Latent Neural ODE for Irregular Time Series

Oualid Missaoui

Agenda

- Historical Note
- Irregular Time Series
- Latent Neural ODE
  - Core idea
  - Intuition
  - Architecture
  - Learning
- References

Historical Note

Neural Ordinary Differential Equations

Ricky T. Q. Chen*, Yulia Rubanova*, Jesse Bettencourt*, David Duvenaud
University of Toronto, Vector Institute
{rtqichen, rubanova, jessebett, duvenaud}@cs.toronto.edu

Abstract

We introduce a new family of deep neural network models. Instead of specifying a discrete sequence of hidden layers, we parameterize the derivative of the hidden state using a neural network. The output of the network is computed using a black-box differential equation solver. These continuous-depth models have constant memory cost, adapt their evaluation strategy to each input, and can explicitly trade numerical precision for speed. We demonstrate these properties in continuous-depth residual networks and continuous-time latent variable models. We also construct continuous normalizing flows, a generative model that can train by maximum likelihood, without partitioning or ordering the data dimensions. For training, we show how to scalably backpropagate through any ODE solver, without access to its internal operations. This allows end-to-end training of ODEs within larger models.

Ricky Chen et al , Neural Ordinary Differential Equations

Latent ODEs for Irregularly-Sampled Time Series

Yulia Rubanova, Ricky T. Q. Chen, David Duvenaud
University of Toronto and the Vector Institute
{rubanova,rtqichen,duvenaud}@cs.toronto.edu

Abstract

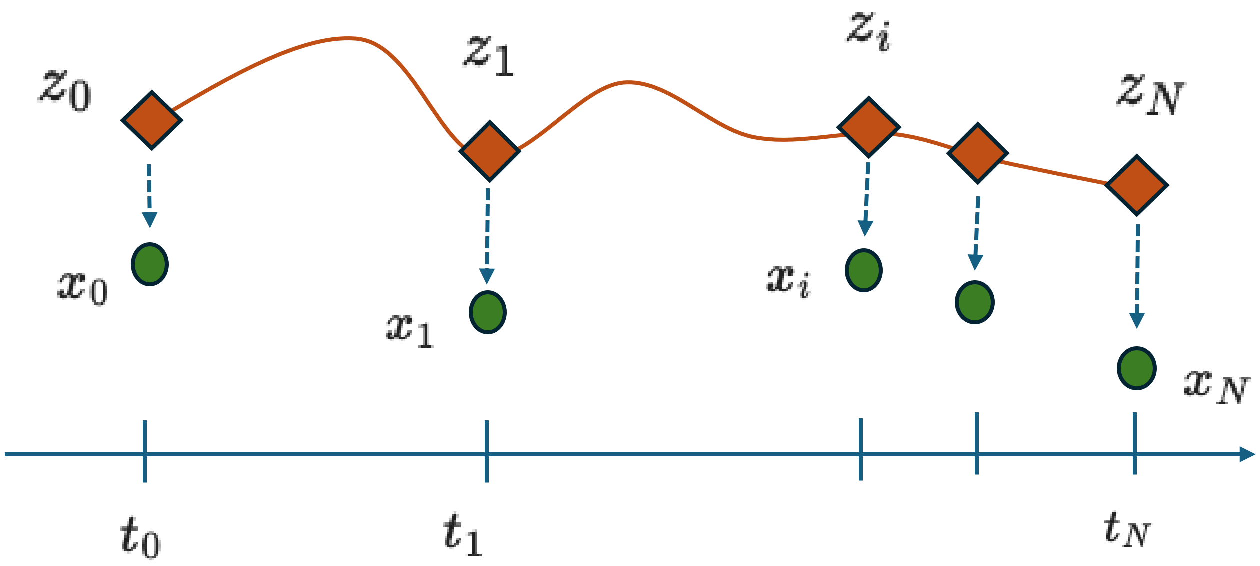
Time series with non-uniform intervals occur in many applications, and are difficult to model using standard recurrent neural networks (RNNs). We generalize RNNs to have continuous-time hidden dynamics defined by ordinary differential equations (ODEs), a model we call ODE-RNNs. Furthermore, we use ODE-RNNs to replace the recognition network of the recently-proposed Latent ODE model. Both ODE-RNNs and Latent ODEs can naturally handle arbitrary time gaps between observations, and can explicitly model the probability of observation times using Poisson processes. We show experimentally that these ODE-based models outperform their RNN-based counterparts on irregularly-sampled data.

