Interpretable time series analysis with Gumbel dynamics

Yiliu Wang[1,2*] **Timothy Doyeon Kim**[1,2] **Eric Shea-Brown**[2] **Uygar Sümbül**[1,2]
[1]Allen Institute [2]University of Washington

Abstract

Switching dynamical systems can model complicated time series data while maintaining interpretability by inferring a finite set of dynamics primitives and explaining different portions of the observed time series with one of these primitives. However, due to the discrete nature of this set, such models struggle to capture smooth, variable-speed transitions, as well as stochastic mixtures of overlapping states (e.g., non-instantaneous state transitions), and the inferred dynamics often display spurious rapid switching on real-world datasets. Here, we propose the Gumbel Dynamical Model (GDM). First, by introducing a continuous relaxation of discrete states and a different noise model defined on the relaxed-discrete state space via the Gumbel distribution, GDM expands the set of available state dynamics, allowing the model to approximate smoother and non-stationary ground-truth dynamics more faithfully. Breaking from established literature, this new class uses a stochastic relaxed state to directly combine reusable dynamical primitives and govern observations, without a separate stochastic continuous trajectory. Second, the relaxation makes the model fully differentiable, enabling fast and scalable training with standard gradient descent methods. We validate our approach on standard simulation datasets and highlight its ability to model soft, sticky states and transitions in a stochastic setting. Furthermore, we apply our model to two real-world datasets, demonstrating its ability to infer interpretable states in stochastic time series with multiple dynamics, a setting where traditional methods often fail.

1 Introduction

Natural behaviors give rise to complex time series data with non-stationary and nonlinear dynamics. Such dynamical phenomena are often well approximated within a temporal neighborhood by a small set of distinct, interpretable motifs [30]. A family of dynamical system models aims to discover these discrete state transitions in an unsupervised manner. In particular, switching linear dynamical systems (SLDSs) formalize this observation by inferring a decomposition of the complex dynamics into locally linear dynamics primitives [2, 6, 22, 13, 10]. Only one of the dynamics primitives is used to describe the underlying data at any time point, which is defined as the state of the system. The model learns to switch between states to improve accuracy, enabling interpretable explanations of the observations. However, many real-world physical and behavioral systems evolve continuously at the temporal resolution of measurement. Even when an abstract regime label changes instantaneously, the observed trajectory may pass through a finite-duration interval whose local dynamics are not well described by either endpoint primitive alone. In such cases, interpretability of SLDS models diminishes. Moreover, switching between discrete states is prone to spurious rapid switching or requires additional transient states under the influence of complex noise processes across multiple states, a phenomenon commonly observed in real datasets.

*Correspondence to: yiliu.wang@alleninstitute.org.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

More broadly, while desirable for interpretability, discreteness poses challenges in analyzing the physical world. One relevant manifestation is the difficulty of incorporating discrete factors into machine learning models: although gradient descent fuels spectacular successes, obtaining gradient estimates around such discrete factors is inherently problematic. The Gumbel distribution, a member of the extreme value distribution family [16, 17], offers a relaxation to produce "soft discrete" samples, where the approximation is controlled by a temperature parameter [20, 23]. Here, we adopt this approach to propose a dynamical model that approximates switching dynamics, is trained with gradient descent, and offers interpretable characterizations even when the parameter estimates deviate substantially.

The Gumbel-soft relaxation of states, the soft transition design of the dynamics, and the efficient inference algorithms together provide several advantages for analyzing complex time series. First, the model accommodates systems with mixed states and stochastic transitions. Second, the soft relaxation reduces spurious rapid switching, leading to more interpretable notions of states. Finally, the models are fast to train and generalize readily to unseen data. We validate our approach on benchmark simulations and two real-world datasets: Formula 1 race telemetry data [26] and the Caltech Mouse Social Interactions dataset (CalMS21) [28]. We observe that our implementation learns faster and produces more interpretable state estimates compared to competitive benchmarks.

1.1 Related work

Our model relates to state-space models such as autoregressive HMMs (AR-HMMs) and switching linear dynamical systems (SLDSs). AR-HMMs extend HMMs with autoregressive observations [21, 15], while SLDSs decompose dynamics into linear primitives [2]. Extensions have improved flexibility and inference, including recurrent SLDSs [22, 13], nonlinear variants [11], duration modeling [3, 27], structured online inference [18], and smooth transitions via Gaussian processes [19]. In contrast, GDM introduces smoothness directly in the state representation, allowing overlapping and continuous state evolution without additional duration variables or boundary assumptions. GDM also directly combines dynamical primitives and removes the stochastic continuous trajectory.

Recent studies emphasize preserving interpretability while increasing expressivity, often by decomposing time series into linear dynamical systems (LDSs). The Kalman VAE combines a variational auto-encoder with linear Gaussian state-space models and a dynamics-parameter network for time-varying weights [12]. Other approaches model transitions as time-varying mixtures of LDSs or introduce hierarchical structures to encourage sparse and smooth dynamics coefficients [24, 10]. While these improve robustness to noise, they remove the notion of discrete states and their recurrent relationships with the environment. More recently, TiDHy decomposes dynamics into orthogonal LDSs operating at different timescales [1].

The Gumbel-Softmax distribution has been explored as a differentiable reparameterization tool in SLDS-related models. Prior work has used it to approximate discrete dynamics weights, relax categorical states for gradient-based training, or replace marginalization in variational inference [12, 8, 11]. GDM instead uses the stochastic relaxed state to govern observations directly, without a separate stochastic continuous trajectory (see Appendix A).

GDM is not restricted to classical SLDS parameterizations and can incorporate expressive sequence models such as RNNs. Modern architectures—including neural differential equations, S4, transformers, and recurrent deep networks [9, 14, 29]—achieve strong performance in sequence modeling but are not designed to produce switching dynamical primitives. Integrating Gumbel-driven state dynamics with such architectures, for example by coupling attention or continuous-time models with learnable switching structure, is a promising direction for combining expressivity with interpretability.

1.2 Summary of contributions

Our contributions are summarized in the following points.

• We propose a Gumbel-driven dynamical system model defined over a relaxed-discrete state space that enables smooth, overlapping state transitions while retaining the interpretability of state-based dynamical primitives. The model is designed for non-stationary time series where regimes may overlap and/or evolve gradually.

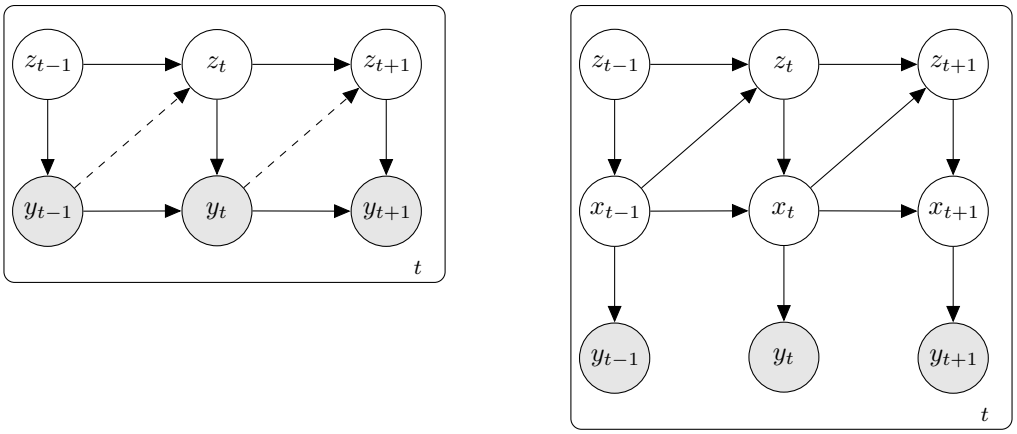


Figure 1: **Graphical model representation of two systems.** Left: 2-level GDM. Right: 3-level Mixture SLDS. Dashed lines denote dependencies that can be removed for equivalence, subject to noise assumptions (Appendix C).

• We formulate a fully differentiable generative model and variational posterior directly over relaxed states, enabling scalable training with standard gradient descent methods.

• We design an amortized inference network for states, enabling fast inference and immediate generalization to unseen sequences *without re-optimizing a per-sequence latent trajectory posterior*, in contrast to many existing methods.

• We evaluate performance using metrics that capture both fit and quality of the inferred states. GDM improves state recovery in soft-sticky NASCAR and yields interpretable dynamical primitives in two real-world datasets.

2 Model formulation

2.1 Gumbel-Softmax trick

The Gumbel–Softmax trick [20, 23] provides a continuous relaxation of discrete random variables, enabling gradient-based optimization. Given logits $\pi \in \mathbb{R}^K$ for a categorical distribution, we proceed as follows. Let $G(\mu, \beta)$ denote the Gumbel distribution with location μ and scale β [17]. We sample $g_i \sim G(0, 1)$ and form perturbed logits $\pi_i + g_i$. The maximum $\max_i\{g_i + \pi_i\}$ follows a Gumbel distribution with location $\log \sum_j \exp(\pi_j)$ and scale 1, and the maximizing index follows the categorical distribution (Gumbel-Max trick): $P(i = \arg\max_j(g_j + \pi_j)) = \frac{\exp(\pi_i)}{\sum_j \exp(\pi_j)}$.

As a member of the extreme-value family, Gumbel noise amplifies differences among competing logits, sharpening the winner-take-all behavior of the Gumbel–Max construction. A continuous relaxation replaces the argmax with a tempered softmax, yielding the Gumbel-Softmax (GS) distribution $z \sim \mathrm{softmax}\left(\frac{\pi+g}{\tau}\right)$, where τ controls the softness. As $\tau \to 0^+$, the softmax approaches the argmax and the GS distribution recovers the categorical distribution.

