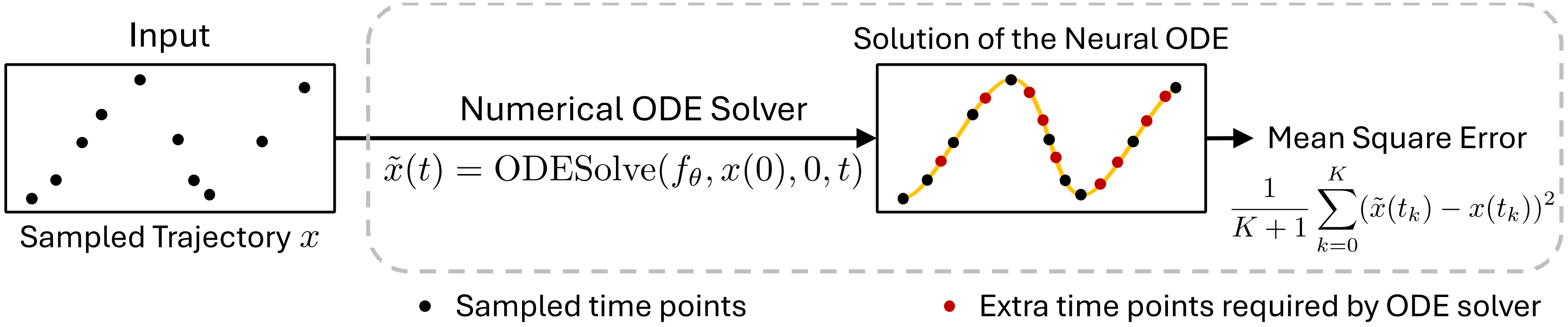
Accelerating Neural ODEs: A Variational Formulation-based Approach

Hongjue Zhao, Yuchen Wang, Hairong Qi,
Zijie Huang, Han Zhao, Lui Sha, Huajie Shao

Background: Neural ODEs

- **Neural ODE:** $\dot{\boldsymbol{x}} = \boldsymbol{f_\theta}(t, \boldsymbol{x})$ → *the vector field is parameterized using DNNs*
- **ODE-Solver-Based Training Methods:**

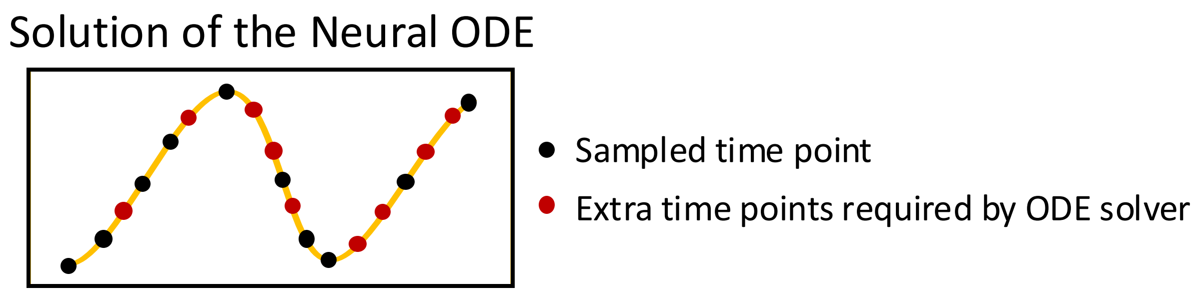


  - Forward Pass: using off-the-shelf numerical ODE solvers
  - Backward Pass:
    - *Discretize-then-Optimize*: directly backpropagating through ODE solvers
    - *Optimize-then-Discretize*: solving additional adjoint ODEs

Limitations of Existing Training Methods

High Computational Cost

- ODE solvers may need to evaluate the DNN-based vector field beyond given sampled data points for accuracy.