Yulia Rubanova et al, Latent ODEs for Irregularly-Sampled Time Series

Irregular Time Series

- **Definition**
  - **Asynchronous timestamps**
    - Trades, quotes, news, order arrivals, etc.
  - **Uneven gaps** and **missing features**
    - e.g. sparse fundamentals vs. dense tick data
- **Challenges for the usual toolkit**
  - Resampling to a grid introduces **bias** (microstructure noise) and **wastes data**.
  - Interpolate / impute $\rightarrow$ then run a standard RNN: **grid artifacts** and **error compounding**.
  - RNNs with Δt inputs (e.g., GRU-D) help, but are still **discrete-time** models.
  - We want a model that *natively* reasons in **continuous time** and at **arbitrary timestamps**.

Core Idea



- **Continuous-time dynamics for irregular data**
  Latent state $z(t)$ evolves via an ODE

$$\dot{z}(t) = f_\theta(z(t), t), \quad z(0) = z_0,$$

  so $z(t)$ is defined for **all** t and can be evaluated at arbitrary, uneven time points.

- **Generative latent process assumption**
  Irregular samples $\{x_i\}$ at times $\{t_i\}$ are noisy observations of a smooth latent path:

$$z_0 \sim p(z_0), \quad z(t) \text{ via ODE}, \quad x_i \sim p_\psi\big(x_i \mid z(t_i)\big).$$

- **Neural ODE as flexible dynamics**
  f_θ is a neural net $\to$ rich, nonlinear latent dynamics (far more expressive than fixed decay / AR models).

- **Forecasting & simulation**
  Once f_θ and the posterior over z_0 are learned, we can solve the ODE to get $z(t)$ for any t and forecast or sample

$$\hat{x}_t \sim p_\psi\big(x_t \mid z(t)\big).$$

- **Key question**
  How do we **identify** this latent continuous-time model from sparse, irregular data?
  $\to$ Need an encoder (e.g. ODE-RNN) + variational framework to infer z_0 and learn f_θ, p_ψ.

Intuition- Latent Neural ODE as Self-Supervised Encoder-Decoder

Latent Generative Process	Encoder–Decoder & Inference

Encoder–decoder viewpoint (self-supervised)
Use the **history itself** as supervision.
• **Encoder:** read the irregular history $\{(t_i, x_i)\}$ and infer an initial latent state

$$q_\phi\big(z_0 \mid \{(t_i, x_i)\}\big).$$

• **Decoder:** from z_0, solve the Neural ODE to get $z(t_i)$, then decode back to reconstruct the series

$$\hat{x}_i \sim p_\psi\big(x \mid z(t_i)\big).$$

All parameters (ϕ, θ, ψ) are learned jointly so reconstructions match the data.

Why an ODE-RNN encoder?
The encoder solves the inverse problem: *given irregular samples of the trajectory, what initial state z_0 would have generated them?*
• evolve a hidden state with an ODE between observations;
• update it with an RNN cell at each t_i;
• map the final hidden state to the posterior over z_0

Generative assumption
There is a smooth latent process $z(t)$ that generates the irregular series:

$$z_0 \sim p(z_0), \quad \dot{z}(t) = f_\theta(z(t), t), \quad x_i \sim p_\psi\big(x_i \mid z(t_i)\big).$$

Observations are noisy samples of this hidden trajectory at irregular times $\{t_i\}$.

More than a vanilla VAE
In a standard VAE, latent variables are just free vectors. Here we impose a **dynamics constraint**:

$$z(t) \text{ must satisfy } \dot{z}(t) = f_\theta(z(t), t),$$

and we only observe / decode at irregular times t_i. This continuous-time constraint is what lets the model handle **asynchronous, non-uniform time steps**.