The Gumbel-Max construction is invariant to shifts in the location parameter μ, while the scale β controls the spread of the noise. Sampling $g \sim G(0, \beta)$ is equivalent to sampling $\tilde{g} \sim G(0, 1)$ and using $z \sim \mathrm{softmax}\left(\frac{\pi}{\tau} + \frac{\beta\tilde{g}}{\tau}\right)$. Without loss of generality, we set $\beta = 1$ and write $z \sim \mathrm{GS}(\pi, \tau)$. This provides a differentiable $q(z|\phi)$ with continuous samples generated from parameter-free Gumbel noise. In practice, we set τ to a moderate value to ensure stable gradient flow while accounting for uncertainty in state transitions. Due to the extreme-value behavior of the Gumbel distribution, GS samples remain close to one-hot under moderate temperatures, preserving the semantics of discrete states while enabling smooth optimization. In the Gumbel dynamical model introduced next, Gumbel noise represents unobserved competition among primitives, and observation noise captures residual variation after their dynamics are specified. Additional background is provided in Appendix B.

2.2 Gumbel dynamical model

We propose a new dynamic switching model to accommodate continuous Gumbel-Softmax state samples, the Gumbel Dynamical Model (GDM):

$$z_1 \sim \mathrm{GS}(\pi_1, \tau), \qquad z_t \mid z_{t-1}, y_{t-1} \sim \mathrm{GS}(\pi_t, \tau), \quad \pi_t = f_\theta(z_{t-1}, Fy_{t-1}), \quad t \geq 2, \tag{1}$$
$$y_1 \mid z_1 \sim \mathcal{N}(z_1 \cdot \mu, R), \qquad y_t \mid y_{t-1}, z_t \sim \mathcal{N}\big(\sum_k z_{t,k}(S_k F y_{t-1} + b_k), R_t\big), \qquad t \geq 2.$$

Here, π_1 is a learnable prior over states, μ is an observation prior, $S_k \in \mathbb{R}^{N \times D}$ captures state-dependent dynamics in the projected observation space, $F \in \mathbb{R}^{D \times N}$ projects observations to a low-dimensional latent space, and R_t models the observation covariance. Importantly, f_θ can be any feed-forward network parameterized by θ. As a simple and interpretable case, f_θ can take a linear recurrent form $f_\theta(z_{t-1}, Fy_{t-1}) = TFy_{t-1} + r$, where T is a learnable $K \times D$ transition matrix and r is a bias vector. To explicitly encourage persistence, a sticky variant mixes the logits with the previous soft state: $\pi_t = (1-\gamma)(TFy_{t-1} + r) + \gamma\, z_{t-1}$.

The Markov-1 assumption in the GDM can be relaxed to incorporate longer history. In this case, we parametrize the transition logits with an RNN: let h_t be the hidden state updated as $h_t = g(h_{t-1}, Fy_{t-1})$ where g is a recurrent architecture such as a GRU. We then define the transition logits as $\pi_t = \mathrm{FNN}(z_{t-1}, h_t)$. While the state dynamics become non-linear, the soft states z_t still correspond to interpretable dynamical motifs, preserving the interpretability of the model.

The GDM can be a highly flexible modeling tool. At the temperature limit, the linear GDM can be seen as an input-driven ARHMM with structured, low-dimensional dynamics, which makes exact inference tractable without requiring variational methods. At the other extreme, the model has a natural extension as a deep GDM, in which multiple layers of GDM are composed to increase expressivity: $h_t^{(l)} = \sum_k z_{t,k}^l A_k^{(l)} h_{t-1}^{(l)} + W^{(l)} h_t^{(l-1)}$ where $h_t^{(1)} = x_t$ denotes the latent projected observation (i.e., $x_t = \mathbb{E}[Fy_t \mid z_{\leq t}]$; see Appendix C for details). The standard GDM corresponds to the single-layer case. This formulation highlights that GDM spans a spectrum from interpretable linear switching models to more expressive hierarchical dynamical systems.

2.3 Relationship to SLDS

GDM is closely related to SLDS, but differs in how stochasticity is modeled. In particular, GDM places stochasticity in the relaxed discrete states via Gumbel noise, rather than in a Gaussian latent trajectory. This leads to a 2-level formulation where state dynamics directly govern observations without introducing additional latent noise sources.

A related 3-level formulation can be derived by introducing latent projected observations $x_t = \mathbb{E}[Fy_t \mid z_{\leq t}]$ for $t \geq 1$, where the expectation is taken conditional on all past states. Replacing Fy_{t-1} in the GDM with x_{t-1} yields a two-level GDM system, which is equivalent to

$$z_1 \sim \mathrm{GS}(\pi_1, \tau), \quad z_t \mid z_{t-1}, x_{t-1} \sim \mathrm{GS}(\pi_t, \tau), \;\; \pi_t = f(z_{t-1}, x_{t-1}), \quad t \geq 2, \tag{2}$$
$$x_1 = z_1 \cdot \mu, \quad x_t \mid x_{t-1}, z_t = \sum_k z_{t,k}\,(A_k x_{t-1} + c_k), \quad t \geq 2,$$

and $y_t \mid x_t \sim \mathcal{N}(Cx_t, Q_t)$, for $t \geq 1$. Here, the continuous latent trajectory x_t at time t is determined by a mixture of dynamics over the soft states z_t. Importantly, x_t is deterministic given z, and is introduced to facilitate interpretation. At each time t, x_t can be viewed as the expected projection of y_t. Uncertainty in the system is thus captured solely by the Gumbel noise on z and the Gaussian noise on y. Figure 1 illustrates the graphical models of the two systems, highlighting their relationships and dependencies. Note that exact distributional equivalence requires allowing temporally correlated noise, while under homogeneous, conditionally independent noise, deterministic dynamics agree but noise processes differ. A proof of system equivalence is provided in Appendix C.1.

More generally, one could allow additional noise in the latent trajectory x by introducing state-dependent covariances. This results in a mixture version of the standard recurrent SLDS with Gumbel state dynamics. Although more expressive in principle, the trajectory dynamics x and the state dynamics z compete to explain the data, and inference becomes more expensive as a flexible posterior is required to capture their intricate dependencies. For completeness, we discuss variational inference for this 3-level mixture model in Appendix C.2.

2.4 Discussion on interpretability

Interpretability of GDM arises from two complementary aspects. First, GDM is a linear dynamical model, meaning that the evolution of latent dynamics is governed by a collection of linear dynamical primitives. Each primitive corresponds to a time-invariant system with well-defined analytical properties, making the learned dynamics directly analyzable. Second, GDM improves state estimation by removing the trade-off between stochasticity in the continuous latent trajectory x and stochasticity in the switching state z, thereby enhancing interpretability in practice. To quantify interpretability, we introduce the following metric that measures the quality of inferred states at the state level.

Inferred State Accuracy. Let $\{\zeta_t\}_{t=1}^T$, $\zeta_t \in \{1, \ldots, C\}$, denote the ground-truth (or expert-labeled) discrete states, where C is the number of annotation categories, and let $\{z_t\}_{t=1}^T$, with $z_t \in \Delta^{K-1}$, denote the inferred states, where Δ^{K-1} is the $(K-1)$-simplex. In particular, discrete inferred states are represented as one-hot vectors in Δ^{K-1}. We train a k-nearest neighbor (k-NN) classifier $f_{\mathrm{KNN}} : \Delta^{K-1} \to \{1, \ldots, C\}$ on the training set by mapping inferred states z_t to ground-truth ζ_t. For test data $\mathcal{D}_{\mathrm{test}}$, predictions are obtained as $\hat{\zeta}_t = f_{\mathrm{KNN}}(z_t), \quad t \in \mathcal{D}_{\mathrm{test}}$. The *Inferred State Accuracy* is then defined as $\mathrm{Acc}_{\mathrm{state}} = \frac{1}{|\mathcal{D}_{\mathrm{test}}|} \sum_{t \in \mathcal{D}_{\mathrm{test}}} \mathbf{1}\left[\hat{\zeta}_t = \zeta_t\right]$.

When the underlying ground truth ζ_t is manually obtained by human annotators, $\mathrm{Acc}_{\mathrm{state}}$ measures annotation agreement, providing external validation of the learned dynamical primitives. This metric depends on the representation of latent states. Models with discrete one-hot states map to a finite set of points in the simplex, limiting their ability to represent a larger number of ground-truth classes. In contrast, continuous state representations can induce a finer partition of the simplex, enabling more flexible alignment with annotations. We therefore interpret $\mathrm{Acc}_{\mathrm{state}}$ as reflecting both state interpretability and representational flexibility. Importantly, it serves only as a proxy: interpretability in GDM primarily stems from the recovery of stable, reusable dynamical primitives, while agreement with annotations provides external validation of their relevance.

Finally, we note that interpretability is also related to identifiability, which asks whether the latent representation can be recovered from observed data in a statistically well-posed way. For GDM, while the projection matrix F and dynamic matrices S_k are identifiable only up to an invertible linear transformation, the remaining parameters are identifiable up to permutations [4] in the limiting case $\tau \to 0$. Note that finite-temperature identifiability remains open (Appendix D).

3 Model Inference

Due to the continuous nature of states, GDM can be trained using standard gradient descent. To infer GDM, we use BBVI [25] with Gumbel-Softmax samples (GS-BBVI): we define a variational distribution $q(z)$, sample soft states z from $q(z)$, and compute unbiased samples of the ELBO gradient.

ELBO. The ELBO for the GDM can be written as follows:

$$
\begin{aligned}
&\log p_\theta(y_{1:T}) \geq \mathbb{E}_{q(z)}\left[\log p(y, z) - \log q(z)\right] \\
=&\mathbb{E}_{q(z)}\left[\log p(z_1) + \log p(y_1|z_1) + \sum_{t=2}^{T} \log p(y_t|y_{t-1}, z_t) + \sum_{t=2}^{T} \log p(z_t|z_{t-1}, y_{t-1})\right] \\
&- \mathbb{E}_{q(z)}\left[\log q(z_{1:T})\right]
\end{aligned}
$$

3.1 Variational posteriors

We approximate the posterior over latent states with an amortized variational distribution $q_\phi(z_{1:T} \mid y_{1:T})$, parameterized by a neural network that maps observations to Gumbel-Softmax logits. Specifically, $q_\phi(z_{1:T} \mid y_{1:T}) = q_\phi(z_1 \mid y_{1:T}) \prod_{t=2}^{T} q_\phi(z_t \mid z_{<t}, y_{1:T})$, where each z_t is a continuous Gumbel-Softmax random variable with logit π'_t and temperature τ.

