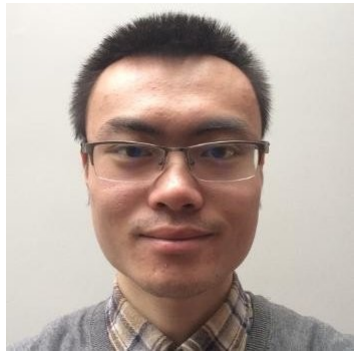
Neural Laplace: Learning diverse classes of differential equations in the Laplace domain

Samuel Holt



Zhaozhi Qian

Mihaela van der Schaar

ICML 2022

Long Oral Presentation

Problem

Model	Equation
ODE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t))$

Problem

Table 1. Families of DEs captured by Neural Laplace.

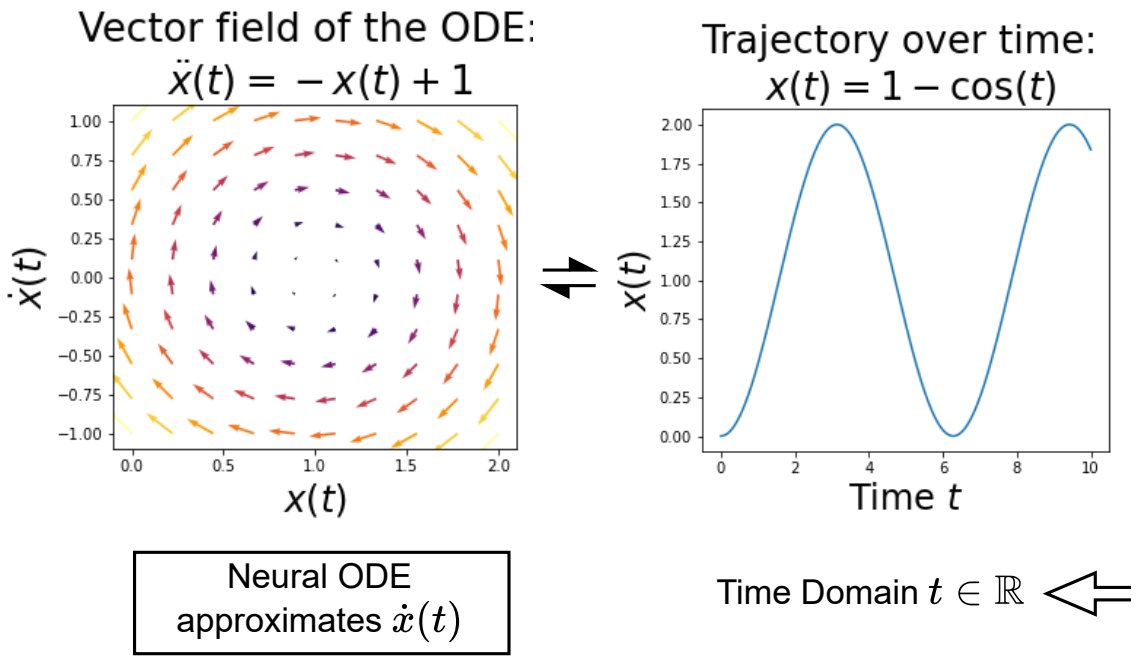
Model	Equation
ODE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t))$
DDE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t), \mathbf{x}(t-\tau)), \tau \in \mathbb{R}^+$
IDE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t)) + \int_0^t \mathbf{h}(\tau, \mathbf{x}(\tau))d\tau$
Forced ODE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t))$
Stiff ODE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t)), \exists\ i, j, \dot{x}_i \gg \dot{x}_j$

Problem

Table 1. Families of DEs captured by Neural Laplace.

Model	Equation
ODE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t))$
DDE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t), \mathbf{x}(t-\tau)), \tau \in \mathbb{R}^+$
IDE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t)) + \int_0^t \mathbf{h}(\tau, \mathbf{x}(\tau))d\tau$
Forced ODE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t))$
Stiff ODE	$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t)), \exists\, i, j, \dot{x}_i \gg \dot{x}_j$

