boxed{\langle a_r \rangle = 0, \qquad \langle a_r^2 \rangle = \frac{k_B T}{\lambda_r}.}$$

OU solution gives the two-time mode covariance

The linear SDE has the exact solution

$$a_r(t) = e^{-\mu\lambda_r t} a_r(0) + \sqrt{2\mu k_B T} \int_0^t e^{-\mu\lambda_r (t-s)} dW_r(s).$$

The stochastic integral uses noise after time 0 and is independent of the stationary $a_r(0)$. Hence

$$\langle a_r(0) a_s(t) \rangle = e^{-\mu\lambda_s t} \langle a_r(0) a_s(0) \rangle.$$

Different normal modes are independent at equilibrium,

$$\langle a_r(0) a_s(0) \rangle = \delta_{rs} \frac{k_B T}{\lambda_r},$$

so

$$\boxed{\langle a_r(0) a_s(t) \rangle = \delta_{rs} \frac{k_B T}{\lambda_r} e^{-\mu\lambda_r t}.}$$

Wick contraction gives the exact correlation function

Insert the modal form of q:

$$C_q(t) = \frac{1}{4} \sum_{r,s,m,n} B_{rs} B_{mn} \langle a_r(0) a_s(0) a_m(t) a_n(t) \rangle_c .$$

For zero-mean Gaussian modes, Wick's theorem gives

$$\langle a_r a_s a_m a_n \rangle_c = \langle a_r a_m \rangle \langle a_s a_n \rangle + \langle a_r a_n \rangle \langle a_s a_m \rangle .$$

Using the OU covariance and symmetry $B_{rs} = B_{sr}$, the two contractions are equal:

$$\boxed{C_q(t) = \frac{(k_B T)^2}{2} \sum_{r,s} \frac{B_{rs}^2}{\lambda_r \lambda_s} e^{-\mu(\lambda_r + \lambda_s)t}.}$$

The response is therefore a positive weighted sum of exponential relaxation modes.

integrate the correlation to obtain η_∞

Define the generalized Green–Kubo response coefficient

$$\eta_\infty = \frac{1}{k_B T V} \int_0^\infty C_q(t)\, dt.$$

Because all $\lambda_r > 0$ on the internal subspace,

$$\int_0^\infty e^{-\mu(\lambda_r+\lambda_s)t} dt = \frac{1}{\mu(\lambda_r + \lambda_s)}.$$

Substituting the exact $C_q(t)$ therefore gives

$$\boxed{\eta_\infty = \frac{k_B T}{2V\mu} \sum_{r,s} \frac{B_{rs}^2}{\lambda_r \lambda_s (\lambda_r + \lambda_s)}.}$$

It is "viscosity-like" only in the Green–Kubo sense of an integrated stress/deformation correlation;

normalized relaxation time τ_∞

At $t = 0$,

$$C_q(0) = \frac{(k_B T)^2}{2} \sum_{r,s} \frac{B_{rs}^2}{\lambda_r \lambda_s}.$$

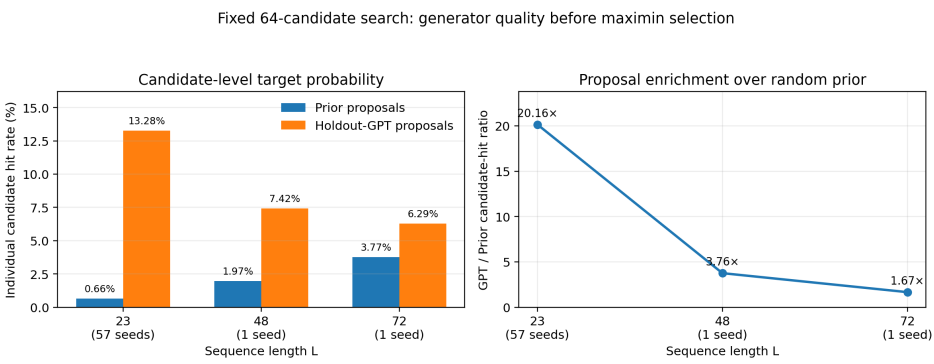
Define

$$\tau_\infty = \frac{\int_0^\infty C_q(t) dt}{C_q(0)}.$$

The common factor $(k_B T)^2/2$ cancels, giving

$$\boxed{\tau_\infty = \frac{\displaystyle\sum_{r,s} \frac{B_{rs}^2}{\lambda_r \lambda_s} \frac{1}{\mu(\lambda_r + \lambda_s)}}{\displaystyle\sum_{r,s} \frac{B_{rs}^2}{\lambda_r \lambda_s}}.}$$

Preliminary sequence-length scaling



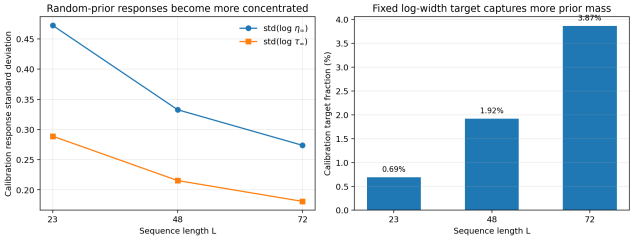
Candidate-level GPT/Prior enrichment: 20.16 at $L = 23$, 3.76 at $L = 48$, and 1.67 at $L = 72$.

64-pool winner hit rates (Prior / GPT): 0.341/0.990, 0.713/1.000, 0.875/0.988.

$L = 23$: 57-seed reference. $L = 48, 72$: single-seed pilots; the trend is descriptive, not yet a scaling law.

34

Why does the apparent advantage shrink with length?



Two effects move in the same direction:

1. **The random baseline becomes easier.** The response distribution narrows, so a fixed-width target contains more prior mass.
2. **The inverse model becomes harder.** The same small Transformer must coordinate 48 or 72 token decisions instead of 23.