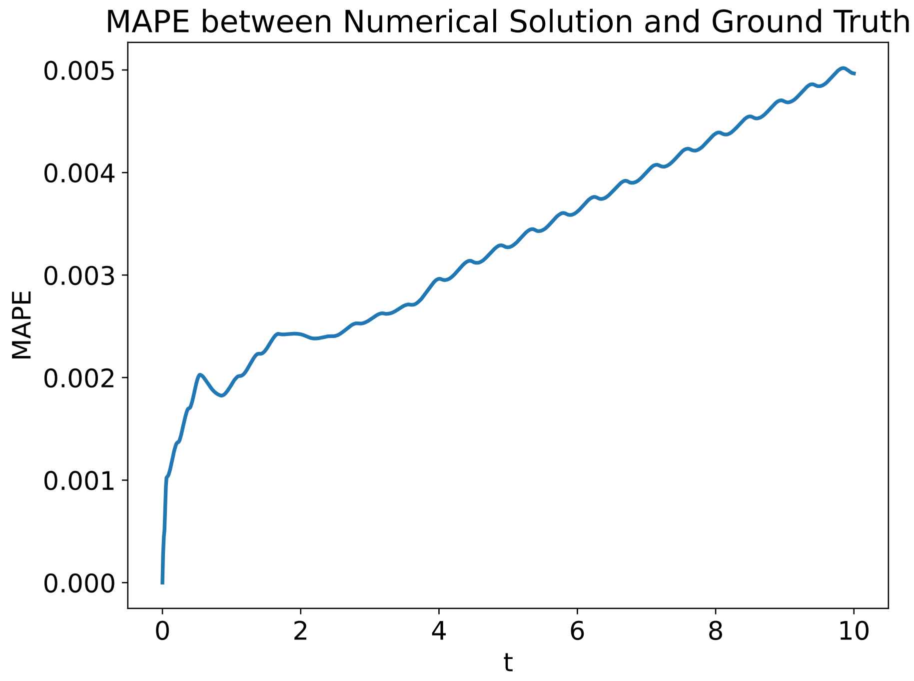
$$\tilde{x}(t) = \mathrm{ODESolve}(f_\theta, x(0), 0, t)$$

- The Optimize-then-Discretize approach may exacerbate this issue by the introduction of additional adjoint ODEs.

Limitations of Existing Training Methods

Low Accuracy

- The auto-regressive nature of ODE solvers can lead to error accumulation.
    - Most ODE solvers can be expressed as
    $$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + h\boldsymbol{g}(t_n, \boldsymbol{x}_n)$$
    - h is the step size, and $\boldsymbol{g}(t_n, \boldsymbol{x}_n)$ is the updating rule
- The Optimize-then-Discretize approach can introduce additional numerical errors [1]



[1] Gholami et. "Anode: Unconditionally accurate memory-efficient gradients for neural odes." *arXiv preprint arXiv:1902.10298* (2019).

Contributions

- We propose a novel variational formulation-based approach, **VF-NODE** to significantly accelerate Neural ODE training. By introducing the VF loss:
  - We drastically reduce the number of function evaluations
  - We eliminate the autoregressive process entirely
- We combine Filon's method with spline regression to efficiently compute oscillatory integrals from noisy and irregular data in the VF loss.
- Evaluations show our method speeds up training by 10–1000× compared to baselines, achieving comparable or better accuracy.

Variational Formulation of ODEs

Define the functional as

$$\mathbf{c}(\boldsymbol{x}, \boldsymbol{f}, \phi) := \int_0^T \boldsymbol{x}(t)\dot{\phi}(t)\,\mathrm{d}t + \int_0^T \boldsymbol{f}(t, \boldsymbol{x}(t))\phi(t)\,\mathrm{d}t$$

$\boldsymbol{x}$ is the solution to the ODE $\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x})$ if and only if

$$\mathbf{c}(\boldsymbol{x}, \boldsymbol{f}, \phi) = \mathbf{0}, \quad \forall \phi \in \mathcal{C}^1[0, T] \quad \text{s.t. } \phi(0) = \phi(T) = 0.$$

In this work, we utilize a series of Hilbert orthonormal basis as $\phi_\ell(t)$ to construct the VF loss:

$$\phi_\ell(t) = \sqrt{2/T}\sin(\pi \ell t/T), \quad \ell = 1, \ldots, L$$

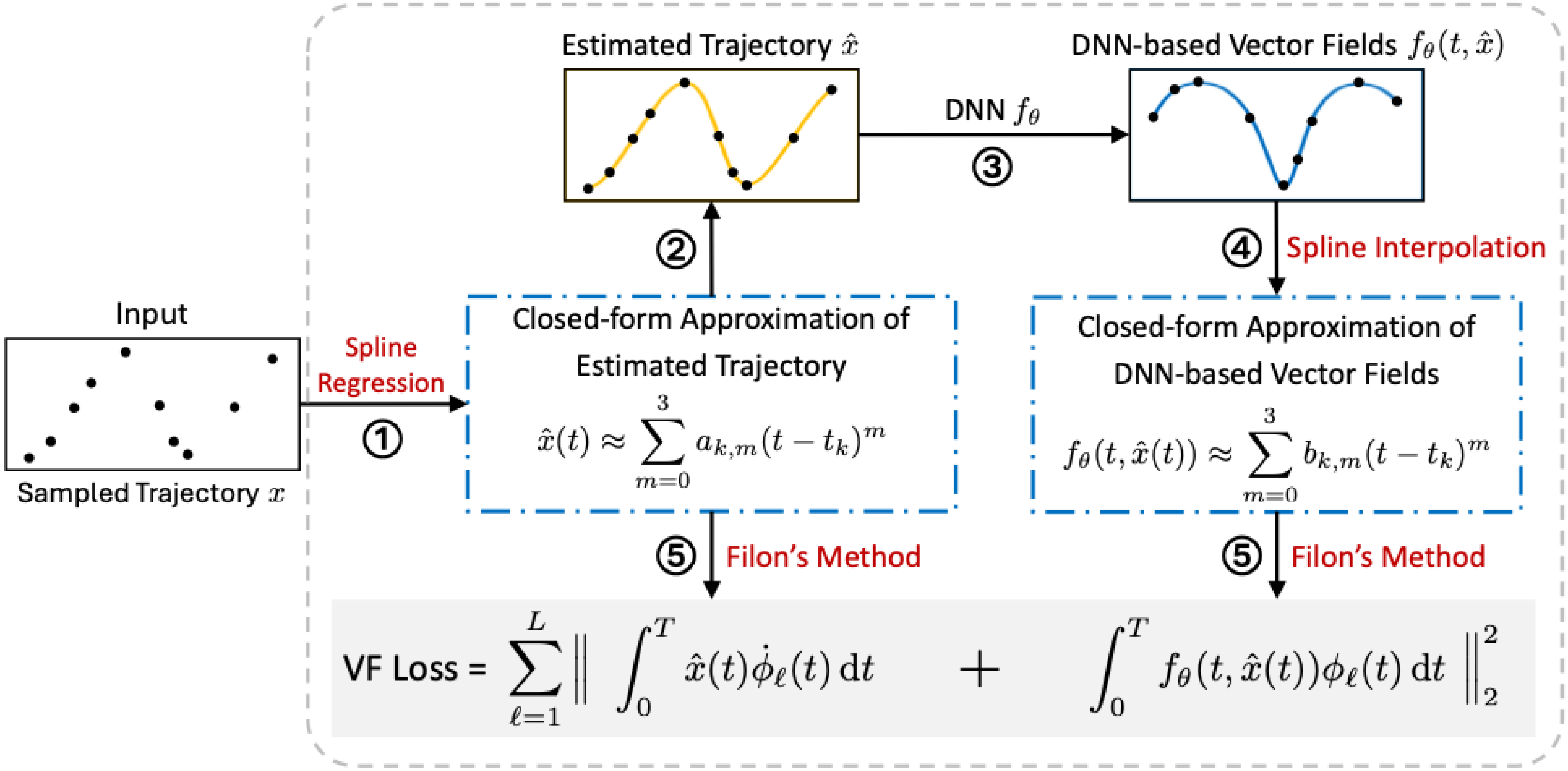
Filon’s Method with Spline Regression

- Key Challenge: compute integrals with sine function based on noisy and irregular data
- We introduce Filon’s method with spline regression to address this issue
- Filon’s method is designed to deal with integrals including oscillatory terms:

$$\int_0^T h(t)\sin(\omega t)\,\mathrm{d}t = \sum_{k=0}^{K-1}\int_{t_k}^{t_{k+1}} h(t)\sin(\omega t)\,\mathrm{d}t \approx \sum_{k=0}^{K-1}\int_{t_k}^{t_{k+1}} q_k(t)\sin(\omega t)\,\mathrm{d}t$$