Since $z_{1:T}$ are continuous, we do not define a discrete transition matrix as in the categorical case. Instead, we directly parameterize the logits $\pi'_{1:T}$ via an inference network $g_\phi(y_{1:T})$, which shares a similar structure to the transition network f_θ. In principle, g_ϕ can be more expressive than f_θ, improving posterior approximation and training speed. However, an overly expressive g_ϕ may

compensate for limitations of f_θ, yielding good fit but less interpretable dynamics. For this reason, we keep their structures aligned.

Concretely, if f_θ is linear, g_ϕ can also be linear, e.g., $\pi'_t = Wy_t + b$. A sticky component depending on z_{t-1} can be added to encourage persistence, $\pi'_t = Wy_t + Bz_{t-1} + b$, with z_1 drawn from a Gumbel-Softmax distribution parameterized by learnable prior logits π'_1. This yields a Markovian factorization $q(z_{1:T}|y_{1:T}) = q(z_1 \mid y_1) \prod_{t=2}^{T} q(z_t \mid z_{t-1}, y_t)$. If f_θ is recurrent, we instead parameterize g_ϕ with a bidirectional RNN or Transformer so that π'_t depends on both past and future observations, e.g., letting $e_{1:T} = \text{BiGRU}(y_{1:T})$, we set $\pi'_t = \text{FNN}(z_{t-1}, e_t)$. This yields a more expressive posterior that leverages temporal context to infer z_t.

Thanks to the Gumbel-Softmax reparameterization, we can sample $q(z)$ sequentially in a differentiable manner. The temperature τ controls the smoothness of state transitions. Empirically, GDM is largely insensitive to τ over a broad range $\tau \in (0.5, 1)$, likely due to the extreme-value nature of the Gumbel distribution, which concentrates samples near simplex corners. In practice, fixing $\tau \approx 1$ provides stable optimization and interpretable state recovery. Annealing from a higher temperature can further improve robustness, although accurate state recovery ultimately depends on learning the transition parameters rather than relying on low temperature alone.

Importantly, amortized variational inference with differentiable $q(z)$ is a key advantage of GDM. The inference network learns a reusable mapping from observations to state logits, allowing new data to be processed directly without re-optimization. This contrasts with many existing models, which typically require re-optimizing a posterior for the latent trajectory on each new dataset.

3.2 Smoothing and prediction

Once the variational posterior and model parameters are trained, the inferred system can be used for smoothing current observations, evaluating quality of fit, predicting future steps, and generating new observations. Given a time series $y_1, \ldots, y_T$ of length T, we first obtain samples $z_1, \ldots, z_T$ from the variational posterior. Smoothed observations $\hat{y}_1, \ldots, \hat{y}_T$ are then computed based on the sampled states and past observations, providing a measure of reconstruction quality.

To predict future steps, we apply the learned transition model to generate next-step states $\hat{z}_2, \ldots, \hat{z}_T$ from the sampled states $z_1, \ldots, z_{T-1}$ and preceding observations $y_1, \ldots, y_{T-1}$. These predicted states are then used to generate corresponding next-step observations $\hat{y}_2, \ldots, \hat{y}_T$. The predicted observations can be recursively fed back into the transition model, enabling multi-step-ahead predictions. We note that an analogous procedure applies to the 3-level mixture formulation. Instead of propagating predicted observations, we propagate the inferred latent trajectory $\hat{x}_2, \ldots, \hat{x}_T$, which serves as input to the state transition function.

Due to stochasticity in the latent states, multi-step predictions naturally yield uncertainty envelopes that reflect transition ambiguity. A k-step-ahead prediction for a series $y_1, \ldots, y_T$ is equivalent to producing k future observations at each of the T possible starting points. Because of the injected Gumbel noise in the latent states z, prediction trajectories may diverge after only a few steps, particularly at higher temperatures τ. These divergent possibilities form a prediction envelope, whose width increases at points of greater transition uncertainty. This widening envelope corresponds naturally to the unpredictability observed in real-world dynamical systems. We will further illustrate this concept via simulation examples in section 4.

4 Experiments

We evaluate GDM on both simulated data and two real-world datasets. We begin with a standard, deterministic simulated example, then introduce soft, sticky, and stochastic transitions. We further evaluate the model on two real-world datasets that feature multiple and highly unpredictable transitions. To assess model performance, we use two complementary metrics. At the observation level, we compute the coefficient of determination R^2 between the smoothed and true observations, which quantifies the quality of fit. At the state level, we use the inferred state accuracy, introduced in Section 2, that measures the agreement between inferred and ground-truth states. We also report per-iteration wall-clock time across methods and datasets in Appendix E. Our code is available at: `https://github.com/yiliuw/GDM/`.

4.1 From deterministic to uncertain: Synthetic NASCAR dataset

The synthetic NASCAR dataset [22] emulates cars going around a track, with four states: two for straightaways and two for semicircular turns. In the standard setting, the state evolution is nearly deterministic given the previous trajectory, resulting in a nearly fixed path from a given starting point (see Appendix F for details).

We also consider a more realistic variant with soft, stochastic transitions by replacing the recurrent relationship in Eqn. (8) with a soft sticky form:

$$z_t|x_{t-1} \sim \mathrm{GS}(\pi_t, \tau), \quad \pi_t = c(1-\gamma)(Sx_{t-1}+s) + \gamma z_{t-1}, \quad t \geq 2. \tag{3}$$

Here, c controls transition softness and γ controls stickiness. Decreasing c, increasing γ, or raising τ leads to more stochastic transitions (Figure 2A).

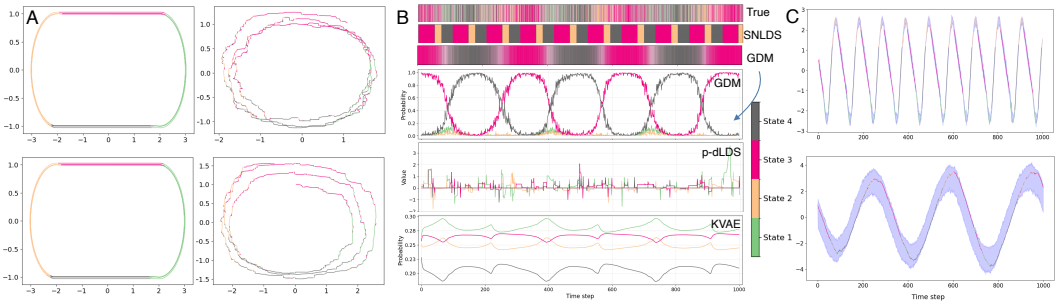


Figure 2: A. Standard and soft sticky NASCAR tracks. Two trials are generated with the same set of parameters ($T = 1000$, and $K = 4$ for both cases, $c = 0.02$ and $\gamma = 0.25$ for the soft-sticky case only). Compared to the standard NASCAR, soft sticky NASCAR introduces greater transition uncertainty. B. True states and exemplar inferred states from GDM ($\tau = 0.99$), SNLDS, p-dLDS and KVAE. For each method, the panel shows the inferred state responsibilities over time: probabilities for GDM and KVAE, and scalar-valued dynamic coefficients for p-dLDS. Colored curves correspond to different dynamic primitives. Results shown are representative of 10 random seeds; full variability is reported in Table 1. C. Inferred 1-step-ahead prediction ranges for the first dimension of NASCAR observations. The top panel shows the standard model, and the bottom panel shows the soft sticky model, with a much wider uncertainty range. Shaded regions indicate ±3 standard deviations around the predicted mean, estimated from 100 Monte Carlo samples per model.

We benchmark against ARHMM, SLDS, rSLDS variants, p-dLDS, KVAE, and SNLDS. All models achieve near-perfect train R^2 (≈0.99) on both datasets. However, in the soft-sticky setting, most baselines require re-optimizing variational posteriors to achieve good test R^2 (otherwise the test R^2 is simply 0.8, i.e., the difference between the top and bottom trials), whereas GDM generalizes directly due to amortized inference (Section 3). KVAE and SNLDS also generalize without retraining, but rely on additional architectures or marginalization strategies that complicate state interpretability. Full training setups are provided in Appendix E.

Figure 2B compares inferred states across methods. GDM recovers the dominant states and represents the remaining structure through soft combinations. In contrast, baseline models struggle to capture stable and interpretable state structure in this regime: SLDS, rSLDS, and SNLDS often exhibit state collapse; KVAE produces noisy mixtures; and ARHMM and p-dLDS fail to reproduce the correct oscillatory patterns. These limitations are consistent with known challenges of discrete-switching models under soft and stochastic transitions.

Table 1 reports the average state quality measured by mapping inferred states to hard-thresholded ground-truth states on the test trial. For the standard NASCAR data, rSLDS with recurrent only transitions achieves the top performance, while our model outperforms all the benchmarks in the soft sticky NASCAR case. Our model treats the observations as inherently stochastic, as discussed in section 3. While this uncertainty aspect is not advantageous in the standard NASCAR case, it allows the model to generalize better in the soft-sticky NASCAR case. This contrast highlights GDM's advantage when transitions are gradual, stochastic, or overlapping. Indeed, GDM correctly identifies that the soft sticky case exhibits greater uncertainty. This is illustrated by the one-step-ahead

	ARHMM	SLDS (S)	rSLDS (S)	rSLDS (R)	p-dLDS	KVAE (M)	SNLDS (C)	GDM
Standard	70 ± 13	82 ± 13	76 ± 10	96 ± 06	74 ± 01	67 ± 16	64 ± 02	88 ± 10
Soft-sticky	32 ± 02	32 ± 02	33 ± 01	43 ± 09	34 ± 02	50 ± 11	45 ± 05	**70** ± 03

Table 1: Comparison of $100\times$ inferred state accuracy on the standard and soft-sticky NASCAR datasets (mean $\pm$ SD). "S" denotes sticky variants, "R" denotes recurrent-only variants, "M" denotes the KVAE with MLP encoders, and "C" denotes the SNLDS with collapsed variational inference. Each model is trained and evaluated 10 times with different random seeds. On soft-sticky NASCAR, GDM remains the strongest method under top-3 and best-run reporting conventions. In particular, its lowest-scoring run (0.640) exceeds every baseline's best run (highest: 0.589 for rSLDS(R)).

prediction envelopes in Figure 2C. While most one-step-ahead observations fall inside the envelopes for both cases, the envelope is clearly wider in the soft sticky case.

