



AFRL

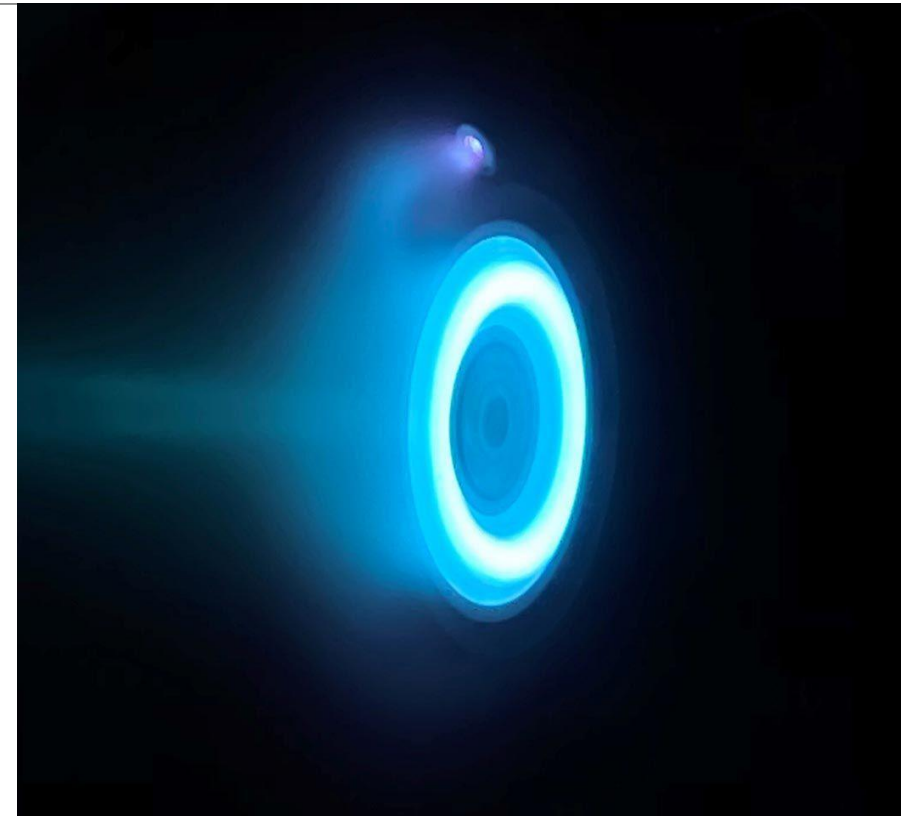
Model-free Parameter Estimation

Presenter: Adrian S. Wong (AFRL)

Work with: Alex T. Lin (UCLA), Dan Eckhardt (AFRL), Rob Martin (ARL)

Motivation

- **Electrical Propulsion Thrusters**
 - Engineered, complex dynamical systems
 - Infeasible models
 - Various operating modes
- **Wear-and-tear on System**
 - “Dial says current is 2A, but thruster is operating as if it were 1.5A.”
- **Facility Effects**
 - “Mode 1 happens at different parameters values in different facilities.”
- **GOAL:**
 - Find the effective parameters given time series measurements
 - “Push” system back to desired mode



Basic Parameter Estimation

- Dynamical model : $\vec{x}_{n+1}^m = F(\vec{x}_n^m; \vec{p}^m)$
- Time index: n , parameter index: m
- Given model and many pairs of $[\vec{x}_{n+1}^m, \vec{x}_n^m]$ data across n

$$\vec{p}^m = \underset{\vec{p}}{\operatorname{argmin}} \left\{ \mathbb{E}_n \left\| \vec{x}_{n+1}^m - F(\vec{x}_n^m; \vec{p}) \right\|^2 \right\}$$

“Give me a model and **trajectory**, I’ll give you the **parameters**.”

Many pairs of $[\vec{x}_{n+1}, \vec{x}_n]$

Using Optimization

$$\min_{\vec{p}} \left\{ \mathbb{E}_n \left\| \vec{x}_{n+1}^m - F(\vec{x}_n^m; \vec{p}) \right\|^2 \right\}$$

- Estimation becomes an optimization problem
- Minimization: Pick your poison (GD, Newton, NAG, AdaM)

Caveat 1: Need **state** variables.

Caveat 2: Need a **good** and **parameterized** model.

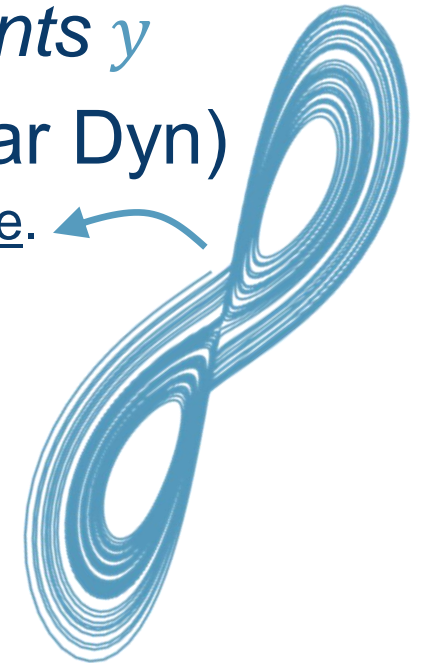
Minimum unique description

Working Around Caveat 1

Caveat 1: Need **state** variables.