Latent Neural ODE Architecture

Encoder

ODE-RNN

Decoder

Neural ODE

MLP

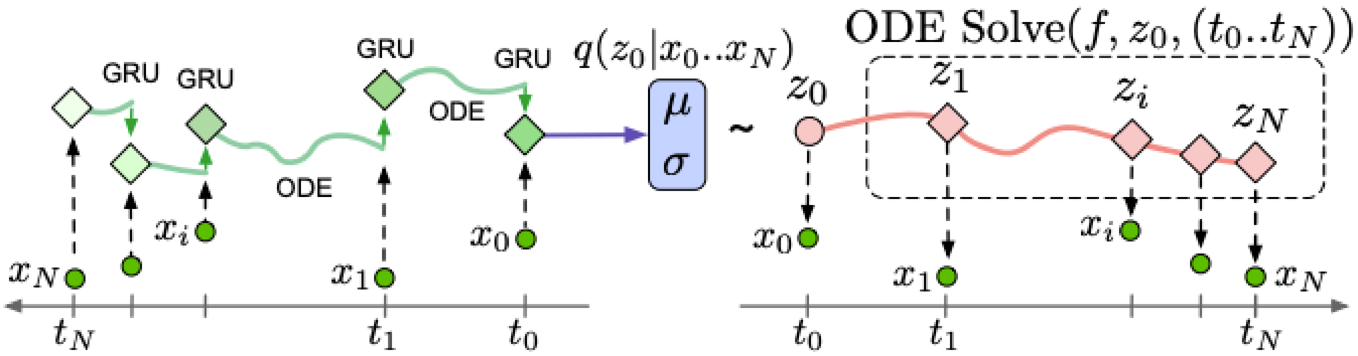


Figure 2: The Latent ODE model with an ODE-RNN encoder. To make predictions in this model, the ODE-RNN encoder is run backwards in time to produce an approximate posterior over the initial state: $q(z_0|\{x_i, t_i\}_{i=0}^{N})$. Given a sample of z_0, we can find the latent state at any point of interest by solving an ODE initial-value problem. Figure adapted from Chen et al. [2018].

Learning Latent Neural ODEs - Goal & Likelihood

- **Problem setup (times are given):** model $p(x_{1:N} \mid t_{1:N})$ for irregular times $t_1 < \cdots < t_N$.
- **Generative story**

$$z_0 \sim p(z_0), \qquad \dot{z}(t) = f_\theta(z(t), t), \qquad x_i \sim p_\psi(x \,|\, z(t_i)).$$

- **Maximum likelihood objective (dataset of sequences $\mathcal{D} = \{\mathcal{D}^b\}$):**

$$\max_{\theta,\psi} \sum_b \log p_{\theta,\psi}(x^{(b)}_{1:N_b} \mid t^{(b)}_{1:N_b}) = \sum_b \log \int p(z_0) \prod_{i=1}^{N_b} p_\psi(x_i^{(b)} \mid z_\theta(t_i^{(b)}; z_0))\, dz_0.$$

Intractable integral over $z_0 \Longrightarrow$ need variational inference.

From MLE to ELBO (intuition)

- Insert an **approximate posterior** $q_\phi(z_0 \mid \mathcal{D}^b)$ and apply Jensen:

$$\log p(x_{1:N}) = \log \mathbb{E}_{q_\phi}\Big[\tfrac{p(x_{1:N},z_0)}{q_\phi(z_0|x)}\Big] \;\geq\; \mathbb{E}_{q_\phi}\big[\log p(x_{1:N} \mid z_0)\big] - \mathrm{KL}\big(q_\phi \| p\big).$$

- **Evidence Lower Bound (ELBO) per sequence $\mathcal{D}^b$:**

$$\mathcal{L}^b(\theta,\psi,\phi) = \underbrace{\mathbb{E}_{q_\phi}\Big[\sum_{i=1}^{N_b} \log p_\psi\big(x_i^{(b)} \mid z_\theta(t_i^{(b)}; z_0)\big)\Big]}_{\text{reconstruction at irregular times}} \;-\; \underbrace{\mathrm{KL}\big(q_\phi(z_0 \mid \mathcal{D}^b) \,\|\, p(z_0)\big)}_{\text{regularize latent}}.$$

- **Learning objective:** maximize $\sum_b \mathcal{L}^b$.

Encoder = ODE-RNN (posterior over the initial state)

- **Goal:** $q_\phi(z_0 \mid \{(t_i, x_i)\}_{i=1}^{N})$ that respects **irregular gaps**.
- **Hidden dynamics (encoder side):**

$$\frac{dh(t)}{dt} = f_\phi^{\mathrm{enc}}(h(t), t).$$

- **Reverse-time encoding (recommended):** process $t_N \to t_1$

$$\begin{aligned} h_i^- &= \mathrm{ODESolve}\big(f_\phi^{\mathrm{enc}},\ h_{i+1}^+,\ t_{i+1} \to t_i\big), \\ h_i^+ &= \mathrm{GRUCell}_\phi\big([x_i,\ m_i,\ \Delta t_i],\ h_i^-\big). \end{aligned}$$

  Output posterior params

$$(\mu_0, \log \sigma_0^2) = g_\phi(h_1^+), \quad q_\phi = \mathcal{N}(\mu_0, \mathrm{diag}\, \sigma_0^2).$$

- **Why ODE-RNN?** Belief $h(t)$ **drifts** continuously between events and **updates** only when data arrives.

What is inside the reconstruction term ?

