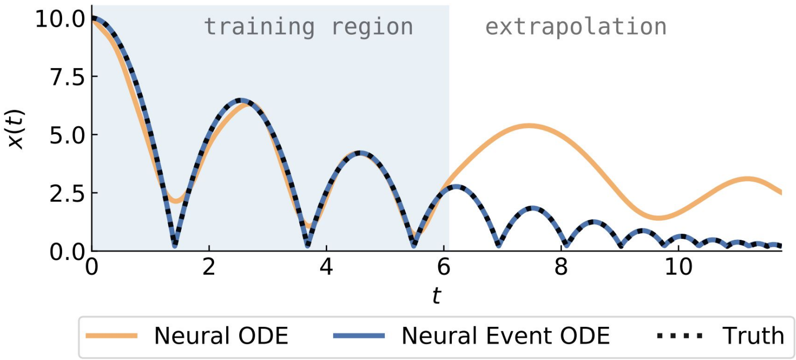
Neural Event Functions

Ricky T. Q. Chen, Brandon Amos, Maximilian Nickel



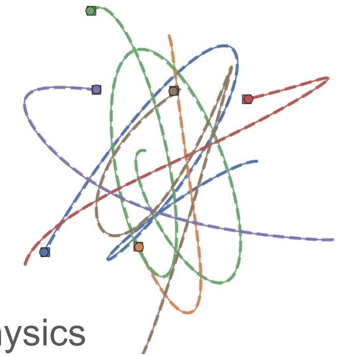
Neural Ordinary Differential Equations (ODEs)

We can (implicitly) define a path $x(t)$ satisfying the constraints

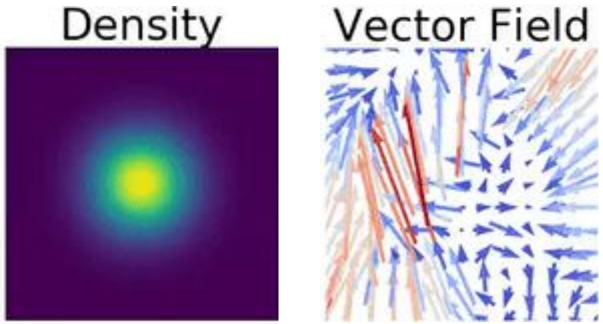
$$\frac{dx}{dt} = f(t, x(t))$$

(derivative)

(can be neural network)



Physics



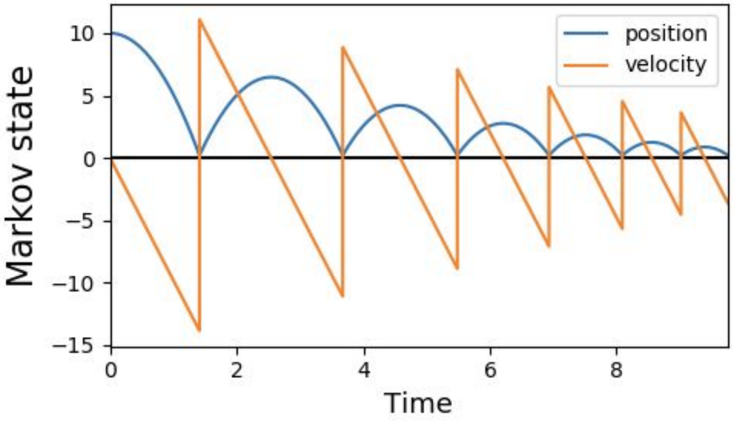
Probabilistic Modeling

Event Handling

- Stop solving “when an event occurs”.
- Defined as $g(x(t)) = 0$ for an *event function* g.
- Can introduce *discontinuities* at event times.

    E.g. State of a ball: (position, velocity)

    Velocity changes discontinuously upon impact.



Event Handling

- Stop solving “when an event occurs”.
- Defined as $g(x(t)) = 0$ for an *event function* g.
- Can introduce *discontinuities* at event times.

  E.g. State of a ball: (position, velocity)

  Velocity changes discontinuously upon impact.

Can we learn a *neural* event function?

- Yes! We can compute gradients using the implicit function theorem.
- Implemented in PyTorch as part of github.com/rtqichen/torchdiffeq.

Neural Event ODEs

Three components:

(1) A derivative / drift function f.

(2) An event function g.