As further ablations, we compare against the categorical limiting case and test whether the gains come from richer state representations only. We compare with an input-driven recurrent ARHMM with the same K, observation family, split, and evaluation. This is the direct categorical limit of linear GDM: both use the preceding observation to determine next-state logits, but recurrent ARHMM selects one categorical regime whereas GDM permits stochastic relaxed combinations. The gain from 0.33 to 0.70 state accuracy supports the value of stochastic mixtures over categorical switching. GDM also has the highest mean Hungarian accuracy, ARI, and NMI after argmax-hardening inferred and ground-truth states (Table 2); the advantage thus persists without continuous features for the classifier.

Model	State accuracy	Hungarian	ARI	NMI
GDM	0.70 ± 0.03	0.55 ± 0.13	0.30 ± 0.13	0.32 ± 0.09
Recurrent ARHMM	0.33 ± 0.02	0.35 ± 0.04	0.05 ± 0.07	0.07 ± 0.06
rSLDS(R)	0.43 ± 0.09	0.49 ± 0.06	0.15 ± 0.08	0.21 ± 0.07

Table 2: Soft-sticky NASCAR controls (mean $\pm$ SD). State accuracy uses the original k-NN metric; the remaining metrics use argmax-hardened states.

4.2 From simple states to more states: F1 dataset

The NASCAR dataset described above represents a simple track with four synthetic segments. Next, we consider a more complex and realistic example: the Formula One (F1) World Championship racetracks. We use the FastF1 package to retrieve telemetry data from past F1 sessions, including trajectory, lap times, and corner counts.

We consider two complementary versions of the F1 experiment, corresponding to different notions of "state". The first uses positional data as input, where we study individual F1 circuits separately, with training and testing performed on the same track using 1st- and 5th-place trajectories. In this setting, ground-truth states are defined by officially annotated corners, so similar maneuvers at different spatial locations are treated as distinct states. As shown in Figure 3A, the Shanghai International Circuit has 16 corners. The second version uses velocity as input and is trained across multiple race tracks, thereby emphasizing shared dynamical structure across circuits rather than spatial segmentation. In this setting, no ground-truth labels are available, and inferred states are expected to reflect common local dynamics (e.g., acceleration patterns).

For the position-based experiment, we compare inferred trajectories on the Shanghai circuit at $K = 8$, selected via held-out likelihood. GDM identifies a small number of dominant states and represents the remaining structure through soft combinations of these primitives. As shown in Figure 3C, the learned dynamics consist of anisotropic shear and rotation: the former collapses trajectories onto low-dimensional directions (dominant eigen-directions), producing piecewise-linear segments and corner-like transitions under regime switching, while the latter induces smooth turning behavior. In contrast, ARHMM and rSLDS exhibit more frequent switching and less stable state assignments, particularly around corner transitions (Figure 3B and Figure 5). SNLDS mitigates rapid switching relative to SLDS due to its increased flexibility, but this comes at the cost of slightly distorted latent

trajectories. Moreover, these models rely on discrete switching and have difficulty capturing smooth or overlapping regimes, while incurring substantially higher computational cost (see Appendix E). Notably, all models achieve comparable training and testing fit, but discrete-switching models often do so at the expense of state quality when K is small. Appendix G provides additional results corresponding to various choices of K.

For the velocity-based experiment, we train on 23 tracks from the 2022 season and test on a held-out track (Shanghai). Due to the computational cost of nonlinear baselines, we compare GDM with linear baselines in this setting. Here, GDM groups together segments with similar local dynamics (e.g., curvature or acceleration), even when they occur at different spatial locations, whereas both ARHMM and rSLDS fail to recover such shared structure. As shown in Figure 3D, eigenanalysis of the dynamic matrices reveals distinct linear flow structures with different directional contraction and coupling, which manifest as direction- and curvature-dependent acceleration behaviors across states. These experiments highlight that the notion of "state" depends on the representation: position-based inputs emphasize spatial segmentation, while velocity-based inputs reveal reusable dynamical motifs.

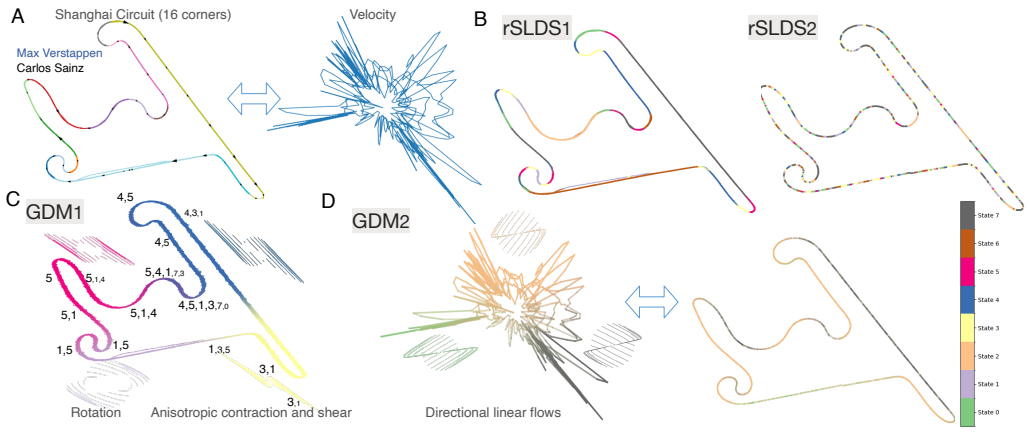


Figure 3: A. Two versions of the F1 dataset with different input observations. *Position-based*: experiments on individual tracks (Suzuka and Shanghai), with training and testing performed on the same track using 1st (blue) and 5th-place (black) trajectories. *Velocity-based*: experiments across 24 tracks, with training on 23 tracks and testing on a held-out track (Shanghai). B-D. Example inferred trajectories for rSLDS and GDM on the Shanghai Circuit (representative of 5 random seeds). Results for other models are shown in Appendix G. For GDM_1, we annotate each segment with state IDs that exceed 1% weight in at least 20% of the time steps within the corresponding expert-labeled segment. State IDs are ordered by presence ratio, and marker sizes reflect their weights (see Appendix G for state usage). For GDM_2, the velocity-based setting yields consistent latent "velocity states" shared across circuits, from which the track trajectory is reconstructed. For each dominant mode, we visualize the phase plot of the corresponding dynamical system for interpretability, initialized from 24 points and evolved for 20 steps.

4.3 Uncertainty and multiple states: CalMS21 dataset

Finally, we apply our method to study mouse social behavior using the first task in the open CalMS21 dataset [28]. The dataset consists of location data for two mice interacting in a cage from multiple trials (89 trials, split into 70 train and 19 test) over 5 years. Each mouse is labeled with 7 keypoints, corresponding to the nose, ears, base of neck, hips, and tail (Figure 4A). As there are 14 keypoints with x, y values per frame, the observation dimension is 28. Importantly, this dataset is expert labeled. All frames in the 89 trials are manually labelled by one expert for four distinct social behaviors (attack, investigation, mount and other).

Human annotations provide one label per frame, while the underlying motion may vary continuously or combine multiple local dynamics. We train our models on the 70 training trials and test them on the 19 test trials, fixing the state dimension as $K = 5$, as selected by GDM based on held-out ELBO without expert labels. Here, K counts dynamical primitives, not annotation categories. In practice,

initializing with a larger K can be beneficial, as redundant states typically become auxiliary and contribute minimally, consistent with Occam's razor.

Figure 4 summarizes the performance of our models (linear-sticky and RNN-based GDMs) alongside the rSLDS benchmark. Both GDM variants achieve higher training and testing accuracy than rSLDS for nearly all trials in this dataset. Consistent with our findings on the F1 dataset, our models also yield substantially higher inferred-state accuracy across all test trials. A key observation emerges when comparing the two GDM variants. The RNN-based GDM model achieves the highest observation-level accuracy for both training and test sets, reflecting the benefit of incorporating nonlinear recurrent functions into both the generative model and the variational posterior. However, its inferred-state accuracy is lower than that of the linear–sticky version. This underlines a key trade-off: adding expressive RNN/ bidirectional RNN components improves predictive accuracy but comes at the cost of decreased interpretability in the latent state dynamics.

GDM identifies three dominant modes with distinct dynamical characteristics, revealed through eigen-analysis and linked to interpretable patterns of social interaction. We illustrate this on an exemplar training session (Figure 4C), where two modes align closely with expert-defined behaviors, while the third (State 3) captures an additional behavioral motif. Importantly, these inferred states do not contradict the annotations; rather, they refine them by decomposing coarse labels into more coherent dynamical patterns. On unseen data, the inferred states can be mapped back to expert labels, while simultaneously providing finer-grained dynamical structure within ambiguous regimes.

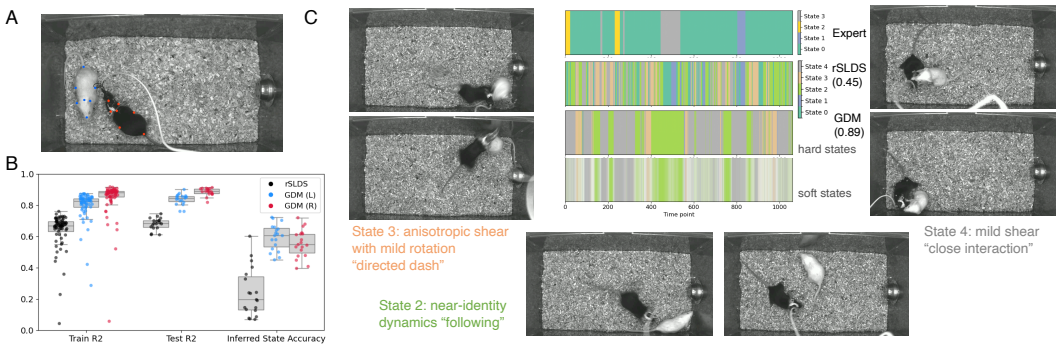


Figure 4: A. Example frame from the CalMS21 data. Seven anatomically defined keypoints are labeled on the body of each mouse. Expert annotations refer to behaviors initiated by the black mouse. B. Comparison of train R^2, test R^2, and inferred state accuracy between our models and the benchmark model rSLDS. "L" denotes the linear sticky GDM, and "R" denotes the RNN-based GDM. C. Expert-labeled states and inferred states from GDM and rSLDS for Trial 34 (the shortest trial containing all states). Two representative frames are shown for each GDM dominant mode.