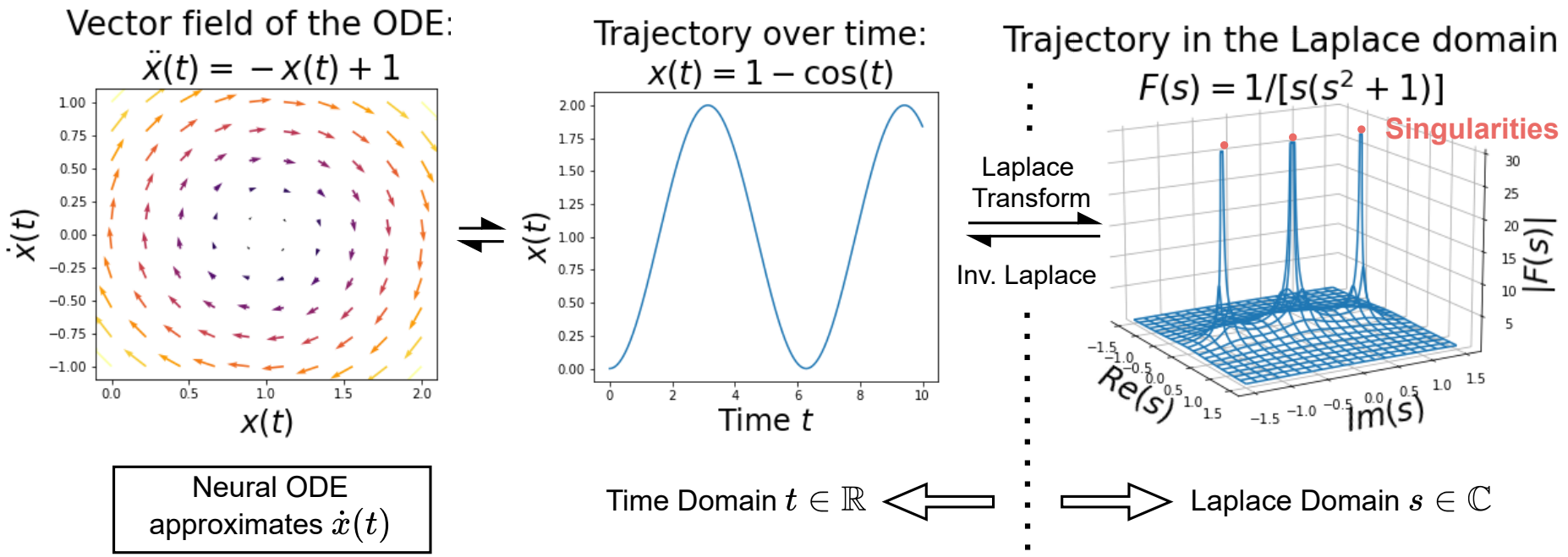
Neural Laplace



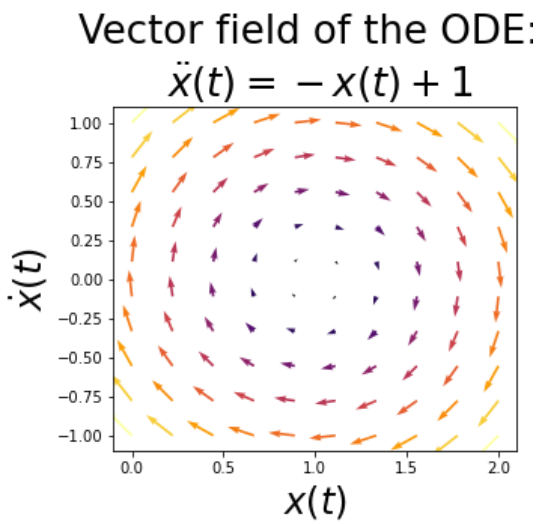
Neural ODE approximates $\dot{x}(t)$

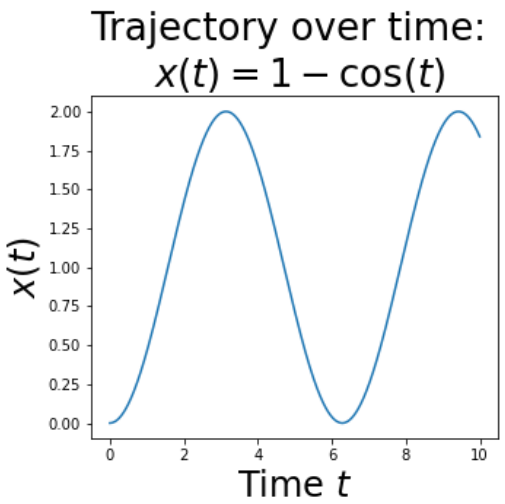
Time Domain $t \in \mathbb{R}$ ⇦

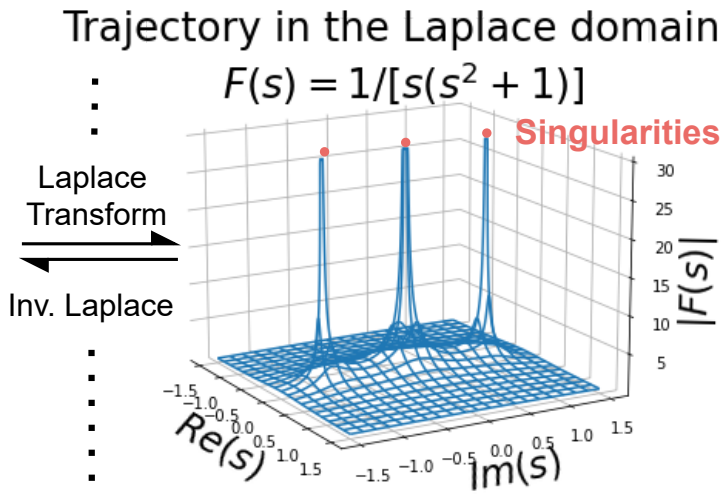
Neural Laplace

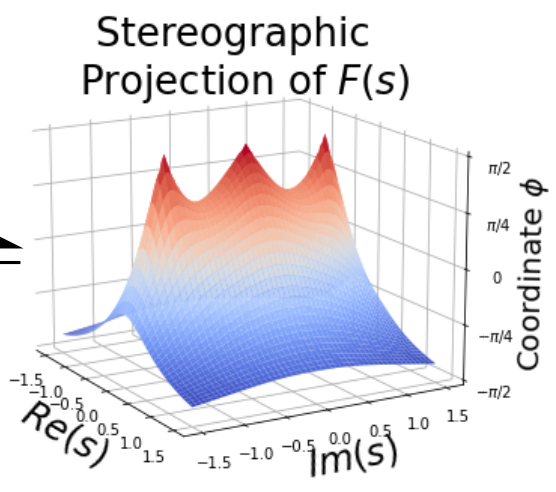


Neural Laplace







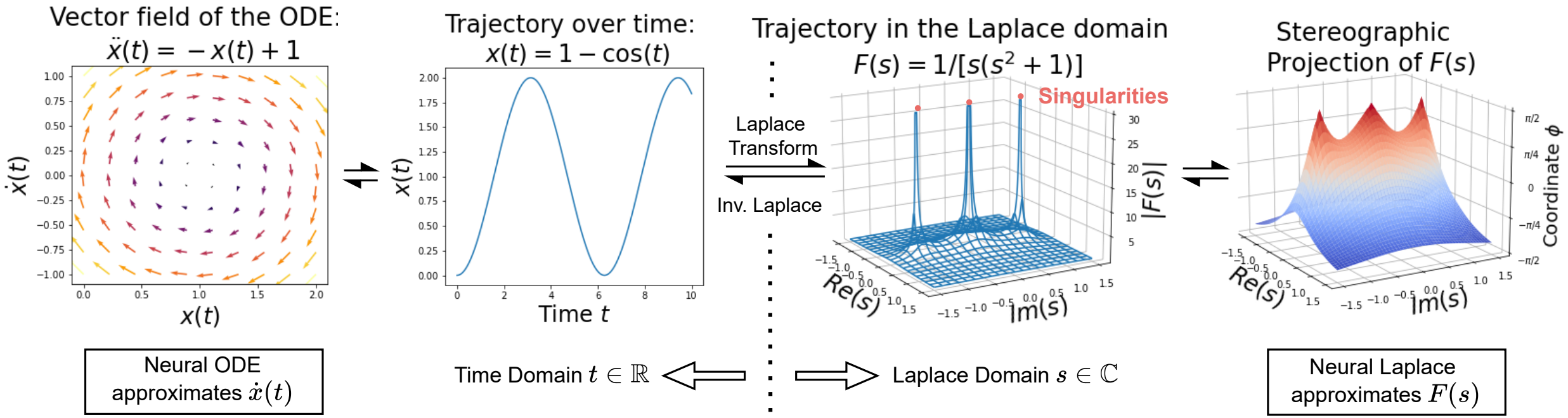


Neural ODE approximates $\dot{x}(t)$

Time Domain $t \in \mathbb{R}$ ⟸ ⋮ ⟹ Laplace Domain $s \in \mathbb{C}$

Neural Laplace approximates $F(s)$

Neural Laplace



- Models the DE in the Laplace domain.
- This *bypasses* the standard numerical ODE solver, constructing the time solution $x(t)$ with goal summations of complex exponentials

Related work

Table 2. Comparison with existing works. Neural Laplace is a unified framework of learning diverse classes of DEs.

Method	Reference	Quantity Modeled	Initial Condition	Representable classes of DEs				
				ODE	DDE	IDE	Forced DE	Stiff DE
Neural ODE	Chen et al. (2018)	$\dot{\mathbf{x}}(t)$	$\mathbf{x}(0)$	✓	✗	✗	✗	✗
ANODE	Dupont et al. (2019)	$\dot{\mathbf{x}}(t), \dot{\mathbf{z}}(t)$	$\mathbf{x}(0), \mathbf{0}$	✓	✗	✗	✗	✗
Latent ODE	Rubanova et al. (2019)	$\dot{\mathbf{z}}(t)$	$\mathbf{z}(0)$	✓	✗	✗	✗	✗
ODE2VAE	Yildiz et al. (2019)	$\dot{\mathbf{x}}(t), \ddot{\mathbf{x}}(t)$	$\mathbf{x}(0), \dot{\mathbf{x}}(0)$	✓	✗	✗	✗	✗
Neural DDE	Zhu et al. (2020)	$\dot{\mathbf{x}}(t)$	$\mathbf{x}(t), t \leq 0$	✗	✓	✗	✗	✗
Neural Flow	Biloš et al. (2021)	$\mathbf{x}(t)$	$\mathbf{x}(0)$	✓	✗	✗	✗	✓
Neural IM	Gwak et al. (2020)	$\dot{\mathbf{x}}(t)$	$\mathbf{x}(0)$	✓	✗	✗	✓	✗
Neural Laplace	This work	$\mathbf{F}(\mathbf{s})$	$\mathbf{p}$	✓	✓	✓	✓	✓

Problem & Background

- **Laplace Transform.** The Laplace transform of $\mathbf{x}$ is

$$\mathbf{F}(\mathbf{s}) = \mathcal{L}\{\mathbf{x}\}(\mathbf{s}) = \int_0^\infty e^{-\mathbf{s}t}\mathbf{x}(t)dt$$

- Where $\mathbf{s} \in \mathbb{C}^d$ is a vector of *complex* numbers and $\mathbf{F}(\mathbf{s}) \in \mathbb{C}^d$ is the *Laplace representation*.

Problem & Background

- **Inverse Laplace Transform.** Is defined as

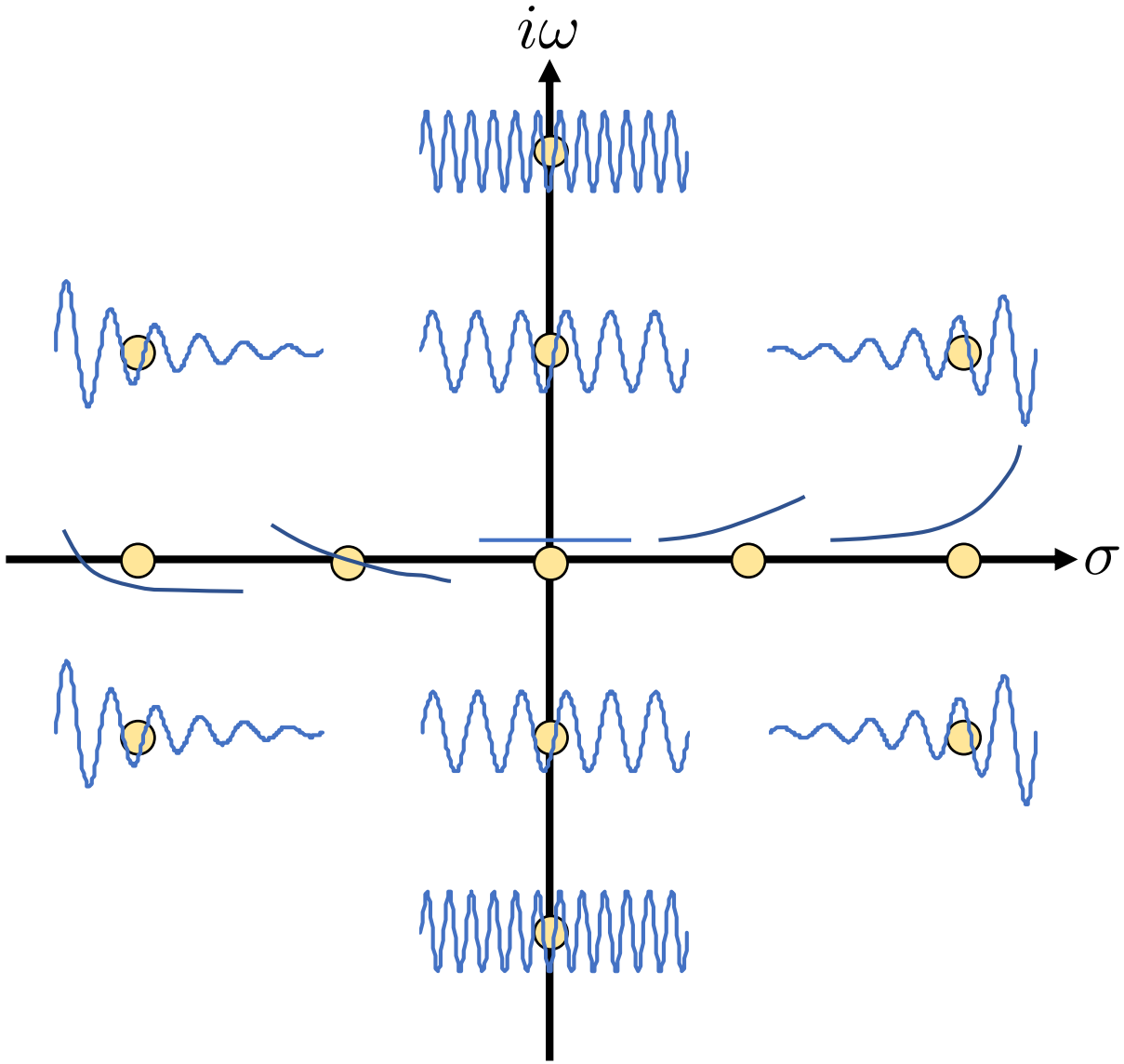
$$\hat{\mathbf{x}}(t) = \mathcal{L}^{-1}\{\mathbf{F}(\mathbf{s})\}(t) = \frac{1}{2\pi i}\int_{\sigma-i\infty}^{\sigma+i\infty} \mathbf{F}(\mathbf{s})e^{\mathbf{s}t}d\mathbf{s}$$

- where the integral refers to the Bromwich contour integral in $\mathbb{C}^d$ with the contour $\sigma > 0$ chosen such that all the singularities of $\mathbf{F}(\mathbf{s})$ are to the left of it.

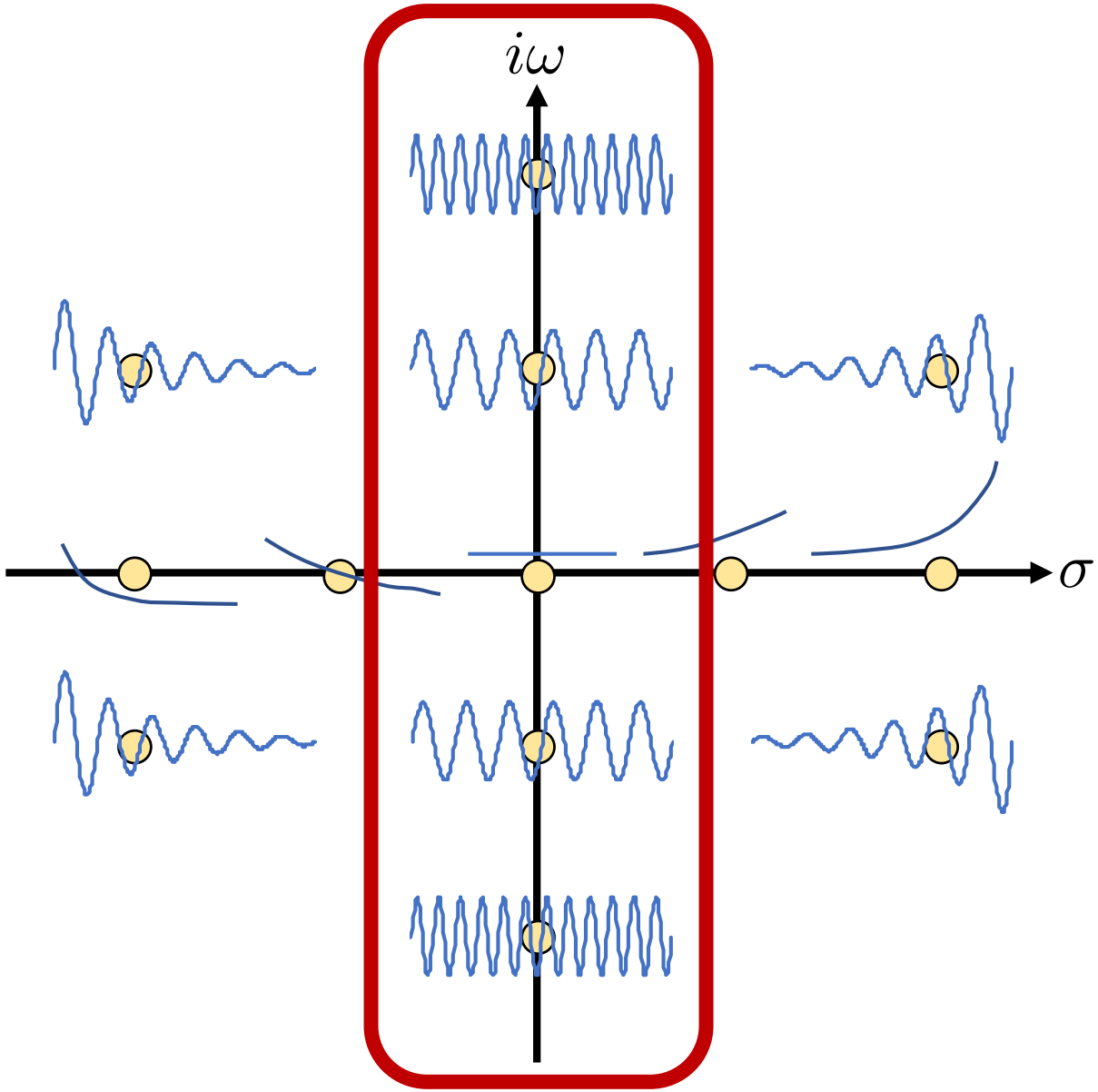
Problem & Background

$$\mathcal{Q}(t) = \text{ILT-Query}(t)$$
$$\hat{\mathbf{x}}(t) = \text{ILT-Compute}\big(\{\mathbf{F}(\mathbf{s}) | \mathbf{s} \in \mathcal{Q}(t)\}\big)$$