(3) An instantaneous update function h.

the number of events is arbitrary and learned

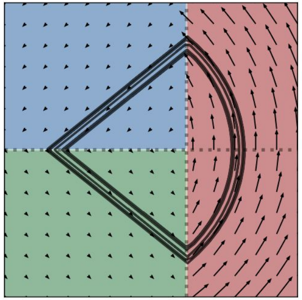
$i = 0$
while $t_i < T$ **do**
  $t_{i+1}, z'_{i+1} = \texttt{ODESolveEvent}(z_i, f, g, t_i)$ ▷ Solve until the next event
  $z_{i+1} = h(t_{i+1}, z'_{t+1})$ ▷ Determine how the event affects the state
  $i = i + 1$
end while
Return: event times $\{t_i\}$ and the piecewise continuous trajectory $\{z_i(t) \text{ for } t_i \leq t \leq t_{i+1}\}$

Switching Linear Dynamical Systems

- Deconstructs a complex dynamical system into interpretable components.
- Used in neuroscience, finance.

$$\frac{dz(t)}{dt} = \sum_{m=1}^{M} w_m \left(A^{(m)} z + b^{(m)}\right)$$

(element of a one-hot vector; switches instantly)



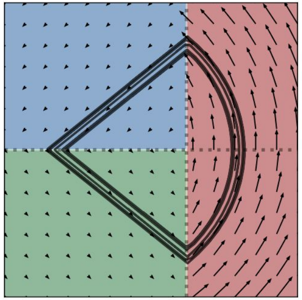
(a) Ground truth

Switching Linear Dynamical Systems

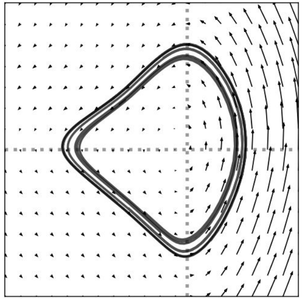
- Deconstructs a complex dynamical system into interpretable components.
- Used in neuroscience, finance.

$$\frac{dz(t)}{dt} = \sum_{m=1}^{M} w_m \left(A^{(m)} z + b^{(m)}\right)$$

(element of a one-hot vector; switches instantly)



(a) Ground truth



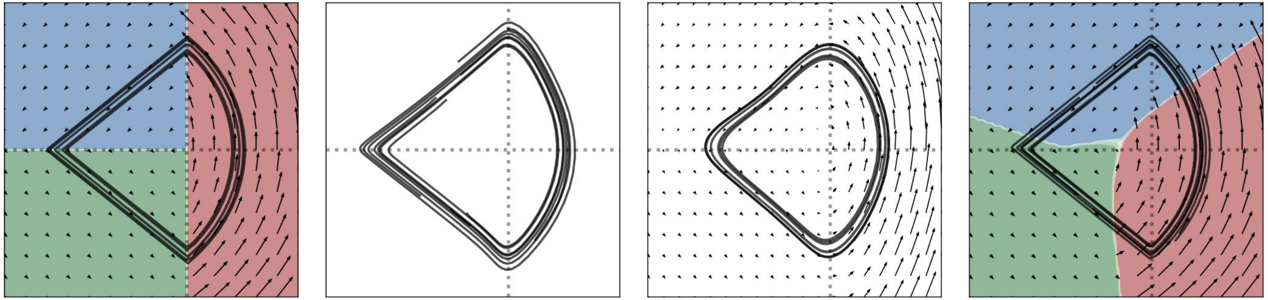
(c) Neural ODE

Switching Linear Dynamical Systems

- Deconstructs a complex dynamical system into interpretable components.
- Used in neuroscience, finance.

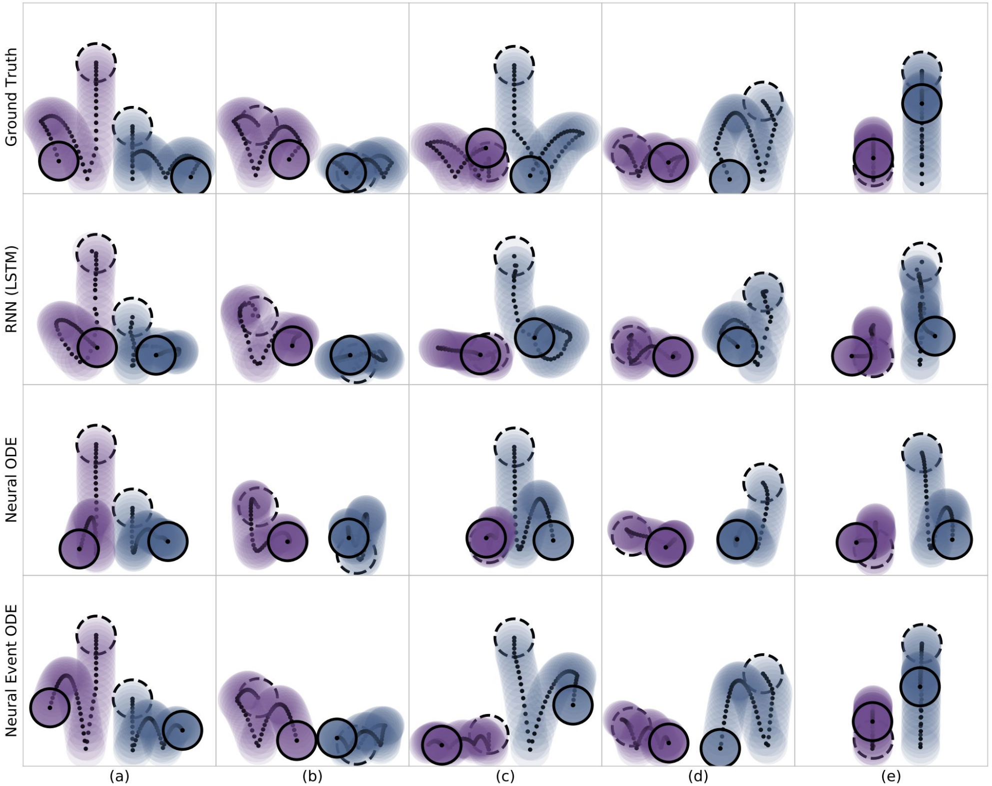
$$\frac{dz(t)}{dt} = \sum_{m=1}^{M} w_m \left(A^{(m)} z + b^{(m)}\right)$$