5 Conclusion

In this work, we proposed the Gumbel Dynamical Model (GDM), a switching dynamical system defined over relaxed discrete states. By replacing hard switching with Gumbel-driven continuous states, GDM captures smooth, stochastic transitions while preserving interpretable dynamical primitives and enabling efficient gradient-based training. Across synthetic and real-world datasets, GDM improves state stability and alignment with annotations in regimes where discrete-switching models struggle.

Limitations Finite-temperature identifiability remains open. In addition, greater model expressivity can reduce interpretability. Our experiments use low- to moderate-dimensional vector observations; end-to-end learning from raw images or videos was not evaluated. A controlled non-Gumbel soft-transition simulator and comparisons with alternative simplex distributions or deterministic softmax also remain untested. Finally, the impact of Gumbel-related parameters on model behavior is not yet fully characterized and warrants further study.

Acknowledgments and disclosures

We wish to thank the founder of the Allen Institute, P.G. Allen, for his vision, encouragement, and support. We wish to thank CiCi Zheng, Ian Convy, Adrienne Fairhall, Christof Koch and Karel Svoboda for useful discussions. Y.W. and T.D.K. were supported by the Shanahan Family Foundation Fellowship at the Interface of Data and Neuroscience at the Allen Institute and the University of Washington, supported in part by the Allen Institute.

The authors declare no competing interests.

References

[1] E. T. Abe and B. W. Brunton. Tidhy: Timescale demixing via hypernetworks to learn simultaneous dynamics from mixed observations. *bioRxiv*, 2025.

[2] G. Ackerson and K. Fu. On state estimation in switching environments. *IEEE transactions on automatic control*, 15(1):10–17, 1970.

[3] A. F. Ansari, K. Benidis, R. Kurle, A. C. Turkmen, H. Soh, A. J. Smola, B. Wang, and T. Januschowski. Deep explicit duration switching models for time series. *Advances in Neural Information Processing Systems*, 34:29949–29961, 2021.

[4] C. Balsells-Rodas, Y. Wang, and Y. Li. On the identifiability of switching dynamical systems. *arXiv preprint arXiv:2305.15925*, 2023.

[5] C. Balsells-Rodas, X. Sumba, T. Narendra, R. Tu, G. Schweikert, H. Kjellstrom, and Y. Li. Causal discovery from conditionally stationary time series. In A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff, and J. Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 2715–2741. PMLR, 13–19 Jul 2025. URL `https://proceedings.mlr.press/v267/balsells-rodas25a.html`.

[6] D. Barber. Expectation correction for smoothed inference in switching linear dynamical systems. *Journal of Machine Learning Research*, 7(11), 2006.

[7] O. Barndorff-Nielsen. Identifiability of mixtures of exponential families. *Journal of Mathematical Analysis and Applications*, 12(1):115–121, 1965.

[8] P. Becker-Ehmck, J. Peters, and P. Van Der Smagt. Switching linear dynamics for variational bayes filtering. In *International conference on machine learning*, pages 553–562. PMLR, 2019.

[9] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[10] Y. Chen, N. Mudrik, K. A. Johnsen, S. Alagapan, A. S. Charles, and C. Rozell. Probabilistic decomposed linear dynamical systems for robust discovery of latent neural dynamics. *Advances in Neural Information Processing Systems*, 37:104443–104470, 2024.

[11] Z. Dong, B. Seybold, K. Murphy, and H. Bui. Collapsed amortized variational inference for switching nonlinear dynamical systems. In *International Conference on Machine Learning*, pages 2638–2647. PMLR, 2020.

[12] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. *Advances in neural information processing systems*, 30, 2017.

[13] J. Glaser, M. Whiteway, J. P. Cunningham, L. Paninski, and S. Linderman. Recurrent switching dynamical systems models for multiple interacting neural populations. *Advances in neural information processing systems*, 33:14867–14878, 2020.

[14] A. Gu, K. Goel, and C. Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

[15] X. Guan, R. Raich, and W.-K. Wong. Efficient multi-instance learning for activity recognition from time series data using an auto-regressive hidden markov model. In *International Conference on Machine Learning*, pages 2330–2339. PMLR, 2016.

[16] E. J. Gumbel. Les valeurs extrêmes des distributions statistiques. In *Annales de l'institut Henri Poincaré*, volume 5, pages 115–158, 1935.

[17] E. J. Gumbel. The return period of flood flows. *The annals of mathematical statistics*, 12(2): 163–190, 1941.

[18] H. Hashempoor. Structured temporal inference in state-space models. In *Proceedings of The 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pages 2602–2610. PMLR, 2026. URL `https://proceedings.mlr.press/v300/hashempoor26a.html`.

[19] A. Hu, D. M. Zoltowski, A. Nair, D. Anderson, L. Duncker, and S. W. Linderman. Modeling latent neural dynamics with gaussian process switching linear dynamical systems. *Advances in Neural Information Processing Systems*, 2024.

[20] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[21] B.-H. Juang and L. Rabiner. Mixture autoregressive hidden markov models for speech signals. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(6):1404–1413, 1985.

[22] S. Linderman, M. Johnson, A. Miller, R. Adams, D. Blei, and L. Paninski. Bayesian learning and inference in recurrent switching linear dynamical systems. In *Artificial intelligence and statistics*, pages 914–922. PMLR, 2017.

[23] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[24] N. Mudrik, Y. Chen, E. Yezerets, C. J. Rozell, and A. S. Charles. Decomposed linear dynamical systems (dlds) for learning the latent components of neural dynamics. *Journal of Machine Learning Research*, 25(59):1–44, 2024.

[25] R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In *Artificial intelligence and statistics*, pages 814–822. PMLR, 2014.

[26] P. Schaefer. FastF1. `https://github.com/theOehrly/Fast-F1`, 2020. Accessed: 2025-09-12.

[27] M. Słupiński. Bayesian inference in recurrent explicit duration switching linear dynamical systems. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pages 100–108. PMLR, 2025. URL `https://proceedings.mlr.press/v258/slupinski25a.html`.

[28] J. J. Sun, T. Karigo, D. Chakraborty, S. P. Mohanty, B. Wild, Q. Sun, C. Chen, D. J. Anderson, P. Perona, Y. Yue, et al. The multi-agent behavior dataset: Mouse dyadic social interactions. *Advances in neural information processing systems*, 2021(DB1):1, 2021.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[30] A. B. Wiltschko, M. J. Johnson, G. Iurilli, R. E. Peterson, J. M. Katon, S. L. Pashkovski, V. E. Abraira, R. P. Adams, and S. R. Datta. Mapping sub-second structure in mouse behavior. *Neuron*, 88(6):1121–1135, 2015.

[31] D. Zoltowski, J. Pillow, and S. Linderman. A general recurrent state space framework for modeling neural dynamics during decision-making. In *International Conference on Machine Learning*, pages 11680–11691. PMLR, 2020.

A Comparison to Gumbel-Softmax line of work

In this section, we provide detailed comparisons to the prior studies using Gumbel-Softmax in the literature of switching linear dynamical systems in terms of core modeling assumptions, dependency structure, the role of the Gumbel distribution, and inference compatibility. These approaches retain a separate stochastic continuous trajectory, whereas GDM's relaxed state directly governs observations and remains compatible with standard amortized BBVI.

Comparison to KVAE GDM has a fundamentally different graphical structure from the KVAE proposed by [12]. As illustrated in Figure 1 of both papers, KVAE combines a VAE with a linear Gaussian state space model (LGSSM): observations are first mapped into a low-dimensional latent space by a deep neural network, and those latents are then explained by a soft mixture of LGSSMs. The dynamics-parameter network in KVAE (Section 3.3) corresponds conceptually to the dynamic-operator weighting in p-dLDS [10], where the model learns continuous weights over multiple linear dynamical components.

Crucially, KVAE imposes no structural prior, sparsity constraint, or regularization on these mixture weights, so the learned dynamics tend to be dense mixtures, not interpretable switches. This limitation is directly visible in our experiments: using the publicly released KVAE code, we evaluated the model on the NASCAR benchmark. Table 1 shows consistently low discrete-state inference accuracy for KVAE compared to our model. Moreover, Figure 2B illustrates an example dynamics-parameter network weighting derived from KVAE, in which the maximum state weights are below 0.30 throughout the time span.

The KVAE appendix also discusses approximately discrete dynamics weights using the Concrete distribution, while retaining the LGSSM structure. GDM instead directly governs observations through its stochastic relaxed state.

Comparison to relaxed SLDS Although GDM shares the use of the Gumbel distribution with relaxed SLDS proposed by [8], the resulting generative model is fundamentally different. [8] use Concrete (Gumbel-Softmax) distributions for the switching prior and posterior while retaining a separate stochastic Gaussian trajectory. Importantly, GDM directly links states to observations and does not blur latent dynamics with Gaussian noise. As the authors note below Eq. 7, their observation likelihood depends on the continuous latent trajectory rather than directly on the switching variable. We further show that removing the important observation-to-state dependency makes GDM equivalent to a three-level mixture SLDS with a deterministic intermediate layer, which is introduced purely for interpretability. Comparing the graphical representations in Figure 1 of both papers highlights these structural differences.

We note that the model of Becker-Ehmck et al. is conceptually similar to the prototype discussed in Appendix C.2. We explicitly analyzed its main weakness—the competition between continuous and discrete dynamics to explain the data—and proposed a BBVI-based solution for completeness. We also implemented this variant early in development but found it unsatisfactory — performing worse than GDM in both accuracy and speed, and scaling poorly to long, high-dimensional time series. Moreover, they use structured inference for both latent variables, whereas our approach supports joint sampling and avoids customized inference machinery.

Finally, Becker-Ehmck et al. evaluate predictive performance and the information encoded by switching variables. Our focus is on recovering interpretable mixtures of dynamical primitives, with inferred-state accuracy evaluated separately from observation fit.