- **ODE-solve:** $z_\theta(t_i^{(b)}; z_0) = \texttt{ODESolve}(f_\theta, z_0, t_i^{(b)})$.
- **Likelihoods by data type (masked for missing dims):**

$$\log p_\psi(x_i^{(b)} \mid z) = \sum_d m_{i,d}^{(b)} \log p_{\psi,d}\big(x_{i,d}^{(b)} \mid z\big),$$

  e.g.
  - real-valued: $\mathcal{N}\left(\mu_\psi(z), \operatorname{diag} \sigma_\psi^2(z)\right)$,
  - binary: Bernoulli $(\sigma(h_\psi(z)))$,
  - counts: Poisson/NegBin.
- **Interpretation:** fit a continuous trajectory $z(t)$ whose **decoded** values have high likelihood at the **exact** irregular times.

Making the ELBO trainable

- **Reparameterize z_0 (amortized inference):**
  $q_\phi(z_0 \mid \mathcal{D}) = \mathcal{N}(\mu_\phi, \operatorname{diag} \sigma_\phi^2)$,
  $z_0 = \mu_\phi + \sigma_\phi \odot \epsilon,\ \epsilon \sim \mathcal{N}(0, I)$.
- **Gradients through the ODE:**
  - Direct backprop through the solver (easy to debug), or
  - **Adjoint method** for constant memory.
- **Final train loss (mini-batch):**

$$\mathcal{J} = -\frac{1}{B} \sum_{b=1}^{B} \Big[\mathbb{E}_{q_\phi} \big[\sum_i \log p_\psi(x_i^{(b)} \mid z_\theta(t_i^{(b)}; z_0)) \big] - \beta \, \mathrm{KL}(q_\phi \| p) \Big],$$

with optional **KL warm-up** ($\beta : 0 \to 1$) and regularizers on f_θ (e.g., $\| f_\theta \|^2$).

KL Term via Monte Carlo (when no closed form)

- **When do we need MC?**
  If either $q_\phi(z_0 \mid \mathcal{D})$ or $p_\eta(z_0)$ is **not** a simple Gaussian (e.g., **normalizing flow**, **mixture prior**, hierarchical/learned prior), the KL has no closed form.

- **Identity to estimate:**

$$\mathrm{KL}\big(q_\phi \,\|\, p_\eta\big) = \mathbb{E}_{z\sim q_\phi}\big[\log q_\phi(z \mid \mathcal{D}) \;-\; \log p_\eta(z)\big].$$

- **Monte Carlo estimator (per sequence):**
  Sample $z^{(k)} \sim q_\phi(z \mid \mathcal{D})$, $k = 1, \dots, K$:

$$\widehat{\mathrm{KL}} = \frac{1}{K} \sum_{k=1}^{K} \Big(\log q_\phi\big(z^{(k)} \mid \mathcal{D}\big) - \log p_\eta\big(z^{(k)}\big) \Big).$$

  Use **reparameterization** (e.g., $z^{(k)} = T_\phi(\epsilon^{(k)}, \mathcal{D})$, $\epsilon \sim \mathcal{N}(0, I)$) so gradients flow through samples.

- **Batch objective contribution:**
  Average the per-sequence $\widehat{\mathrm{KL}}$ over the mini-batch; optionally apply β**-weighting** or **free-bits**.

- **Variance control:**
  Small K (1–5) usually suffices; use common random numbers across steps, antithetic pairs, or low-variance base noise if needed.

- **Note:**
  If q_ϕ and p_η are tractable densities (flows, mixtures), $\log q_\phi$ and $\log p_\eta$ are computed **exactly**; only the **expectation** is approximated by MC.

Training loop (nuts & bolts)

1. **Encode** irregular history $\{(t_i, x_i)\}$ with ODE-RNN $\rightarrow \mu_\phi, \sigma_\phi$.
2. **Sample** z_0 (reparameterization).
3. **Integrate** latent ODE to all observation times $\{t_i\} \rightarrow z(t_i)$.
4. **Decode** to likelihood params; compute masked log-likelihood.
5. **KL** to prior on z_0.
6. **Maximize ELBO** (or minimize ELBO) via Adam; clip grads; monitor NFEs.

> **Intuition:** ELBO balances *faithful reconstruction at irregular timestamps* with a *simple, generalizable latent initialization*, pushing dynamics f_θ to explain temporal structure rather than overfitting z_0.

References

- Ricky Chen et al , Neural Ordinary Differential Equations
- Yulia Rubanova et al, Latent ODEs for Irregularly-Sampled Time Series