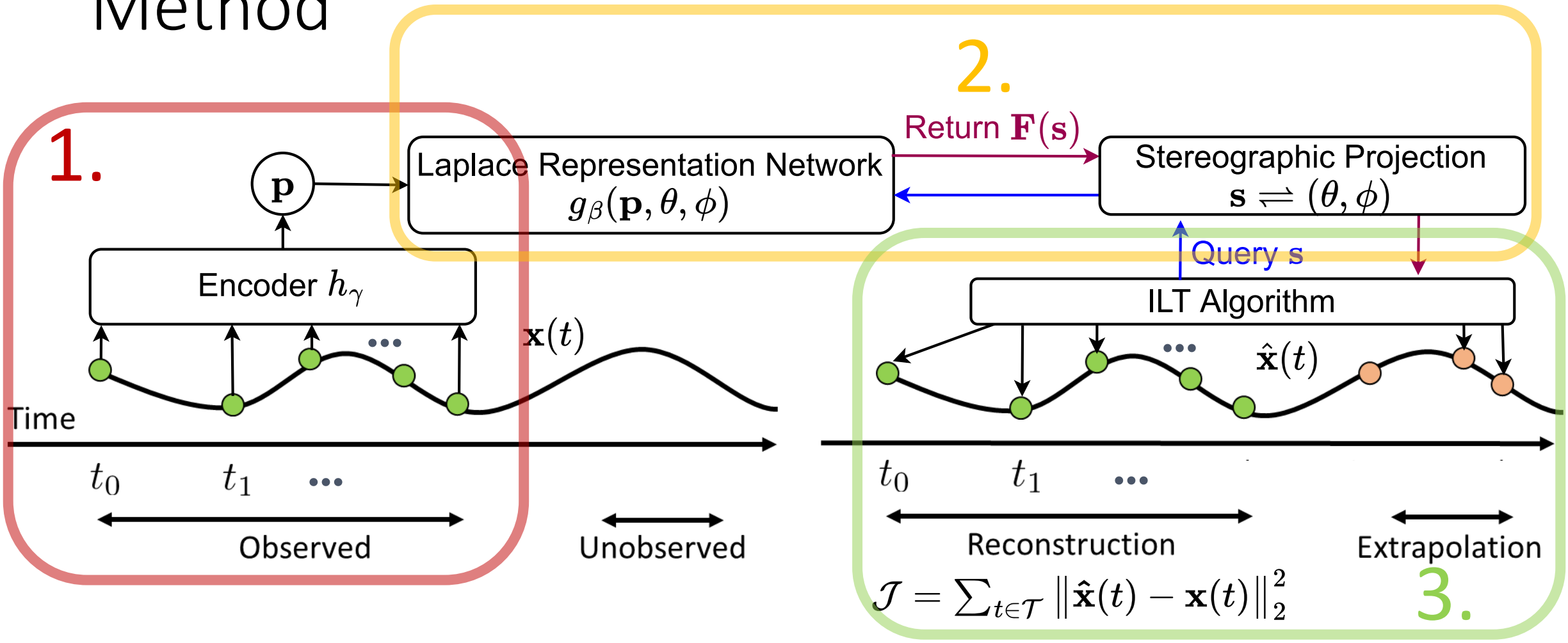
$$\hat{\mathbf{x}}(t) = \mathcal{L}^{-1}\{\mathbf{F}(\mathbf{s})\}(t) = \frac{1}{2\pi i}\int_{\sigma-i\infty}^{\sigma+i\infty} \mathbf{F}(\mathbf{s})e^{\mathbf{s}t}d\mathbf{s}, \quad (3)$$



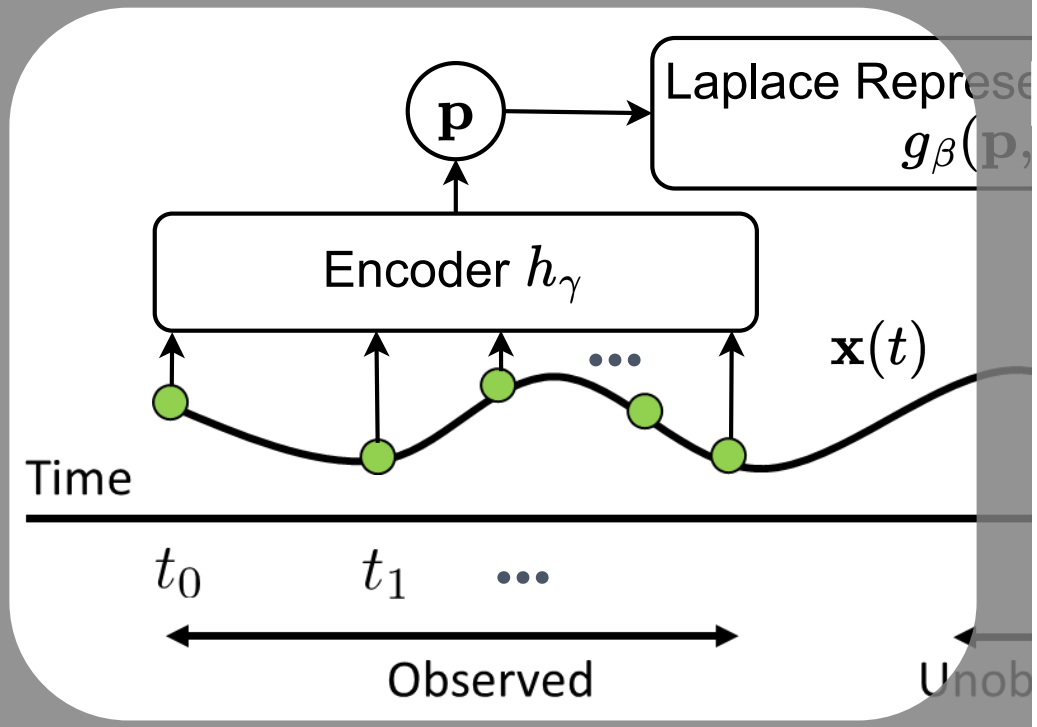
$$\hat{\mathbf{x}}(t) = \mathcal{L}^{-1}\{\mathbf{F}(\mathbf{s})\}(t) = \frac{1}{2\pi i}\int_{\sigma-i\infty}^{\sigma+i\infty} \mathbf{F}(\mathbf{s})e^{\mathbf{s}t}d\mathbf{s}, \quad (3)$$



Method


1.
2.
3.
$\mathbf{p}$
Laplace Representation Network
$g_\beta(\mathbf{p}, \theta, \phi)$
Return $\mathbf{F}(\mathbf{s})$
Stereographic Projection
$\mathbf{s} \rightleftharpoons (\theta, \phi)$
Query s
Encoder h_γ
ILT Algorithm
$\mathbf{x}(t)$
$\hat{\mathbf{x}}(t)$
Time
t_0
t_1
...
Observed
Unobserved
Reconstruction
Extrapolation
$\mathcal{J} = \sum_{t \in \mathcal{T}} \left\| \hat{\mathbf{x}}(t) - \mathbf{x}(t) \right\|_2^2$

Method

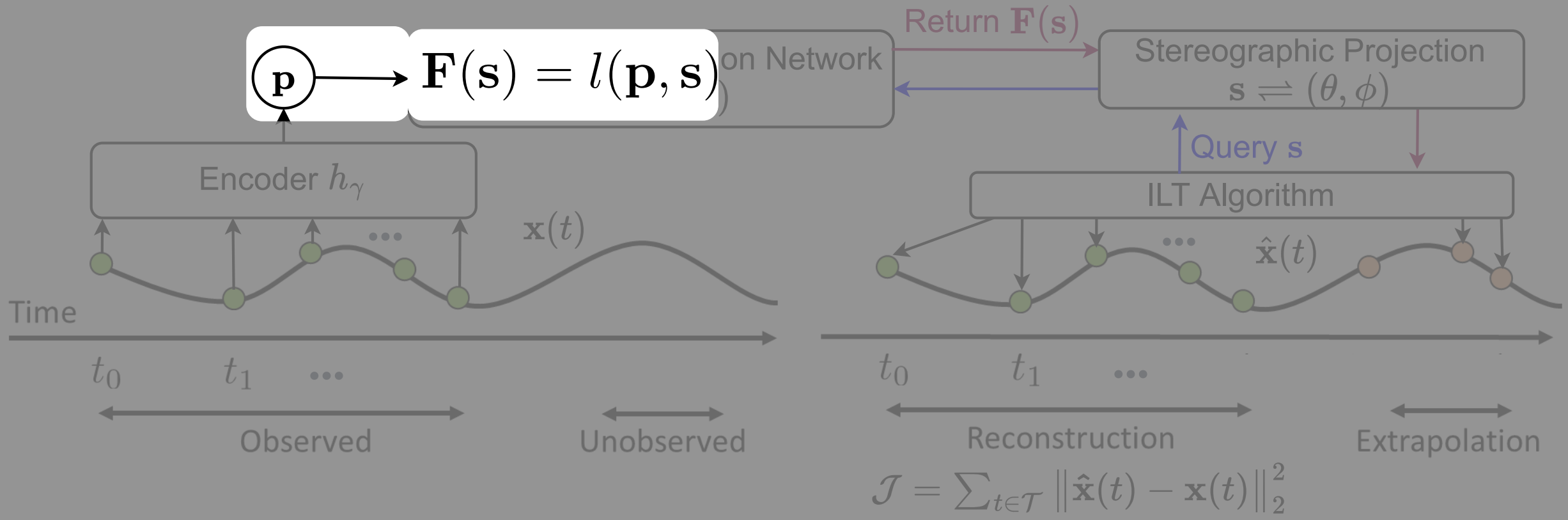


The encoder. Uses an encoder network to learn a ***representation*** of the initial condition.