Comparison to GS-SNLDS Table 1 evaluates SNLDS with collapsed inference, which marginalizes the categorical sequence; GS-SNLDS is a distinct variant discussed here. The Gumbel–Softmax SNLDS of [11] is conceptually and technically distinct from our work, despite superficial similarity. In their method, the Gumbel–Softmax relaxation appears only in the variational posterior as a stand-in for marginalizing discrete states. It does not define the switching dynamics, and the generative model remains a standard SNLDS. In contrast, GDM is a Gumbel-driven dynamical system in which Gumbel noise drives the switching process and determines the latent evolution. Moreover, while amortized inference is standard, the generative structure of GDM makes such amortization fundamentally simpler. In GDM, there is no need to approximate $q(x)$ and no latent-to-latent stochasticity that gradients must pass through. Because observations depend directly on z, gradients propagate

cleanly through the transition logits, making the model fully compatible with unmodified off-the-shelf amortized inference.

Additionally, we note that the GS-SNLDS variational posterior is structurally mismatched to its own generative model. It is mean-factorized so that the continuous latent posterior does not depend on the discrete state, even though in SNLDS the discrete state selects the transition dynamics. This breaks a core dependency in the model. This issue is reflected in their experimental results (Table 1): GS-SNLDS performs substantially worse than all other baselines including linear SLDS variants, despite SNLDS being strictly more expressive, empirically demonstrating that the proposed GDM is different in key aspects.

B Background

SLDS The standard SLDS model generates the observation y from the continuous latent trajectory x and the discrete latent state z. The discrete states $z \in \mathbb{R}^K$ can depend on the latent trajectory x,

$$z_t \sim \mathrm{Cat}(\pi_t), \quad \pi_t = f(z_{t-1}, x_{t-1})$$

where f can be linear or nonlinear. If the discrete state at time t only depends on the latent trajectory at time $t-1$, the model is called recurrent only.

The continuous latent state $x_t \in \mathbb{R}^D$ follows conditionally linear dynamics determined by state z_t,

$$x_t \sim \mathcal{N}(A_{z_t} x_{t-1} + b_{z_t}, Q_{z_t})$$

where $A \in \mathbb{R}^{K \times D \times D}$ are the dynamics matrices, $b \in \mathbb{R}^{K \times D}$ are the shifts, and $Q \in \mathbb{R}^{K \times D \times D}$ are the covariance matrices. K denotes the number of unique discrete states.

Finally, a linear Gaussian observation $y_t \in \mathbb{R}^N$ is generated from the corresponding latent state $x_t \in \mathbb{R}^D$,

$$y_t \sim \mathcal{N}(Cx_t + d, \sigma)$$

where $C \in \mathbb{R}^{N \times D}$ is the emission matrix.

While the variational Laplace expectation-maximization (vLEM) algorithm is a popular choice for inference [13, 31], it does not guarantee improvement in the evidence lower bound (ELBO) in the E-step because it relies on a second-order Taylor approximation around the mode of the posterior, which can be poor in high-dimensional or multimodal settings. On the other hand, general stochastic optimization-based variational inference methods such as Black-Box Variational Inference (BBVI) can in principle handle discrete latent variables via score-function estimators. However, in models like SLDS, these estimators typically suffer from high variance when applied to long discrete state sequences, making them difficult to use effectively in practice. In contrast, reparameterization-based gradients cannot be directly applied to discrete latent states, which motivates the use of continuous relaxations such as the Gumbel–Softmax.

BBVI BBVI uses Monte Carlo gradients to optimize the ELBO. For an SLDS with latent variables z, x and observation y,

$$\mathrm{ELBO} = \mathbb{E}_{q(x,z)} \left(\log p(y, x, z) - \log q_\phi(x, z) \right) \leq \log p_\theta(y)$$

To optimize the ELBO with stochastic optimization, consider the gradient of the ELBO as an expectation with respect to the variational distribution,

$$\nabla_\phi \mathrm{ELBO} = \mathbb{E}_{q(z,x)} \left[\nabla_\phi \log q(z, x|\phi) \left(\log p(y, x, z) - \log q(z, x|\phi) \right) \right]$$

Noisy unbiased samples of the ELBO gradient can be computed using Monte Carlo samples from $q(z, x)$.

$$\nabla_\phi \mathrm{ELBO} \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_\phi \log q(x_s, z_s|\phi) \left(\log p(y, x_s, z_s) - \log q(x_s, z_s|\phi) \right)$$

Note that the score function and sampling algorithms depend only on the variational distribution, not the underlying model. With samples from the variational distribution, the only requirement is the computation of the log joint $\log p(y, x_s, z_s)$.

C Relationship to SLDS

In GDM, the observation y_t at time step t feeds back into the state dynamics through the projection matrix F, such that Fy_t recovers the low-dimensional latent trajectory. In fact, GDM can be related to the family of switching linear dynamical systems (SLDS) by introducing a latent projected observation $x_t = \mathbb{E}[Fy_t \mid z_{\leq t}]$ for $t \geq 1$, where the expectation is taken conditional on all past states. In this section, we prove the equivalence of the 2-level and 3-level formulations and discuss an alternative variational inference method for the 3-level mixture model.

C.1 Proof of system equivalence

In this section, we derive the equivalence relationship between the mixture model and the 2-level GDM. Recall we defined the dependency-removed 2-level GDM as follows,

$$z_1 \sim \mathrm{GS}(\pi_1, \tau), \quad z_t|z_{t-1} \sim \mathrm{GS}(\pi_t, \tau), \text{ s.t. } \pi_t = f(z_{t-1}, \mathbb{E}[Fy_{t-1}|z_{\leq t-1}]), \quad t \geq 2 \tag{4}$$
$$y_1|z_1 \sim \mathcal{N}(\sum_k z_{1,k}\mu_k, R_t), \quad y_t|y_{t-1}, z_t \sim \mathcal{N}(\sum_k z_{t,k}(S_k F y_{t-1} + b_k), R_t), \quad t \geq 2$$

And we defined the 3-level mixture model as follows (see eqn. (2)),

$$z_1 \sim \mathrm{GS}(\pi_1, \tau), \quad z_t|z_{t-1}, x_{t-1} \sim \mathrm{GS}(\pi_t, \tau), \text{ s.t. } \pi_t = f(z_{t-1}, x_{t-1}), \quad t \geq 2 \tag{5}$$
$$x_1 = \sum_k z_{1,k}\mu_k, \quad x_t|x_{t-1}, z_t = \sum_k z_{t,k}(A_k x_{t-1} + c_k), \quad t \geq 2 \tag{6}$$
$$y_t|x_t \sim \mathcal{N}(Cx_t, Q_t), \quad t \geq 1 \tag{7}$$

Firstly, we derive the 3-level mixture model (2) from system 4. The state transition equation of model (2) follows from a straightforward substitution. To obtain eqn.(6), we consider

$$\mathbb{E}_{y_t|z_{\leq t}}(Fy_t|z_{\leq t}) = F\mathbb{E}_{y_{t-1}|z_{\leq t}}[\mathbb{E}_{y_t|y_{t-1}}(y_t|y_{t-1}, z_{\leq t})]$$

Splitting time steps before t into time steps before $t-1$ and time step t we have,

$$\begin{aligned}
\mathbb{E}_{y_t|z_{\leq t-1}, z_t}(Fy_t|z_{t-1}, z_t) &= \mathbb{E}_{y_{t-1}|z_{\leq t-1}, z_t} \sum_k z_{t,k} F(S_k(Fy_{t-1}) + b_k) \\
&= \sum_k z_{t,k} \mathbb{E}_{y_{t-1}|z_{\leq t-1}, z_t}(FS_k(Fy_{t-1}) + Fb_k) \\
&= \sum_k z_{t,k}(FS_k x_{t-1} + Fb_k)
\end{aligned}$$

The last line is derived from the definition $x_{t-1} = \mathbb{E}(Fy_{t-1}|z_{\leq t-1})$ and the fact that y_{t-1} and z_t are conditionally independent given z_{t-1}. Conditioning on x_{t-1} and z_t, $x_t = \mathbb{E}_{y_t|z_{\leq t}}(Fy_t|z_{\leq t})$ is equivalent to the LHS of eqn.(6), as x_{t-1} is fully determined by states before time step $t-1$. The RHS of the equation above can be put into RHS of eqn.(6) by setting $A_k = FS_k$, and $c_k = Fb_k$. Finally, to obtain eqn.(7), we consider the mean and variance of y_t. If we set $C = F^\dagger$, we have $\mathbb{E}(y_t|x_t) = Cx_t$. To obtain the variance, we consider

$$\begin{aligned}
Q_t = \mathrm{Var}(y_t|x_t) &= \mathbb{E}\mathrm{Var}(y_t|y_{t-1}, x_t, z_t) + \mathrm{Var}\mathbb{E}(y_t|y_{t-1}, x_t, z_t) \\
&= R_t + \mathrm{Var}(\sum_k z_{t,k}(S_k F y_{t-1} + b_k))
\end{aligned}$$

We can remove the dependency on x_t in both summation terms, since x_t is fixed given z_t and z_{t-1}, and y_t is independent of z_{t-1} given z_t. In practice, we can assume a diagonal covariance structure $R_t = \sigma I$.

Next, we show the reverse derivation from the mixture model to the GDM.

To obtain the Gumbel dynamics equation for the GDM, we consider

$$\mathbb{E}_{y_t|z_{\leq t}}(Fy_t|z_{\leq t}) = \mathbb{E}_{x_t|z_{\leq t}}\mathbb{E}_{y_t|x_t, z_{\leq t}}(Fy_t|x_t, z_{\leq t}) = \mathbb{E}_{y_t|x_t}(Fy_t|x_t)$$

The inner expectation reduces to $\mathbb{E}_{y_t|x_t}(Fy_t|x_t)$ as y_t is independent of z_t given x_t. The outer expectation can be removed as x_t is fully determined by states before time step t.

By eqn. (7), we know that

$$x_t = \mathbb{E}_{y_t|x_t}(Fy_t|x_t) = \mathbb{E}_{y_t|z_{\leq t}}(Fy_t|z_{\leq t})$$

where $F = C^\dagger$. This gives the Gumbel dynamics equation for the GDM by substituting $\mathbb{E}(Fy_{t-1}|z_{\leq t-1})$ in eqn. (5).