cubic spline approximation of $h(t)$ on $[t_k, t_{k+1}]$

VF-NODEs



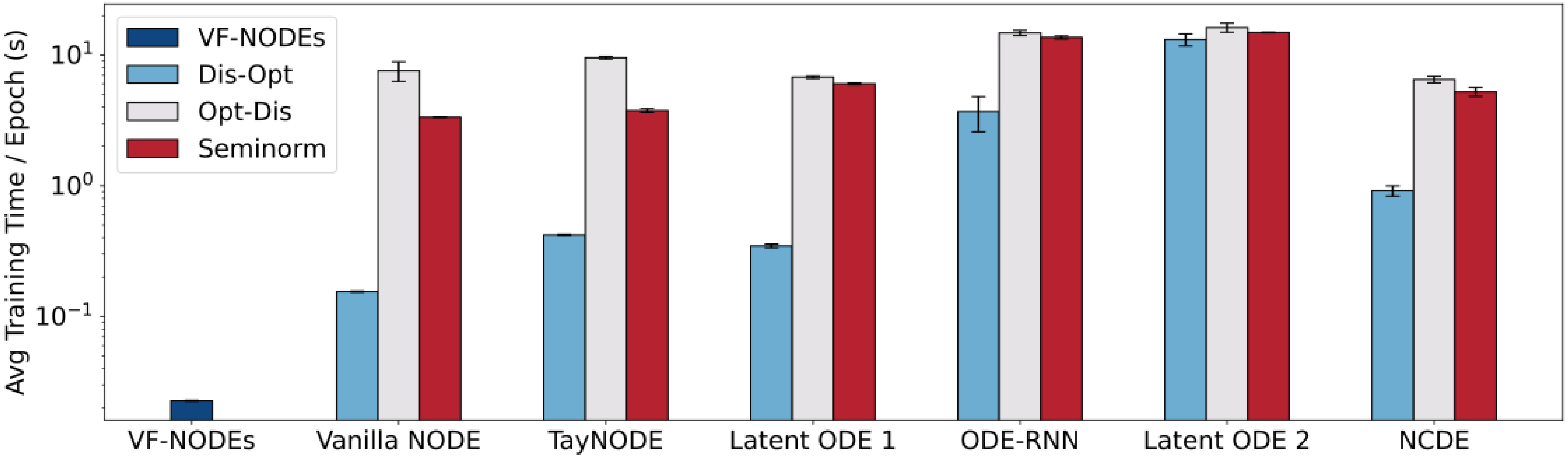
$\phi_\ell(t) = \sqrt{2/T}\sin(\pi \ell t/T)$: Hilbert orthonormal basis

Analysis for the Acceleration of VF-NODEs

- Reducing the Number of Function Evaluations:
  - Based on ODE solvers, Vanilla NODEs must evaluate the vector field beyond sampled data points for accuracy
  - Filon's method in VF-NODEs only need to evaluate the vector field at given points
- Improving parallelizability:
  - Due to the autoregressive nature of ODE solvers, these vector fields must be evaluated step by step
  - Based on numerical integration techniques, these vector fields can be evaluated simultaneously

Experimental Results

Average training time per epoch (second) on the glycolytic oscillator



Our method can accelerate the training 10-1000 times.

Experimental Results (Continued)

Testing MSE for interpolation tasks on four dynamical systems with 80% observed data

	Glycolytic	Toggle	Repressilator	AgeSIR
Vanilla NODE	(1.51e-03)±(1.40e-03)	**(8.00e-04)±(8.69e-04)**	**(2.25e-02)±(5.04e-03)**	**(7.54e-03)±(6.58e-04)**
TayNODE	(3.20e-03)±(1.32e-03)	(1.37e-02)±(9.96e-03)	(1.43e-01)±(1.72e-02)	(3.18e-01)±(7.36e-02)
Latent ODE 1	(2.21e-01)±(3.35e-02)	(6.57e-01)±(1.48e-01)	(2.33e+01)±(9.23e-01)	(4.18e+01)±(6.56e+00)
ODE-RNN	**(8.80e-05)±(2.30e-05)**	(1.63e-03)±(5.56e-04)	(1.41e-01)±(7.97e-03)	(6.75e+00)±(3.43e-01)
Latent ODE 2	(1.00e-01)±(1.87e-02)	(6.31e-01)±(4.69e-01)	(7.47e-01)±(8.77e-02)	(1.36e+09)±(1.93e+09)
NCDE	(3.49e-02)±(1.36e-02)	(3.96e-02)±(3.43e-02)	(1.51e+00)±(1.22e+00)	(1.22e+01)±(3.75e+00)
ResNet Flow	(2.84e-01)±(5.69e-02)	(7.09e-01)±(2.21e-01)	(1.03e+01)±(8.14e-01)	(2.50e+00)±(1.96e-01)
GRU Flow	(3.80e-01)±(5.34e-02)	(2.45e+00)±(1.88e-01)	(7.45e+00)±(4.68e-02)	(4.19e+01)±(1.22e-01)
VF-NODE (Ours)	**(6.35e-05)±(2.68e-06)**	**(1.69e-04)±(6.09e-05)**	**(1.92e-02)±(2.62e-04)**	**(7.39e-03)±(6.71e-04)**

Testing MSE for extrapolation tasks on four dynamical systems with 80% observed data