$$\mathbf{p} = h_\gamma((\mathbf{x}(t_1), t_1), (\mathbf{x}(t_2), t_2), \ldots, (\mathbf{x}(T), T))$$

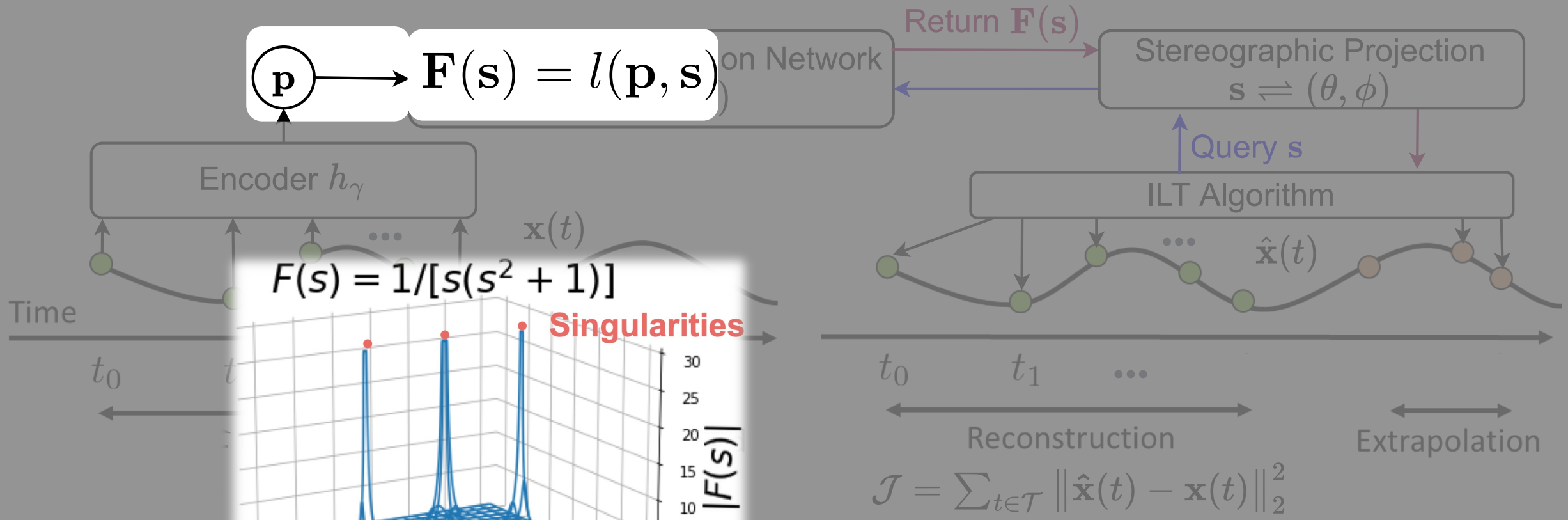
The vector $\mathbf{p} \in \mathbb{R}^K$ is the learned initial condition representation, where $K \geq D$ is a hyper-parameter. The encoder h has trainable weights γ .

Method



$$\mathcal{J} = \sum_{t \in \mathcal{T}} \left\| \hat{\mathbf{x}}(t) - \mathbf{x}(t) \right\|_2^2$$

Method

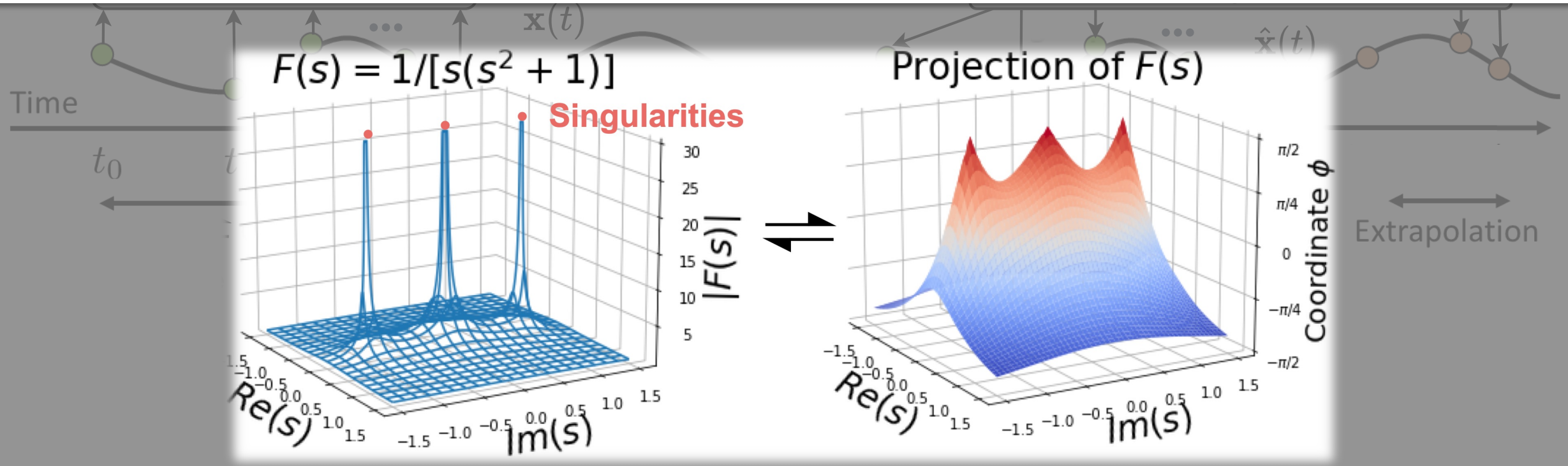

$\mathbf{p}$
$\mathbf{F}(\mathbf{s}) = l(\mathbf{p}, \mathbf{s})$
on Network
Return $\mathbf{F}(\mathbf{s})$
Stereographic Projection
$\mathbf{s} \rightleftharpoons (\theta, \phi)$
Query s
Encoder h_γ
ILT Algorithm
$\mathbf{x}(t)$
$\hat{\mathbf{x}}(t)$
Time
t_0
t_1
Reconstruction
Extrapolation
$\mathcal{J} = \sum_{t \in \mathcal{T}} \|\hat{\mathbf{x}}(t) - \mathbf{x}(t)\|_2^2$

$F(s) = 1/[s(s^2 + 1)]$
Singularities
|F(s)|
Re(s)
Im(s)

We use a stereographic projection of a Riemann sphere. This translates any complex number $s \in \mathbb{C}$ into a coordinate on the Riemann sphere, i.e.

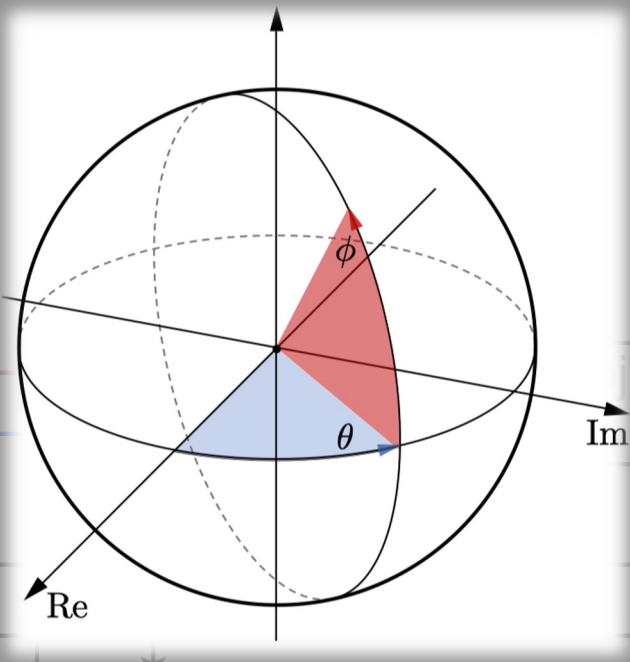
$$(\theta, \phi) \in \mathcal{D} = (-\pi, \pi) \times (-\frac{\pi}{2}, \frac{\pi}{2})$$

$$u(s) = \left(\arctan\left(\frac{\Im(s)}{\Re(s)}\right), \arcsin\left(\frac{|s|^2 - 1}{|s|^2 + 1}\right)\right) \qquad s = v(\theta, \phi) = \tan\left(\frac{\phi}{2} + \frac{\pi}{4}\right) e^{i\theta}$$



Method

$$\mathbf{p} \longrightarrow \mathbf{F}(\mathbf{s}) = v\Big(g_\beta\big(\mathbf{p}, u(\mathbf{s})\big)\Big)$$