To derive the observation level for the GDM, we substitute eqn. (6) into eqn. (7). Specifically, we write $y_t = Cx_t + \epsilon$ where $\epsilon \sim \mathrm{N}(0, Q)$. Then we have,

$$\begin{aligned} y_t &= C\sum_k z_{t,k}(A_k x_{t-1} + c_k) + \epsilon \\ &= C\sum_k z_{t,k}(A_k(Fy_{t-1} - \tilde{\epsilon}) + c_k) + \epsilon \\ &= \sum_k z_{t,k}(CA_k Fy_{t-1} + Cc_k) + \epsilon - \sum_k z_{t,k} CA_k \tilde{\epsilon} \end{aligned}$$

where $\tilde{\epsilon} \sim \mathcal{N}(0, FQF^\intercal)$ is another Gaussian noise term. The second line comes from eqn. (7), as we have $Fy_{t-1} = x_{t-1} + \tilde{\epsilon}$ where $\tilde{\epsilon} \sim \mathcal{N}(0, FQF^\intercal)$. Therefore, if we set $S_k = CA_k$, $b_k = Cc_k$ and $R_t = Q + (\sum_k z_{t,k} CA_k) FQF^\intercal (\sum_j z_{t,j} A_j^\intercal C^\intercal)$, we recover the observation dynamics in GDM.

We note that this substitution introduces a composite noise term that depends on both current and previous Gaussian perturbations, yielding an effective moving-average structure in the observation noise. As a result, the covariance R_t becomes time-dependent even when Q is diagonal. In practice, however, we typically assume time-homogeneous and conditionally independent Gaussian noise (i.e., $R_t = R$ and $Q_t = Q$) when fitting models. Therefore, the equivalence is exact when allowing general, potentially time- and state-dependent covariance structures. Under the common restriction to conditionally independent, time-homogeneous noise, the two formulations remain equivalent in their deterministic dynamics, while differing in their induced noise processes.

C.2 Variational inference for 3-level mixture model

As discussed in the main text, inference for the general 3-level mixture model is more challenging as we need to define variational distributions for both the latent variables x and z. We define a joint variational distribution $q(x_{1:T}, z_{1:T})$ that captures dependencies between x and z, implemented through an alternating sampling scheme. For z, we define a conditional variational distribution that depends on the current estimate of x, i.e., $q(z_{1:T}|x_{1:T}) = q(z_1)\prod_{t=2}^T q(z_t|z_{t-1}, x_{t-1})$. For x, we introduce dependencies that span multiple time steps by assuming a Gaussian with block tri-diagonal precision for $x_{1:T}$.

$$q(x_{1:T}|z_{1:T}) = \mathcal{N}(x_{1:T}|\mu, \Sigma) = \mathcal{N}(x_{1:T}|J, h)$$

where J is the precision matrix, h is the linear potential, $\mu = J^{-1}h$ is the mean, and $\Sigma = J^{-1}$ is the inverse precision (covariance) matrix. It can be written as the following pairwise linear Gaussian dynamics,

$$q(x_{1:T}) \propto [\prod_{t=1}^{T-1} \mathcal{N}(x_{t+1}|A_t x_t + b_t, Q_t)] \cdot [\prod_{t=1}^{T} \mathcal{N}(x_t|m_t, R_t)]$$

Note that it is easier to work with the pairwise LDS structure as the precision matrix J can be efficiently inverted and sampled from. We assume that the transition parameters A_t, Q_t, and b_t are state-dependent, $A_t = A_{z_t}$, $b_t = b_{z_t}$, and $Q_t = Q_{z_t}$. Note that the parameters of this distribution depend on the sampled trajectory of z, inducing an implicit coupling between x and z.

Sampling mechanism Note that sequential sampling is feasible for z but not for x. Recall the standard way of sampling from $\mathcal{N}(\mu, \Sigma)$ as follows. If Σ has Cholesky decomposition $\Sigma = LL^\intercal$, then we can generate samples using $x = \mu + L\eta$ where $\eta \sim \mathcal{N}(0, I)$. In our case, we need z_t for all time steps t to compute linear potential h and precision matrix J. To sample from J, we solve two equations: $J\mu = h$ and $U^\intercal \tilde{x} = \eta$ where U is the Cholesky decomposition of J s.t. $J = UU^\intercal$. The final sample of x is the sum of μ and $\tilde{x}$.

To sample from $q(x, z)$, we use an alternating procedure. We first initialize $x_{1:T}$ from the observations $y_{1:T}$. Given x, we sample z sequentially as follows: 1) Sample z_1 from the GS distribution with ϕ_1;

2) Compute logits ϕ_t using the learnable transition function and sample z_t, for all $t \geq 2$. Given the sampled $z_{1:T}$, we then construct the Gaussian parameters J and h and sample $x_{1:T}$ from $q(x_{1:T}|z_{1:T})$ using the linear-Gaussian structure described above. Note that this alternating scheme defines a valid joint sampler without requiring an explicit factorization of $q(x, z)$ into mutually conditioning components.

Complete ELBO The ELBO for the 3-level mixture model is:

$$\begin{aligned}
\log p_\theta(\boldsymbol{y}_{1:T}) \geq & \mathbb{E}_{q(x,z)} \left[\log p(y, x, z) - \log q(x, z)\right] \\
= & \mathbb{E}_{q(x,z)} \left[\sum_{t=1}^{T} \log p(y_t|x_t) + \log p(x_1|z_1) + \sum_{t=2}^{T} \log p(x_t|x_{t-1}, z_t)\right. \\
& \left. + \log p(z_1) + \sum_{t=2}^{T} \log p(z_t|z_{t-1}, x_{t-1})\right] \\
& - \mathbb{E}_{q(x,z)} \left[\log q(x_{1:T}|z_{1:T}) + \log q(z_1) + \sum_{t=2}^{T} \log q(z_t|z_{t-1}, x_{t-1})\right]
\end{aligned}$$

D Identifiability considerations for GDM

In the limiting case $\tau \to 0$, the GDM has an equivalent formulation as a finite mixture model analogous to an AR-HMM. Following the same notations in [4], for finite horizon T, one can define a bijective path indexing function ψ that maps each $i \in \{1, \ldots, K^T\}$ to a set of states $z_{1:T}$. Then, the family of GDMs can be seen as a finite mixture over all possible discrete state paths.

Let $M_k = S_k F$ denote the product of the dynamic matrix S_k and projection matrix F. At the observation level, GDM then satisfies the *unique-indexing assumption* on Gaussian means and initial states used in [4]. By Theorem 3.2 in their paper, under these conditions, the family of GDMs is identifiable up to permutations. Importantly, this identifiability result does not require restricting the form of the state transitions, and arbitrary recurrence from the switches is allowed [5]. In the case that transition logits π_t depend only on the previous discrete state z_{t-1}, one can uniquely recover the transition matrix.

For $\tau > 0$, identifiability becomes more subtle. The continuous GS relaxation means that latent state z_t at each time step t takes values on the simplex Δ^{K-1}, so GDM is no longer a finite mixture but rather behaves like an infinite mixture over continuous paths. The considerations above rely heavily on the use of finite mixture modeling techniques, and characterizing identifiability in the non-limiting regime remains an open problem. The categorical-limit argument therefore does not provide an identifiability theorem for finite-temperature GDM; interpretation in that regime rests on the learned primitives and empirical state-recovery controls.

Developing a full identifiability theory for the non-limiting case will require new mathematical statements. A potential route toward a formal proof may draw on the ideas in [7].

E Computational advantages and setups

We used official baseline implementations and their recommended training and test-time inference procedures. To keep observation-model flexibility comparable, we simplified the observation layers of SNLDS and KVAE to match those of the linear models. Hyperparameters, including learning rate and latent dimension, were selected per dataset using the appropriate validation objective on held-out training trials or segments; neither test data nor expert labels were used for selection. We held K equal across methods in direct state-recovery comparisons.

GDM replaces variational EM with fully differentiable amortized inference, leading to two computational advantages:

- **Training time** Each iteration of GDM uses standard backpropagation with reparameterized Gumbel-Softmax samples, avoiding the iterative E-step optimization over latent trajec-

tories required in prior methods. Empirically, this yields substantially faster wall-clock convergence under comparable architectures and hardware (as discussed below).

- **Inference** Latent states are obtained via a single forward pass, whereas baselines often require per-sequence optimization over latent trajectories, leading to substantially higher test-time cost (often an order-of-magnitude difference in practice).

To isolate per-iteration efficiency, Table 3 reports per-iteration wall-clock time across methods and datasets. Experiments were conducted on a single GPU (NVIDIA g4dn.4xlarge instance, 16GB GPU RAM). Batch sizes were matched across methods. As shown in Table 3, GDM consistently achieves lower cost, especially as sequence length and dataset size grow.

	SLDS	rSLDS	pdLDS	SNLDS	GDM
Nascar	0.080	0.130	0.920	2.12	0.010
F1	1.650	3.860	20.25	—	0.036
CalMS	—	54.88	—	—	1.18

Table 3: Per-iteration wall-clock time (in seconds) across datasets. Lower values indicate better performance. Missing entries correspond to CUDA out-of-memory errors or execution timeouts.

In addition to computational efficiency, inference strategies differ qualitatively across models and impact generalization behavior. As discussed in the main text, most SDS baselines require re-optimizing variational posteriors on new sequences to achieve strong test performance, reflecting the lack of amortization. In contrast, GDM performs inference via a single forward pass.

KVAE and SNLDS also support amortized inference and can generalize without retraining, but rely on additional architectural or inference design choices. KVAE introduces two separate networks: a VAE that maps observations into a low-dimensional latent trajectory, and a dynamics-parameter network that maps this trajectory to time-varying system parameters. SNLDS, which inherits collapsed variational inference, supports expressive latent dynamics and amortized inference for continuous variables; however, because discrete switching variables are marginalized out rather than inferred directly, the discrete states must be recovered post hoc, limiting interpretability.

Combined with the absence of test-time optimization, this leads to significantly faster end-to-end performance for GDM.