(element of a one-hot vector; switches instantly)



(a) Ground truth (b) RNN (LSTM) (c) Neural ODE (d) Neural Event ODE

Modeling Physics with Collision



Baselines hover instead of bounce.

Neural Event ODE on par with nonlinear Neural ODE,

but uses 10x less function evaluations to simulate.

Threshold-based Event Functions

Event occurs when an accumulator reaches a threshold.

$$t^* \text{ such that } s = \int_{t_0}^{t^*} \lambda(t)\, dt$$

(known) **(scalar; positive; neural network)**

Appears in event-based sampling, temporal point processes (TPP).

E.g. TPP sampling:

1) Sample the threshold $s \sim \mathrm{Exp}(1)$.
2) Compute t^*.

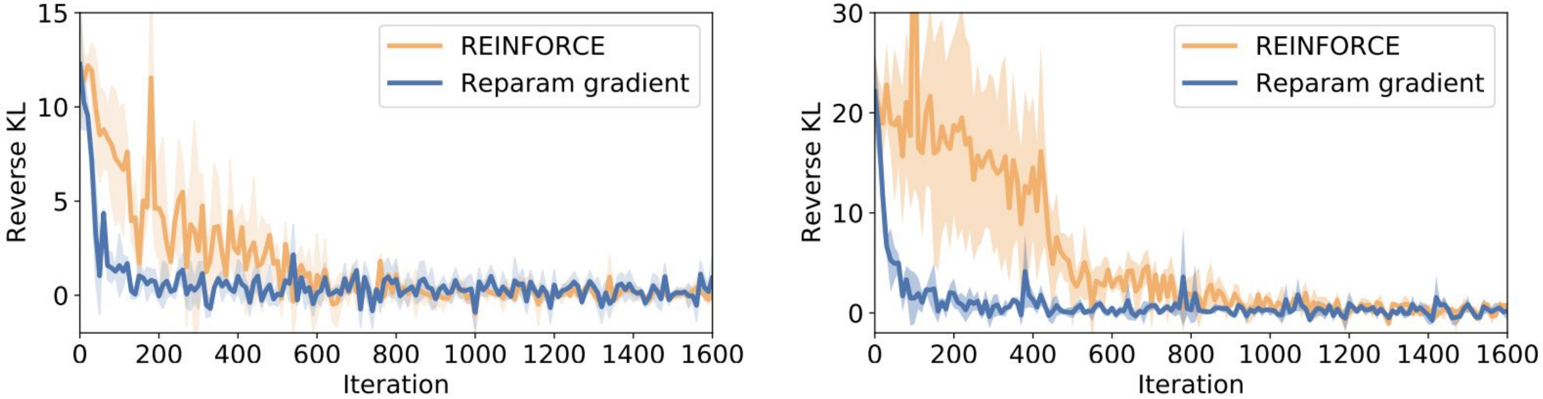
Repeat →

Samples:

$\{t_1, t_2, t_3, \ldots\}$

Temporal Point Processes (TPPs)

- We define the *reparameterization gradient* for TPPs.
- Previous works had to resort to REINFORCE gradient (high variance).

