Neural CDEs for Long Time Series via the Log-ODE Method

James Morrill, Patrick Kidger, Cristopher Salvi, James Foster, and Terry Lyons

University of Oxford; Alan Turing Institute.

Summary

Neural CDEs can be viewed as a continuous-time RNN, much as Neural ODEs with ResNets. However, as with RNNs, training can begin to breakdown for long time series.

In this paper we show:

- How to convert the CDE equation into an equivalent ODE by using the *log-ODE method*, which is a standard method for solving CDEs in the field of rough analysis. Tools from Neural ODEs can then be directly applied.
- In particular this allow us to take integration steps *larger* than the discretisation of the data – whilst retaining solution accuracy.
- On long time series, it can result in speed-ups of up to 100×, significantly reduced memory requirements, and improvements in performance.

Background: Neural CDEs

Consider a time series $\mathbf{x}$ as a collection of points $x_i \in \mathbb{R}^{v-1}$ with corresponding time-stamps $t_i \in \mathbb{R}$ such that $\mathbf{x} = ((t_0, x_0), (t_1, x_1), \ldots, (t_n, x_n))$, and $t_0 < \ldots < t_n$ and let $X \colon [t_0, t_n] \to \mathbb{R}^v$ be some interpolation of the data such that $X_{t_i} = (t_i, x_i)$.

Let $\xi_\theta \colon \mathbb{R}^v \to \mathbb{R}^w$ and $f_\theta \colon \mathbb{R}^w \to \mathbb{R}^{w \times v}$ be neural networks and let $\ell_\theta \colon \mathbb{R}^w \to \mathbb{R}^q$ be linear.

We define Z as the hidden state and Y as the output of a *neural controlled differential equation* driven by X if

$$Z_{t_0} = \xi_\theta(t_0, x_0), \quad \text{with} \quad Z_t = Z_{t_0} + \int_{t_0}^{t} f_\theta(Z_s)\,\mathrm{d}X_s \quad \text{and} \quad Y_t = \ell_\theta(Z_t) \quad \text{for } t \in (t_0, t_n]. \tag{1}$$

- If instead of $\mathrm{d}X_s$ there was $\mathrm{d}s$, then this would be a Neural ODE.
- The $\mathrm{d}X_s$ is what allows the hidden state to be updated from incoming data. (Unlike Neural ODEs, for which the solution is determined by the initial condition.)
- For a differentiable control path X, the simple way to convert to an ODE via $\mathrm{d}X_s = \frac{\mathrm{d}X}{\mathrm{d}s}(s)\mathrm{d}s$.

Background: The Log-ODE Method

Consider the CDE integral equation for Z_t defined above. The log-ODE method states:

$$Z_b \approx \widehat{Z}_b \quad \text{where} \quad \widehat{Z}_u = \widehat{Z}_a + \int_a^u \widehat{f}(\widehat{Z}_s)\frac{\mathrm{LogSig}^N_{a,b}(X)}{b-a}\mathrm{d}s, \quad \text{and} \quad \widehat{Z}_a = Z_a. \tag{2}$$

- That is, the solution to the CDE in (1) can be approximated by the solution to the ODE in (2).
- The $\mathrm{d}X_s$ term has been replaced by $\frac{\mathrm{LogSig}^N_{a,b}(X)}{b-a}\mathrm{d}s$.
- $\mathrm{LogSig}^N_{a,b}(X)$ is a particular map to a vector of values that summarise X. Intuitively, these are the terms that are most relevant to solving the CDE equation.
- This approximation becomes arbitrarily good as $N \to \infty$.

Updating the Neural CDE Hidden State Equation

Assume the time series is very long. Pick points r_i such that $t_0 = r_0 < r_1 < \cdots < r_m = t_n$ with $m \ll n$. In model training we will simply split the integral from equation (1) over each $[r_i, r_{i+1}]$ and update via

$$Z_{r_{i+1}} = Z_{r_i} + \int_{r_i}^{r_{i+1}} \widehat{f}_\theta(\widehat{Z}_s)\frac{\mathrm{LogSig}^N_{a,b}(X)}{b-a}\mathrm{d}s. \tag{3}$$

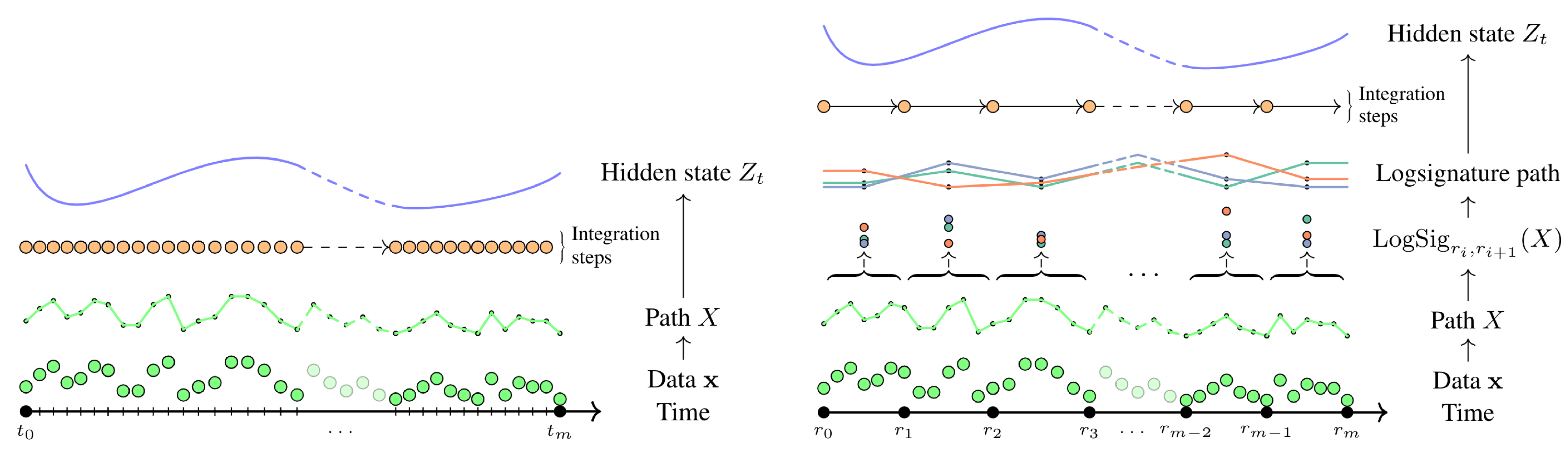

Figure 1: **Left:** The original NCDE formulation. The path X_t is quickly varying, meaning a lot of integration steps are needed to resolve it. **Right:** The log-ODE method. The log-signature path is more slowly varying (in a higher dimensional space), and needs fewer integration steps to resolve.

- The length of the sequence is now m, which was chosen to be much smaller than n. This is the source of the speed-ups of this method; we observe typical speed-ups by a factor of about 100.
- If depth $N = 1$ and steps $r_i = t_i$ are used, then the above formulation exactly reduces onto the original Neural CDE formulation. Thus the log-ODE method in fact generalises the original approach.

Experiments

- Tested on four problems with lengths up to 17,000.
- Observe significant training speed-ups, reduced memory requirements, and even improvements in performance when compared to the original NCDE model.

Find out more:

This paper:	`https://arxiv.org/abs/2009.08295`
NCDEs paper:	`https://arxiv.org/abs/2005.08926`
Code:	`https://github.com/jambo6/neuralCDEs-via-logODEs`
Library:	`https://github.com/patrick-kidger/torchcde`
Email:	`morrill@maths.ox.ac.uk`