Encoder h_γ

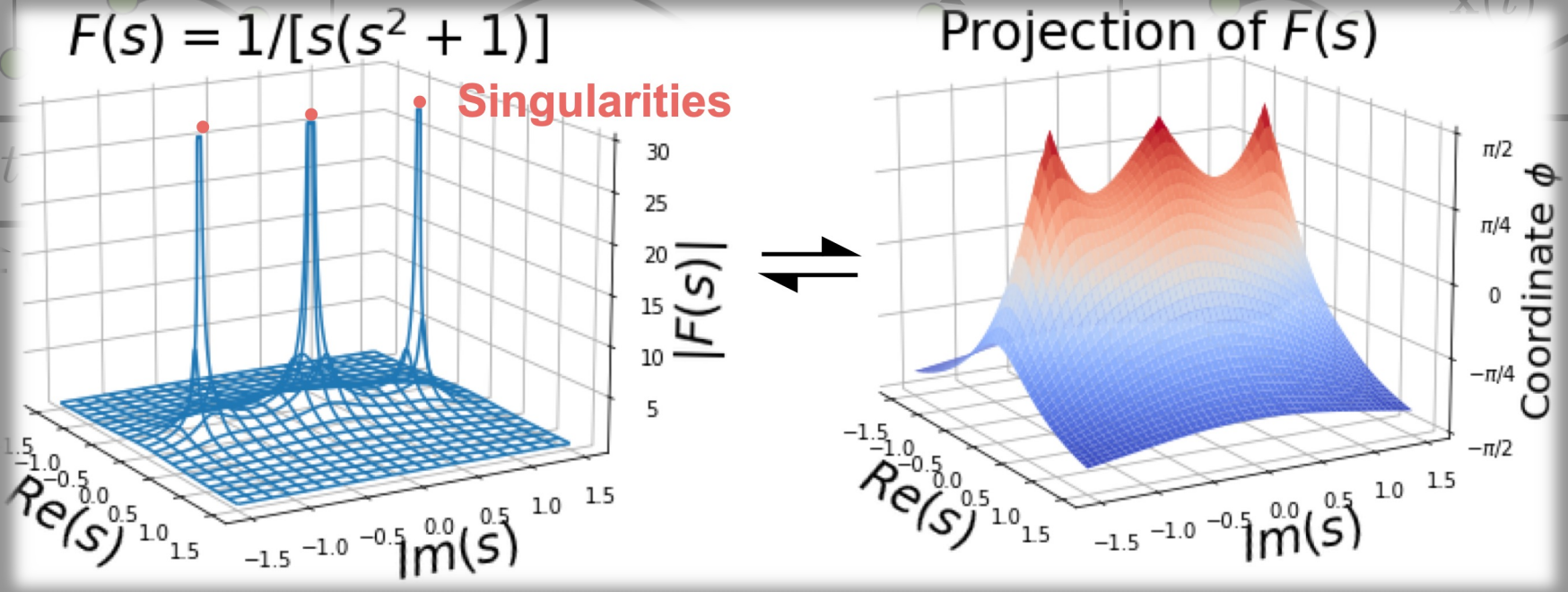
$\mathbf{x}(t)$

$\hat{\mathbf{x}}(t)$

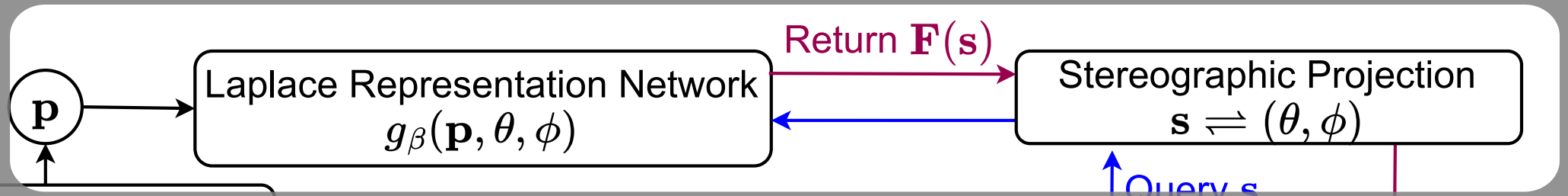
Time

t_0

Extrapolation



Method



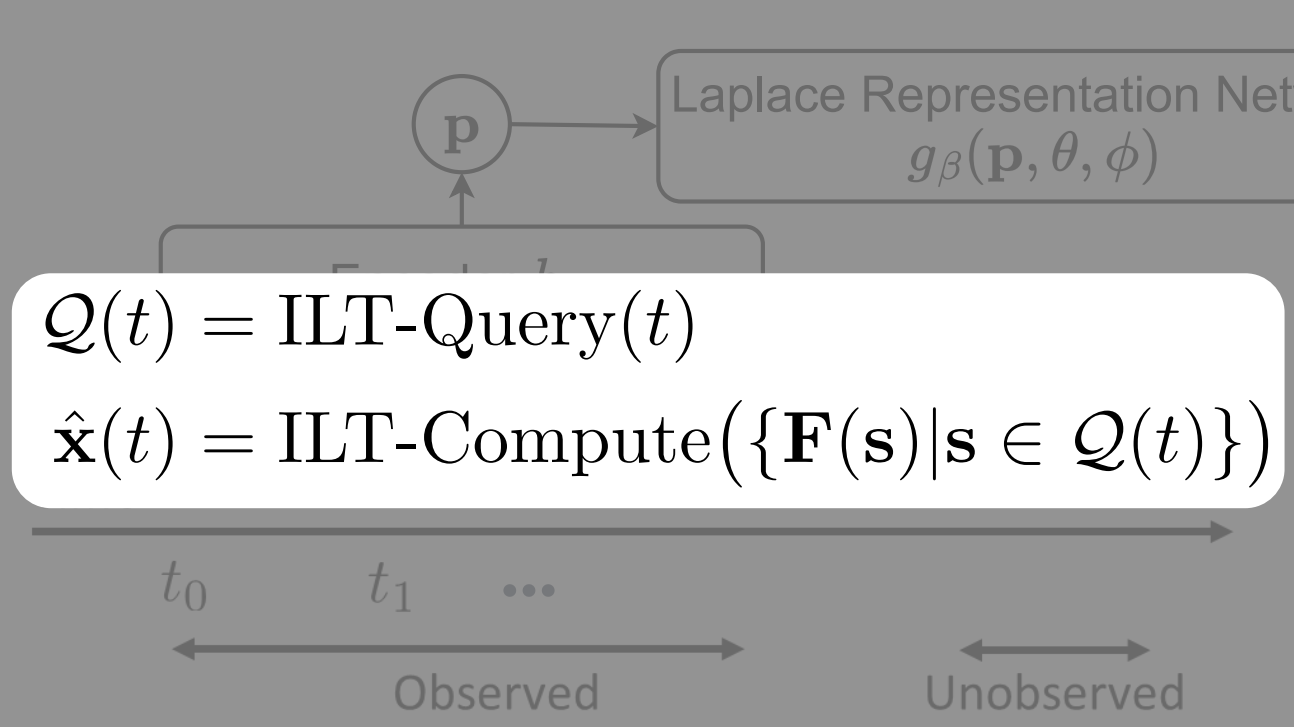
With the stereographic projection, we introduce a feed-forward neural network g to learn the Laplace representation of the DE solution.

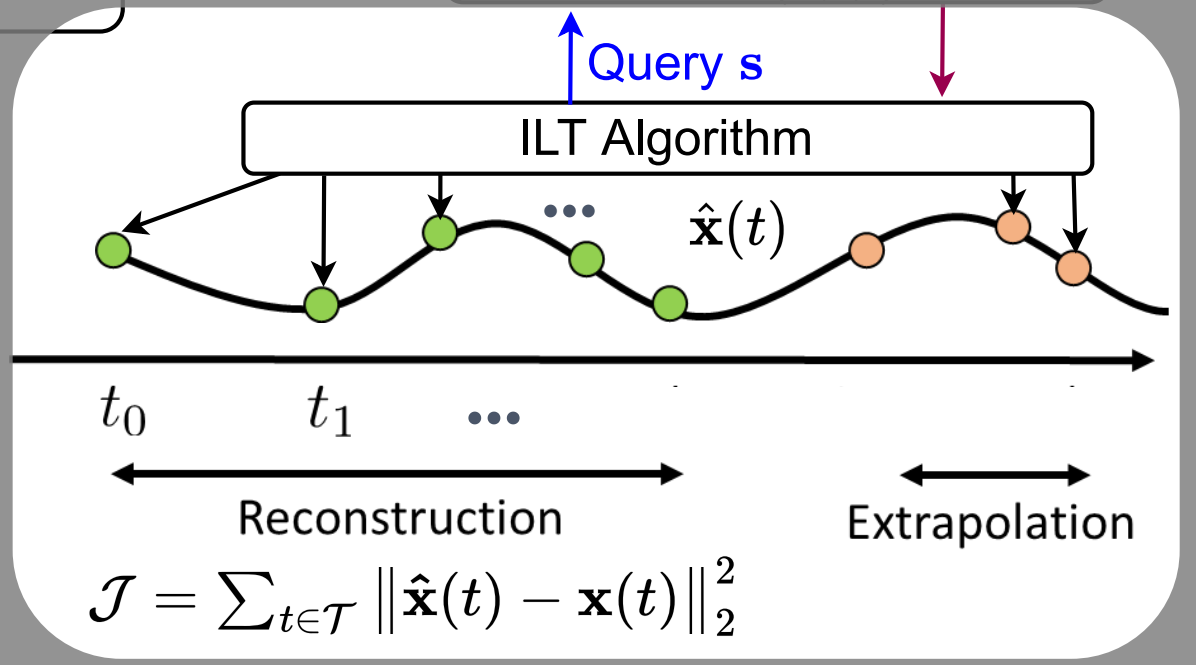
$$\mathbf{F}(\mathbf{s}) = v\Big(g_\beta\big(\mathbf{p}, u(\mathbf{s})\big)\Big)$$

$$u(s) = \left(\arctan\left(\frac{\Im(s)}{\Re(s)}\right), \arcsin\left(\frac{|s|^2 - 1}{|s|^2 + 1}\right)\right)$$

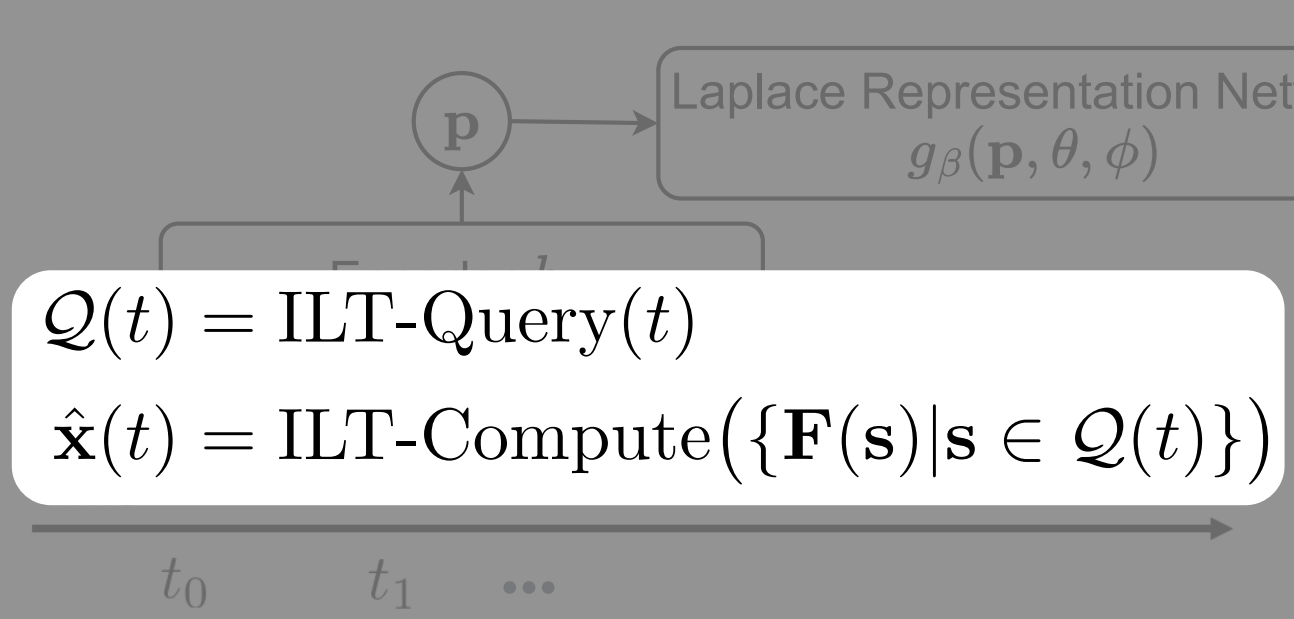
$$s = v(\theta, \phi) = \tan\left(\frac{\phi}{2} + \frac{\pi}{4}\right) e^{i\theta}$$