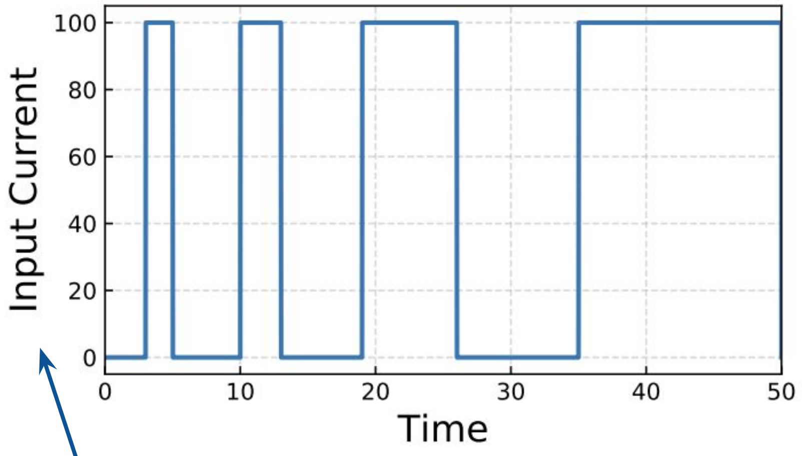


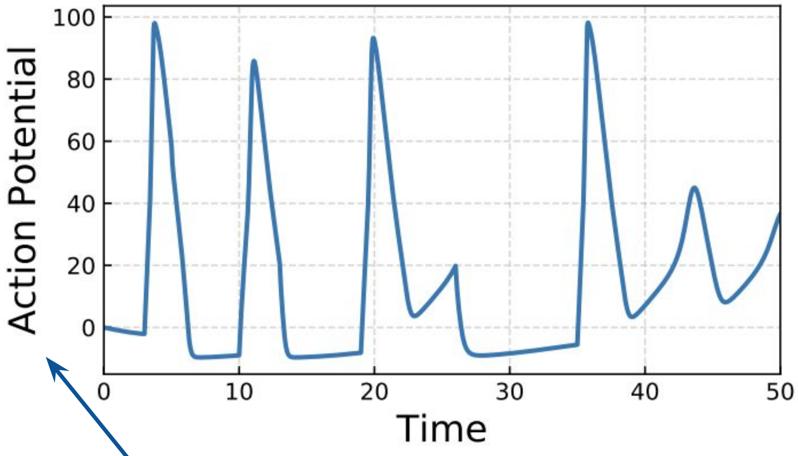
(a) Five events

(b) Ten events

Discrete Control in Continuous Time

Example of a neuronal dynamical system:



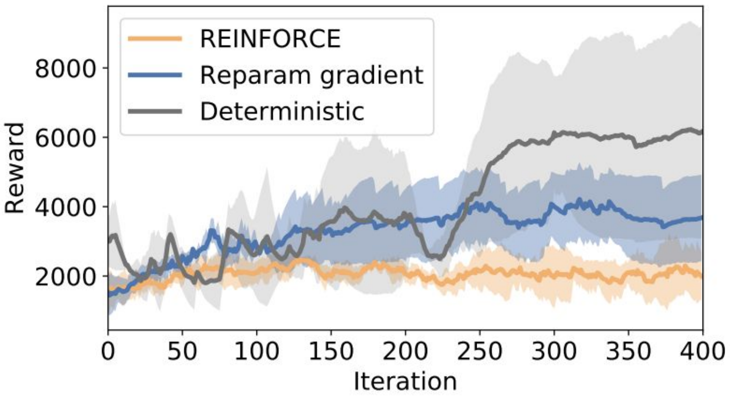


(policy; takes on a finite set of values)

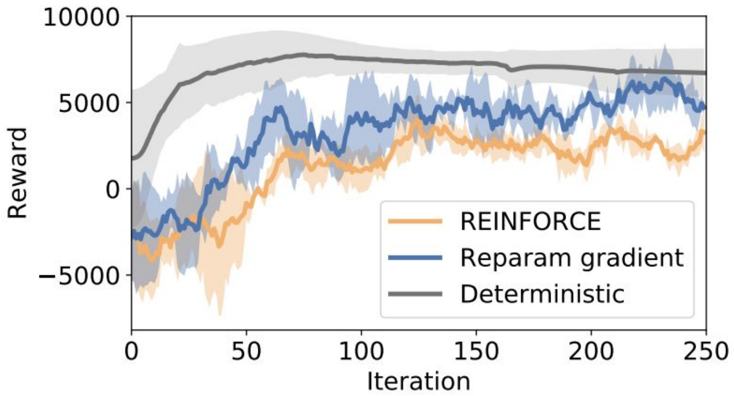
(environment; put a reward on this)

Discrete Control in Continuous Time

We learned discrete control in two systems:



(c) HIV dynamics model



(d) Hodgkin-Huxley dynamics model

Furthermore, we can learn **deterministic** discrete control policies.

Takeaway

- Event functions provide an implicit method of terminating ODEs.
- We can differentiate and train *neural event functions*.

Future applications:

- Useful for modeling robotic arms?
- Motion planning?