	Glycolytic	Toggle	Repressilator	AgeSIR
Vanilla NODE	(8.79e-04)±(7.64e-04)	**(8.14e-07)±(6.72e-07)**	**(1.25e-01)±(3.11e-02)**	**(1.99e-02)±(1.69e-03)**
TayNODE	(4.71e-03)±(3.60e-03)	(5.09e-02)±(5.09e-02)	(8.73e-01)±(1.35e-01)	(4.52e-01)±(1.31e-01)
Latent ODE 1	(2.39e-01)±(7.29e-02)	(1.48e+00)±(1.28e+00)	(9.14e+00)±(1.19e+00)	(2.30e+02)±(4.85e+01)
ODE-RNN	**(4.68e-05)±(1.15e-05)**	(2.04e-04)±(1.42e-04)	(2.02e-01)±(7.51e-03)	(7.48e+00)±(9.12e-02)
Latent ODE 2	(1.82e-01)±(1.61e-01)	(4.96e+00)±(5.70e+00)	(3.09e+00)±(2.99e-01)	(1.36e+09)±(1.93e+09)
NCDE	(8.00e-01)±(4.48e-01)	(1.03e+00)±(7.38e-01)	(6.73e+00)±(4.85e+00)	(2.13e+01)±(8.27e+00)
ResNet Flow	(3.47e+00)±(2.82e+00)	(5.32e+00)±(1.97e+00)	(6.56e+01)±(2.23e+01)	(1.95e+01)±(3.29e-01)
GRU Flow	(7.39e-01)±(2.23e-01)	(5.03e+00)±(5.22e-01)	(1.84e+01)±(5.74e-01)	(6.02e+01)±(1.89e-01)
VF NODE (Ours)	**(1.63e-04)±(3.05e-05)**	**(4.79e-07)±(5.24e-08)**	**(1.23e-01)±(1.48e-02)**	**(2.37e-02)±(1.61e-03)**

While significantly accelerating the training of Neural ODEs, our method can still achieve better or comparable accuracy

Experimental Results (Continued)

Testing MSE on the real-world COVID-19 dataset [2]

	Japan	Italy	Norway	India
Vanilla NODE	(1.42e+00)±(7.44e-01)	(1.35e-02)±(1.39e-02)	(1.03e-03)±(6.30e-05)	(8.86e-04)±(3.76e-04)
TayNODE	(2.02e+00)±(8.00e-01)	(3.88e-02)±(5.70e-03)	(5.57e-04)±(1.13e-04)	(1.18e-02)±(1.08e-02)
Latent ODE 1	(1.02e+01)±(4.42e-01)	(6.56e-01)±(2.87e-01)	(1.83e-01)±(1.10e-01)	(5.69e-01)±(1.59e-01)
ODE-RNN	(8.43e+00)±(6.56e-01)	(1.10e-01)±(1.12e-02)	(5.58e-03)±(1.48e-03)	(2.09e-01)±(1.74e-01)
Latent ODE 2	(1.12e+01)±(1.31e+00)	(3.36e-01)±(2.70e-01)	(4.12e-01)±(2.90e-01)	(4.07e-01)±(1.54e-01)
NCDE	(1.14e+01)±(1.10e+00)	(8.72e-01)±(2.46e-01)	(1.27e-02)±(1.20e-02)	(3.94e-01)±(8.18e-02)
ResNet Flow	(9.33e-01)±(1.69e-01)	(3.69e-02)±(3.20e-03)	(2.51e-02)±(1.65e-02)	(2.04e-02)±(2.01e-03)
GRU Flow	(1.39e+00)±(2.96e-02)	(8.63e-03)±(1.17e-04)	(3.07e-03)±(3.07e-07)	(2.52e-03)±(1.84e-04)
VF NODE (Ours)	**(1.87e-01)±(4.82e-02)**	**(1.64e-03)±(2.19e-04)**	**(3.43e-04)±(1.18e-04)**	**(5.68e-04)±(2.28e-04)**

Our method achieves significantly better performance compared to the baselines.

[2] Guidotti, Emanuele, and David Ardia. "COVID-19 data hub." *Journal of Open Source Software* 5.51 (2020): 2376.

Conclusion

- We introduced a novel variational formulation-based training method for Neural ODEs
  - Specifically, we introduced the VF loss into the training of Neural ODEs
  - Significantly reduce NFEs & Mitigate the auto-regression
- We employed Filon's method and spline regression to handle oscillatory integrals in VF-NODEs
- Evaluations showed our method speeds up training by 10–1000× compared to baselines, achieving comparable or better accuracy

Thank you!