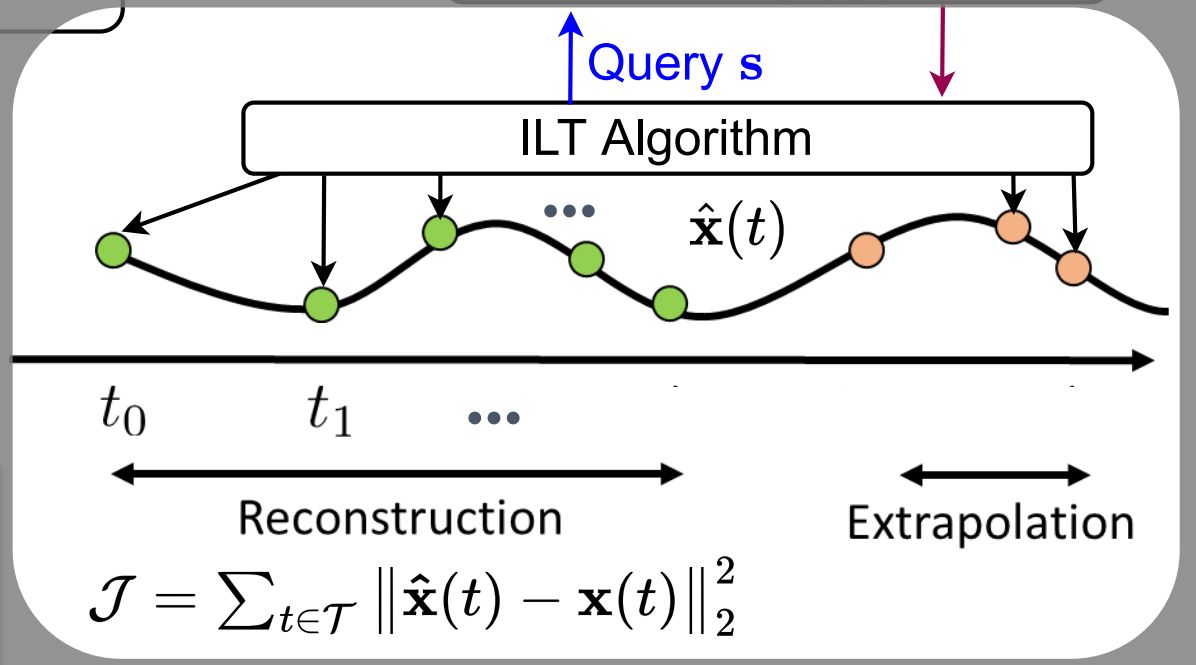
Method


p
Laplace Representation Network
$g_\beta(\mathbf{p}, \theta, \phi)$
Return $\mathbf{F}(\mathbf{s})$
Stereographic Projection
$\mathbf{s} \rightleftharpoons (\theta, \phi)$
$\mathcal{Q}(t) = \text{ILT-Query}(t)$
$\hat{\mathbf{x}}(t) = \text{ILT-Compute}(\{\mathbf{F}(\mathbf{s}) | \mathbf{s} \in \mathcal{Q}(t)\})$
t_0
t_1
Observed
Unobserved


Query s
ILT Algorithm
$\hat{\mathbf{x}}(t)$
t_0
t_1
Reconstruction
Extrapolation
$\mathcal{J} = \sum_{t \in \mathcal{T}} \|\hat{\mathbf{x}}(t) - \mathbf{x}(t)\|_2^2$

Method





We can evaluate $\hat{\mathbf{x}}(t)$ *at any time* $t \in \mathbb{R}$ as the Laplace representation is independent of time once learnt.

Method

Algorithm 1 Neural Laplace Training Procedure

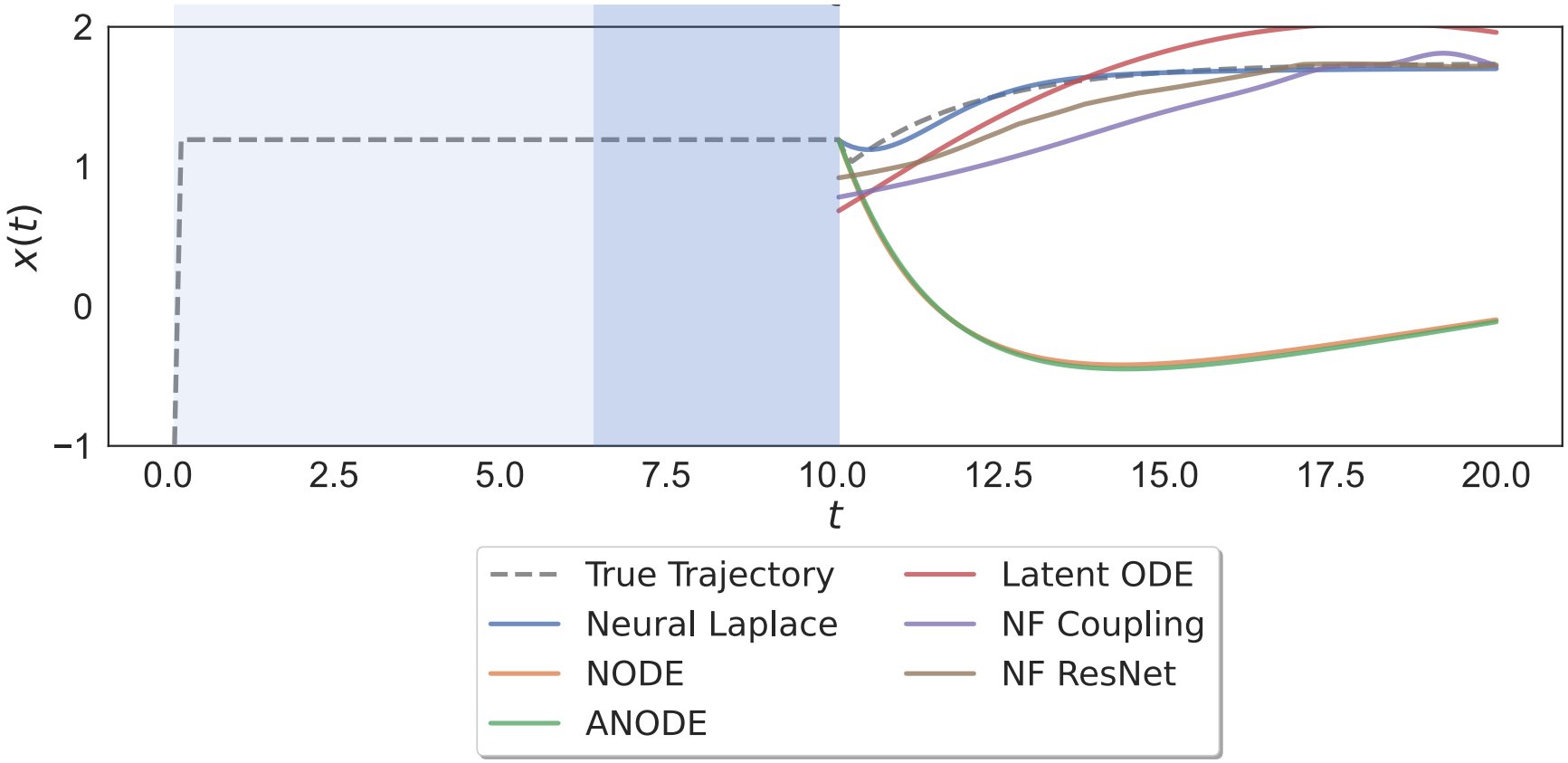
Input: Observed trajectory $\mathbf{x}(t), t \in \mathcal{T}$
Input: Training iterations I
for $i \in 1 : I$ **do**
  $\mathbf{p} = h_\gamma\big(\{\mathbf{x}(t) | t \in \mathcal{T}\}\big)$ Eq. 6
  for $t \in \mathcal{T}$ **do**
    $\mathcal{Q} = \text{ILT-Query}(t)$ Eq. 4
    for $s \in \mathcal{Q}$ **do**
      $\mathbf{F}(\mathbf{s}) = v\Big(g_\beta\big(\mathbf{p}, u(\mathbf{s})\big)\Big)$ Eq. 9
    end for
    $\hat{\mathbf{x}}(t) = \text{ILT-Compute}\big(\{\mathbf{F}(\mathbf{s}) | \mathbf{s} \in \mathcal{Q}\}\big)$ Eq. 5
  end for
  $\mathcal{J} = \sum_{t \in \mathcal{T}} \|\hat{\mathbf{x}}(t) - \mathbf{x}(t)\|_2^2$ Eq. 10
  Compute gradients of $\mathcal{J}$ via back propagation.
  Update neural network weights γ and β.
end for
Output: the trained neural networks h_γ and g_β

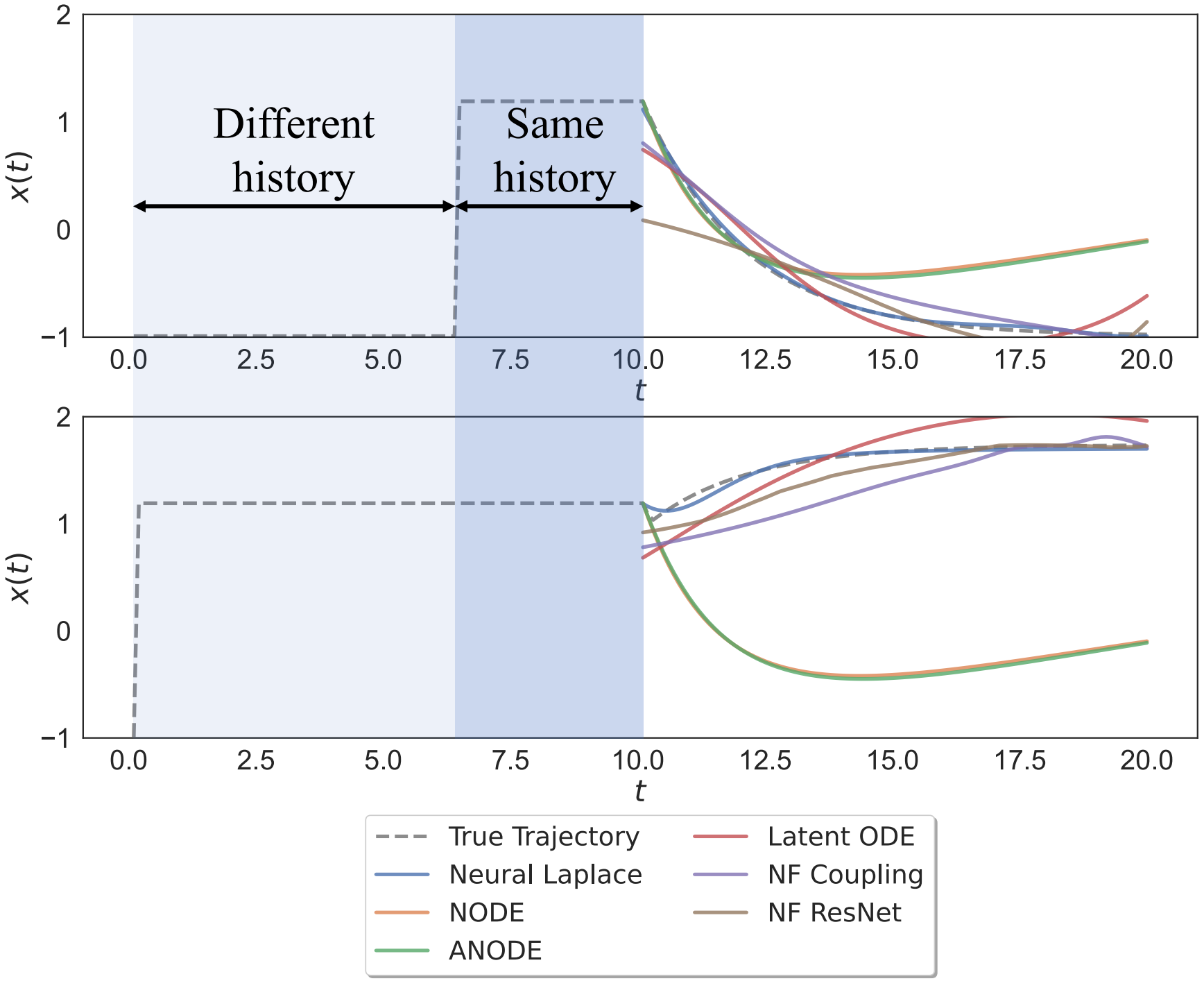
Experiments

Table 3. Each DE system we use for comparison against the benchmarks, and their properties for comparison.