F More discussions on the NASCAR dataset

The full generative model used to simulate the NASCAR dataset is described as follows,

$$z_1 \sim \mathrm{GS}(\pi_1, \tau), \quad z_t | x_{t-1} \sim \mathrm{GS}(T x_{t-1} + r, \tau) \quad t \geq 2 \tag{8}$$

$$x_1 = \sum_{k=1}^{4} z_{1,k} \mu_k, \quad x_t | x_{t-1}, z_t = \sum_{k=1}^{4} z_{t,k} (A_k x_{t-1} + c_k) \quad t \geq 2 \tag{9}$$

$$y_t | x_t \sim \mathcal{N}(C x_t, \sigma I), \quad t \geq 1 \tag{10}$$

This can be achieved by setting extreme Gumbel-Softmax logits in eqn. (8). As an example, the transition matrix T and the bias r can be defined as

$$T = \begin{bmatrix} 10 & 0 \\ -10 & 0 \\ 0 & 10 \\ 0 & -10 \end{bmatrix} \qquad r = \begin{bmatrix} -20 \\ -20 \\ -10 \\ -10 \end{bmatrix}$$

Eqn. (8) can be viewed as a classifier that divides the space into four regions such that the logit of each region k is computed as $T_k \cdot x + r_k$ where $x \in \mathbb{R}^2$ denotes the point on the 2D trajectory. For example, if $x_1 > 2$ and $-1 < x_2 < 1$, the first logit will be greater than 0 while other logits will be smaller than 0, so the point is highly likely to be classified in state $k = 1$.

Eqn. (9) specifies how the system moves in each state. For the standard NASCAR, the ground truth dynamics matrices are defined as,

$$A_1 = A_2 = \text{expm}\left(\begin{bmatrix} 0 & \frac{\pi}{24} \\ -\frac{\pi}{24} & 0 \end{bmatrix}\right), \quad A_3 = A_4 = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

where the first two states correspond to the semicircular turns of $7.5°$ at the end of the straight track. The ground-truth offsets are defined as,

$$c_k = \begin{cases} -(A_1 - I) \cdot \text{FP}_1, & k = 1 \\ -(A_2 - I) \cdot \text{FP}_2, & k = 2 \\ [0.1 \quad 0], & k = 3 \\ [-0.25 \quad 0], & k = 4 \end{cases}$$

where c_1 and c_2 specify rotations around $\text{FP}_1 = (2, 0)$ and $\text{FP}_2 = (-2, 0)$ at the semicircular turns, while c_3 and c_4 specify the constant speed along the straight track.

To model variable-speed transitions, we may introduce another parameter s that denotes a varied speed for the dynamics equation (9) such that $\tilde{c}_k = sc_k$ where $s \in [s_{min}, 1]$ is uniformly sampled between a minimum low speed s_{min} and full speed and is applied throughout the linear segment of the track. The observation is generated in the same way as before. Given the previous location in the trajectory x_{t-1} and the current state z_t, we can generate the next trajectory point using eqn. (9). The trajectory is then mapped to the observations. Note that the shape of the trajectory will not be changed fundamentally by varying speed as the movement direction of each state remains unchanged.

G Additional results for F1 dataset

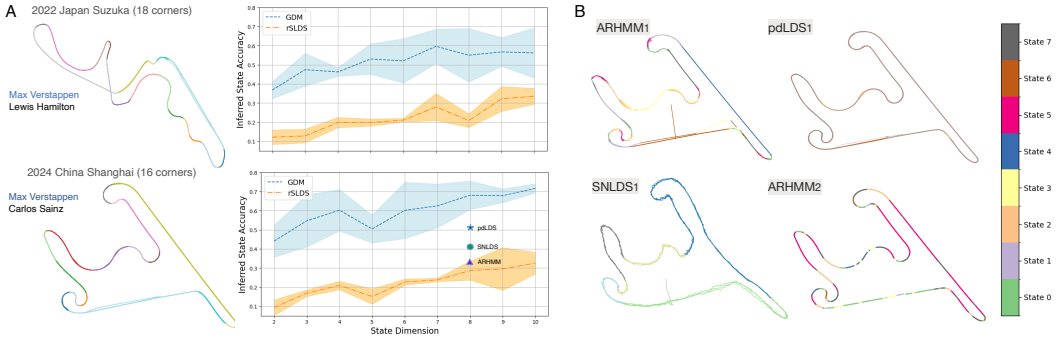


Figure 5: A. Comparison of inferred state accuracy between GDM and rSLDS across state dimensionalities on the Shanghai International Circuit and Suzuka Circuit. Performance is evaluated over 5 train/test splits with different random seeds; shaded regions denote standard deviation across seeds. GDM consistently achieves higher state accuracy, particularly at low dimensions. B. Example inferred trajectories from additional baselines not shown in the main figure, including ARHMM, pdLDS, and SNLDS for the position-based setting, and ARHMM for the velocity-based setting. The corresponding inferred state accuracy for this single run is marked in Figure 5A.

As mentioned in the main text, GDM utilizes all states, but not equally. In Figure 6A, we show the complete state usage of GDM for the trial illustrated in Figure 3C. For demonstration purposes, we display the first three laps around the track. As seen in the plot, while all states capture the three laps as three clear peaks in probability, States 1, 3, 4, and 5 are more dominant than the other four states. This is also reflected in the state annotations in Figure 3C. Here, we provide a more detailed version of Figure 3C by lowering the presence threshold to 5% of all time steps associated with the expert-labeled state. The complementary states for each segment are greyed out.

The unequal usage of states helps explain the observation that the inferred state accuracy of GDM improves rapidly in the initial steps and then plateaus. GDM allocates additional states to less dominant roles, so the marginal gain of increasing the number of states decreases after the first few.

For practical visualization, we put an emphasis on the dominant states. Specifically, we set transparency to the maximum value of state proportions at each time step and mix colors according to

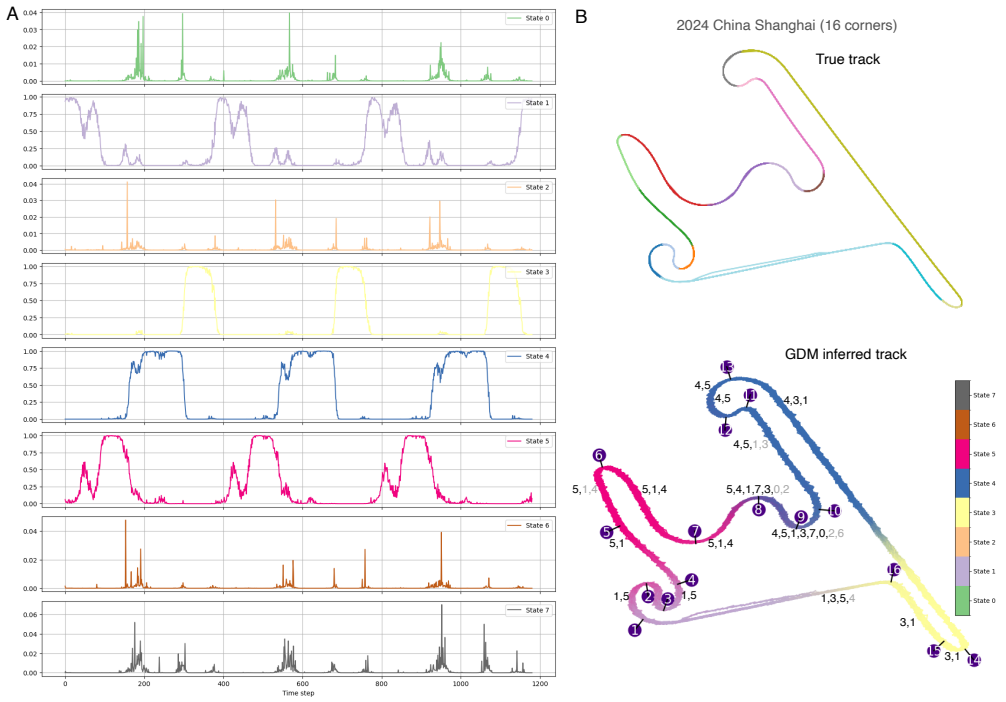


Figure 6: A. Complete state usages for Figure 3C. Example inferred trajectory for GDM, with complementary states annotated in grey.

the proportions of active states. This yields a gradual change in color across transitions and more transparent segments where mixtures of overlapping states occur.

H Ethics statement

We conceived GDM as a tool to improve analysis of dynamical phenomena. While we hope that it will benefit society in the longer run by supporting progress across scientific disciplines, we do not think our work carries any immediate societal impact.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? Answer: [Yes]

   Justification: The abstract and introduction clearly state the proposed Gumbel Dynamical Model (GDM), its motivation (modeling soft and stochastic transitions), and its empirical advantages. These claims are supported by both methodological development (Section 2–3) and experimental results (Section 4).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: A dedicated limitations paragraph is included in the conclusion.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper provides formal model definitions and derivations in the main text, with additional theoretical details (including equivalence results and identifiability discussion) deferred to the appendix with appropriate references.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The model architecture, training procedure, inference method, and evaluation metrics are described in the main text, with detailed experimental setups, dataset descriptions, and additional implementation details provided in the appendix. Code is available in a public GitHub repository.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: A public code repository is provided in the paper. All datasets used (synthetic NASCAR, F1 via FastF1, and CalMS21) are publicly available, and instructions for accessing them are described.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies dataset splits, model configurations (linear vs RNN-based), evaluation metrics, and training procedures. Additional details are provided in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Results are reported across multiple random seeds (e.g., Table 1 reports mean and standard deviation over 10 runs), capturing variability due to initialization and stochastic training.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports per-iteration wall-clock time comparisons and implementation details. The models are trained using standard deep learning methods and moderate computational resources.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is methodological, uses publicly available datasets, and does not involve sensitive data or harmful applications. Care is taken to present results transparently and reproducibly.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: An ethics statement is included in Appendix H. The work is primarily methodological with no immediate societal deployment, and potential impacts are discussed conservatively.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk models or datasets; it introduces a general modeling framework and uses publicly available data.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets (NASCAR, FastF1, CalMS21) and prior methods are properly cited. These are publicly available resources with established usage terms.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce new datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human subject experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: The paper does not involve crowdsourcing or human subject experiments. All datasets are pre-existing and publicly available.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as part of the methodology or experiments.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.