System	Piecewise DE	Discontinuous differential	Integro DE	Delay DE	Stiff solutions	Periodic solutions
Spiral DDE	✓	✓	✗	✓	✗	✗
Lotka-Volterra DDE	✓	✓	✗	✓	✗	✗
Mackey–Glass DDE	✓	✓	✗	✓	✗	✗
Stiff Van der Pol Oscillator DE	✗	✓	✗	✗	✓	✓
ODE with piecewise forcing function	✓	✗	✗	✗	✗	✗
Integro DE	✗	✗	✓	✗	✗	✗

Experiments




Different history
Same history
x(t)
t
2
1
0
−1
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
True Trajectory
Neural Laplace
NODE
ANODE
Latent ODE
NF Coupling
NF ResNet

Experiments

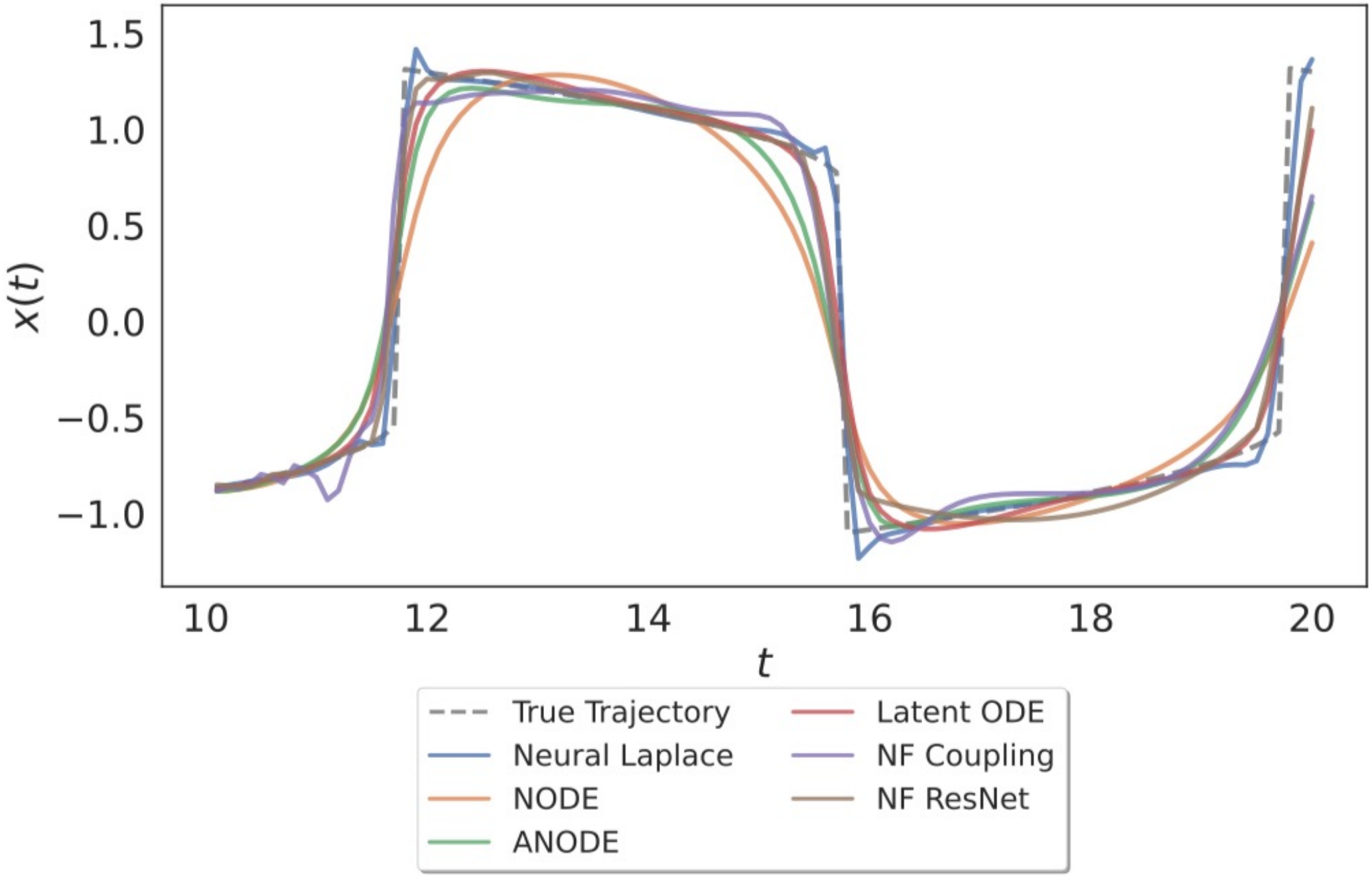
Table 4. Test RMSE for datasets analyzed. Best results bolded. Averaged over 5 runs.

Method	Spiral DDE	Lotka-Volterra DDE	Mackey-Glass DDE	Stiff Van der Pol Oscillator DE	ODE piecewise forcing function	Integro DE
NODE	.0389 ± .0029	.3102 ± .0151	.8225 ± .0403	.2833 ± .0032	.2274 ± .0298	.0730 ± .0016
ANODE	.0365 ± .0011	.2930 ± .0239	.8214 ± .0415	.2444 ± .0167	.0644 ± .0211	.0036 ± .0003
Latent ODE	.0481 ± .0033	.2182 ± .0153	**.0385 ± .0217**	.1932 ± .0154	.1401 ± .0457	.0109 ± .0009
NF Coupling	.6938 ± .1036	.7266 ± .0310	**.0539 ± .0181**	.1829 ± .0209	.0752 ± .0052	.0042 ± .0013
NF ResNet	.1905 ± .0479	.2257 ± .0608	**.0350 ± .0223**	**.1468 ± .0396**	.0399 ± .0119	.0027 ± .0004
Neural Laplace	**.0331 ± .0023**	**.0475 ± .0061**	**.0282 ± .0246**	**.1314 ± .0218**	**.0035 ± .0004**	**.0014 ± .0005**

Experiments

Table 4. Test RMSE for datasets analyzed. Best results bolded. Averaged over 5 runs.

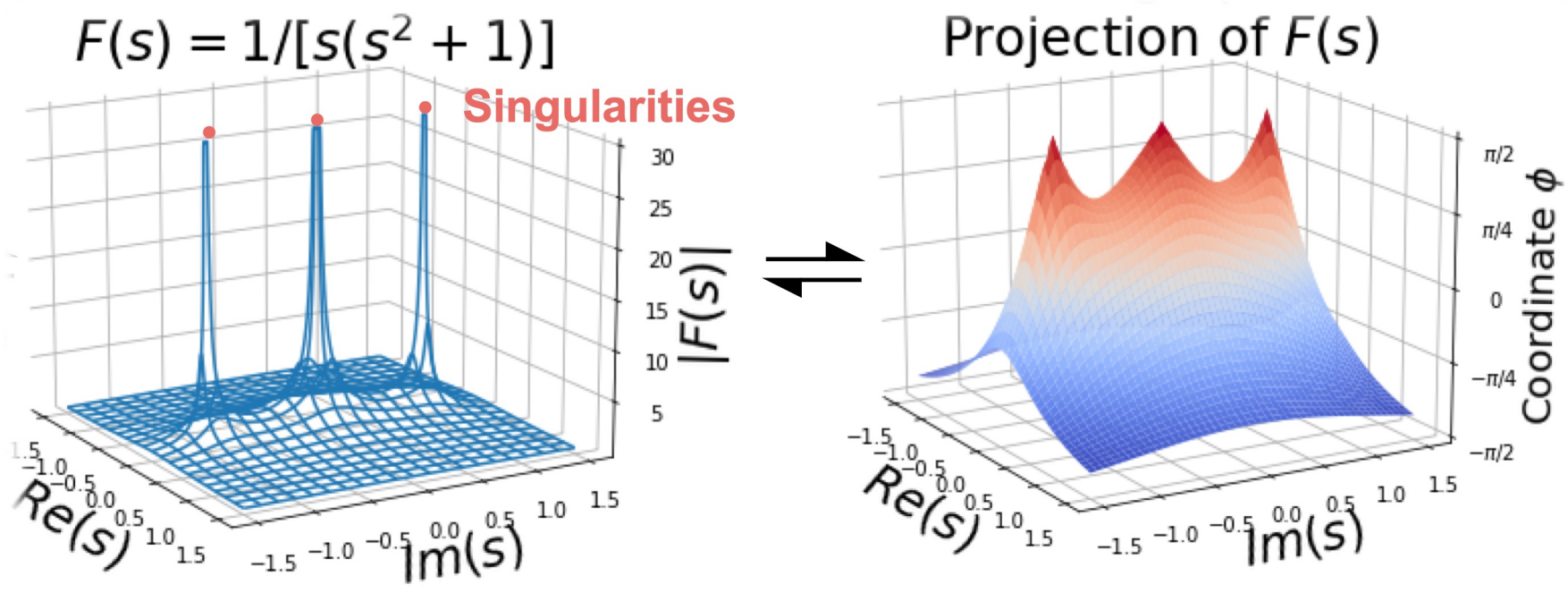
Method	Spiral DDE	Lotka-Volterra DDE	Mackey-Glass DDE	Stiff Van der Pol Oscillator DE	ODE piecewise forcing function	Integro DE
NODE	.0389 ± .0029	.3102 ± .0151	.8225 ± .0403	.2833 ± .0032	.2274 ± .0298	.0730 ± .0016
ANODE	.0365 ± .0011	.2930 ± .0239	.8214 ± .0415	.2444 ± .0167	.0644 ± .0211	.0036 ± .0003
Latent ODE	.0481 ± .0033	.2182 ± .0153	**.0385 ± .0217**	.1932 ± .0154	.1401 ± .0457	.0109 ± .0009
NF Coupling	.6938 ± .1036	.7266 ± .0310	**.0539 ± .0181**	.1829 ± .0209	.0752 ± .0052	.0042 ± .0013
NF ResNet	.1905 ± .0479	.2257 ± .0608	**.0350 ± .0223**	**.1468 ± .0396**	.0399 ± .0119	.0027 ± .0004
Neural Laplace	**.0331 ± .0023**	**.0475 ± .0061**	**.0282 ± .0246**	**.1314 ± .0218**	**.0035 ± .0004**	**.0014 ± .0005**


1.5
1.0
0.5
0.0
−0.5
−1.0
x(t)
10
12
14
16
18
20
t
True Trajectory
Neural Laplace
NODE
ANODE
Latent ODE
NF Coupling
NF ResNet

Experiments - Ablation

Table 5. Neural Laplace ablation study and sensitivity analysis. RMSE is reported under different study configurations on different dynamical systems. Averaged over 5 runs.

Study	Config.	Lotka-Volterra DDE	Stiff Van der Pol Oscillator DE	ODE piecewise forcing function	Integro DE
Stereographic projection	✗	.1617 ± .0741	.1836 ± .0586	.0249 ± .0066	.0048 ± .0007
	✓	**.0614 ± .0469**	**.1286 ± .0170**	**.0036 ± .0007**	**.0013 ± .0003**



Experiments - Ablation

Table 5. Neural Laplace ablation study and sensitivity analysis. RMSE is reported under different study configurations on different dynamical systems. Averaged over 5 runs.

Study	Config.	Lotka-Volterra DDE	Stiff Van der Pol Oscillator DE	ODE piecewise forcing function	Integro DE
Stereographic projection	✗	.1617 ± .0741	.1836 ± .0586	.0249 ± .0066	.0048 ± .0007
	✓	**.0614 ± .0469**	**.1286 ± .0170**	**.0036 ± .0007**	**.0013 ± .0003**
Dimensionality K	1	.4416 ± .0898	.1520 ± .0240	.0036 ± .0007	.0010 ± .0002
	2	.0405 ± .0113	.1308 ± .0159	.0033 ± .0009	.0012 ± .0002
	4	.0427 ± .0049	.1334 ± .0103	.0038 ± .0013	.0012 ± .0003
	8	.0408 ± .0134	.1294 ± .0173	.0038 ± .0004	.0010 ± .0002
	16	.0380 ± .0053	.1334 ± .0197	.0036 ± .0006	.0012 ± .0005
	32	.0398 ± .0045	.1337 ± .0203	.0032 ± .0005	.0013 ± .0003

$$\textbf{initial condition } \mathbf{p} \in \mathbb{R}^K$$

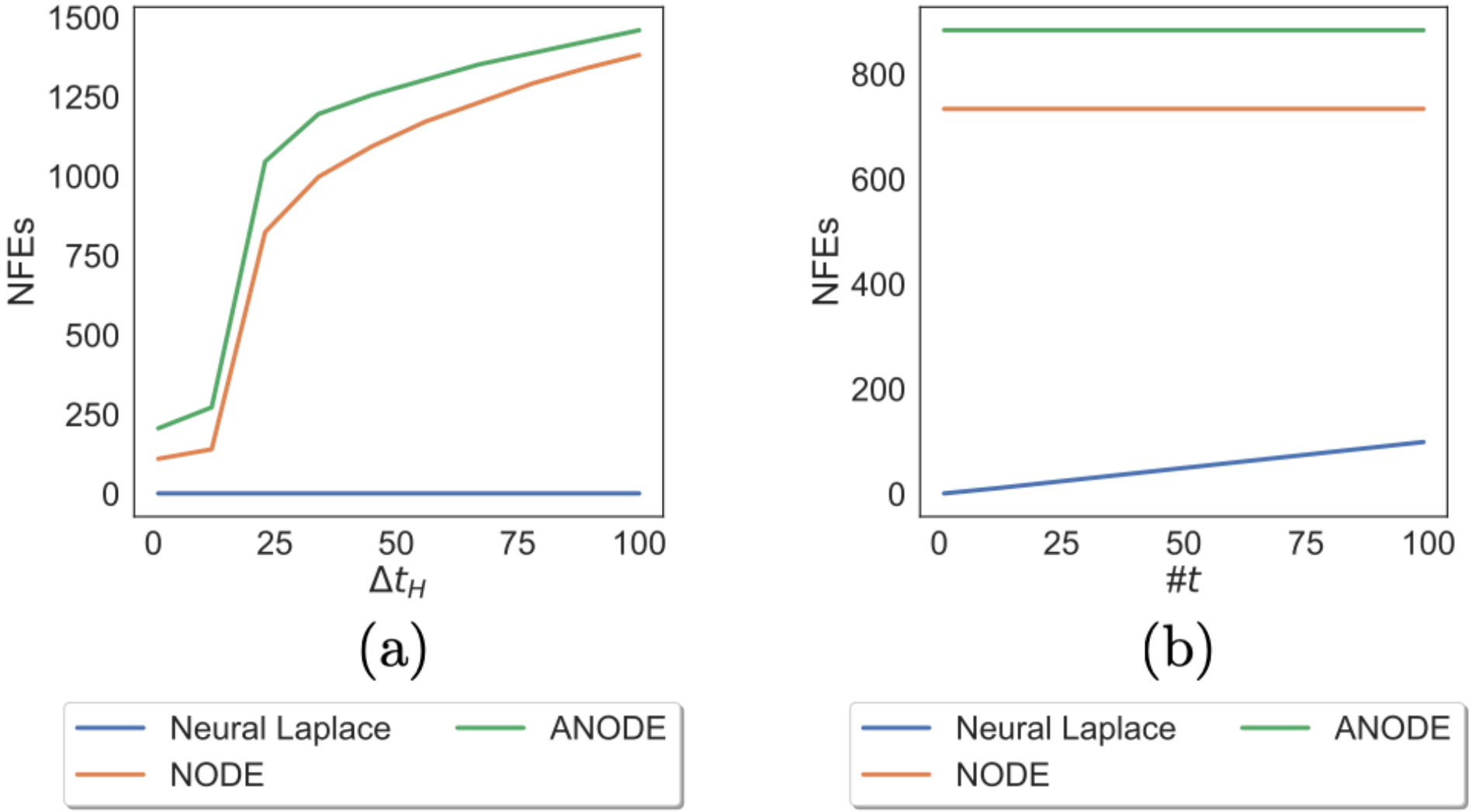
Experiments - Ablation

Table 5. Neural Laplace ablation study and sensitivity analysis. RMSE is reported under different study configurations on different dynamical systems. Averaged over 5 runs.

Study	Config.	Lotka-Volterra DDE	Stiff Van der Pol Oscillator DE	ODE piecewise forcing function	Integro DE
Stereographic projection	✗	.1617 ± .0741	.1836 ± .0586	.0249 ± .0066	.0048 ± .0007
	✓	**.0614 ± .0469**	**.1286 ± .0170**	**.0036 ± .0007**	**.0013 ± .0003**
Dimensionality K	1	.4416 ± .0898	.1520 ± .0240	.0036 ± .0007	.0010 ± .0002
	2	.0405 ± .0113	.1308 ± .0159	.0033 ± .0009	.0012 ± .0002
	4	.0427 ± .0049	.1334 ± .0103	.0038 ± .0013	.0012 ± .0003
	8	.0408 ± .0134	.1294 ± .0173	.0038 ± .0004	.0010 ± .0002
	16	.0380 ± .0053	.1334 ± .0197	.0036 ± .0006	.0012 ± .0005
	32	.0398 ± .0045	.1337 ± .0203	.0032 ± .0005	.0013 ± .0003

$$\textbf{initial condition } \mathbf{p} \in \mathbb{R}^K$$

Experiments - Complexity

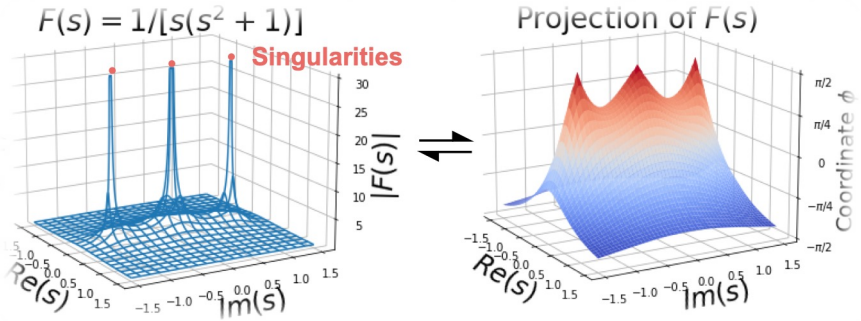


Experiments – Computation time

Table 14. Average wall clock time taken to train on one epoch, 1,000 trajectories on the Lotka-Volterra DDE dataset.

Method	seconds per epoch
NODE (dopri5)	13.78
NODE (euler)	1.92
ANODE (dopri5)	17.23
ANODE (euler)	4.56
Latent ODE	3.56
NF Coupling	10.83
NF ResNet	0.1
Neural Laplace	0.1

Contributions



- Neural Laplace can learn diverse classes of DEs, by modelling them in the Laplace domain.
- It can reconstruct any trajectory point $\mathbf{x}(t)$ at any time $t \in \mathbb{R}$.
- It scales linearly with the number of time points to evaluate, whilst modelling diverse DE systems better than existing methods.

- We have released a PyTorch implementation of Neural Laplace, including GPU implementations of several ILT algorithms. The code for this is at https://github.com/samholt/NeuralLaplace and https://github.com/vanderschaarlab/neurallaplace.

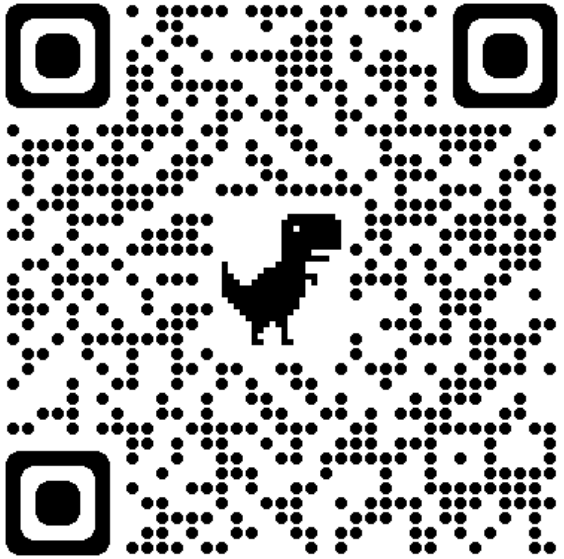
Future work

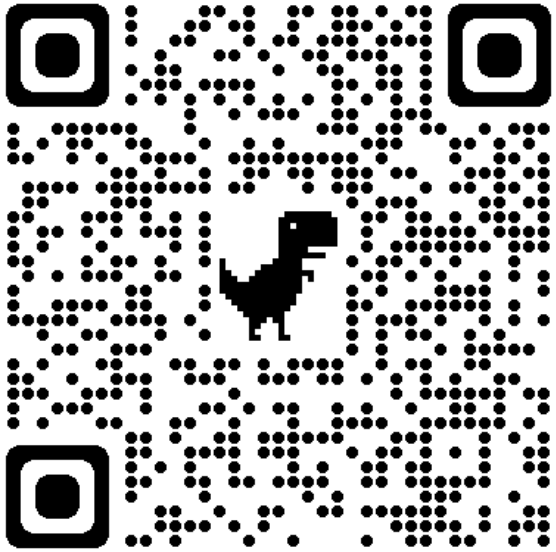
- We wish to use the learned Laplace representation to investigate the other unique properties of this representation, such as:
  - Stability analysis
  - Limiting its frequency reconstruction terms
  - using the Laplace final limit theorem as an additional regularizer

More info:

paper

code





AstraZeneca

Office of Naval Research
ONR
Science & Technology
USN

arxiv.org/abs/2206.04843

github.com/samholt/NeuralLaplace
github.com/vanderschaarlab/neurallaplace

van_der_Schaar
\ LAB
vanderschaar-lab.com

UNIVERSITY OF CAMBRIDGE

sih31@cam.ac.uk
github.com/samholt
@samianholt
linkedin.com/in/samuel-holt