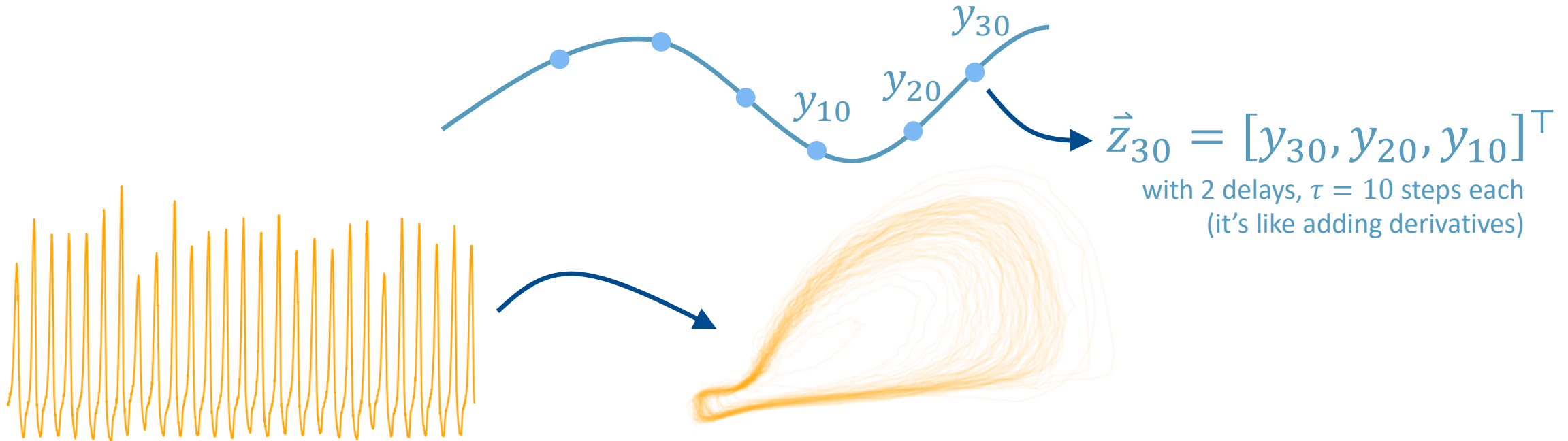
- State variables are often unknown or unmeasurable directly
- Have a univariate/scalar time series, aka *measurements* y
- Time-delay Embedding Theorems (Takens' + Nonlinear Dyn)
 - With sufficient number of delays, data can preserve topological structure.
 - Number of delays is bounded *reasonably*

“Measurements with enough delays are equivalent to state variables.”



Time Delay Embedding

- Given a univariate trajectory $y = \{y_1, y_2, \dots, y_N\}$
- Embed y in a D -variate trajectory $Z = \{\vec{z}_1, \vec{z}_2, \dots, \vec{z}_{N-(D-1)*\tau}\}$
- Form $\vec{z}_n = [y_n, y_{n-\tau}, \dots, y_{n-(D-1)*\tau}]^T$, where τ is the delay



Working Around Caveat 2

Caveat 2: Need a **good** and **parameterized** model.

- Need the function F s.t. $\vec{x}_{n+1}^m = F(\vec{x}_n^m; \vec{p}^m)$, but...
- Good models are often unknown or hard to come by

“Make our own surrogate model (that works on time delay data).”

*Shallow Neural Network approach: Multi-layer Perceptron (MLP)
[3 hidden layer, 2000 nodes each]

Neural Network as Surrogate

State variables: \vec{x} Measurements: y Time-delay: $\vec{z}(y)$

- Originally want: F s.t. $\vec{x}_{n+1}^m = F(\vec{x}_n^m; \vec{p}^m)$, but instead...
- *Make* our own surrogate \tilde{F} s.t. $\vec{z}_{n+1}^m = \tilde{F}(\vec{z}_n^m, \vec{p}^m; w)$
 - Prescribe structure \tilde{F} , solve for the remaining weights w

$$w = \underset{w'}{\operatorname{argmin}} \left\{ \mathbb{E}_{n,m} \left\| \vec{z}_{n+1}^m - \tilde{F}(\vec{z}_n^m, \vec{p}^m; w') \right\|^2 \right\}$$

secret sauce:

learning the entire
parameter **family**

Weights complete the surrogate model.

Model-free Parameter Estimation

“Give me a ~~model~~ and **trajectory**, I’ll give you the **parameters**.”

$$\vec{p}^m = \operatorname{argmin}_{\vec{p}} \left\{ \mathbb{E}_n \left\| \vec{x}_{n+1}^m - F(\vec{x}_n^m; \vec{p}) \right\|^2 \right\}$$

Caveat 2: model unknown

Caveat 1: state space unknown

$$\vec{p}^m = \operatorname{argmin}_{\vec{p}} \left\{ \mathbb{E}_{n,m} \left\| \vec{z}_{n+1}^m - \tilde{F}(\vec{z}_n^m, \vec{p}; w) \right\|^2 \right\}$$

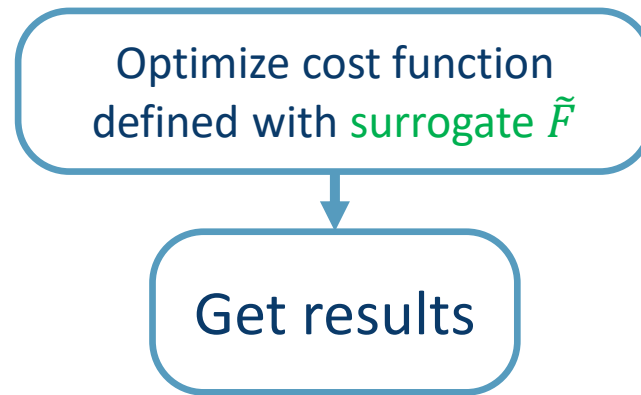
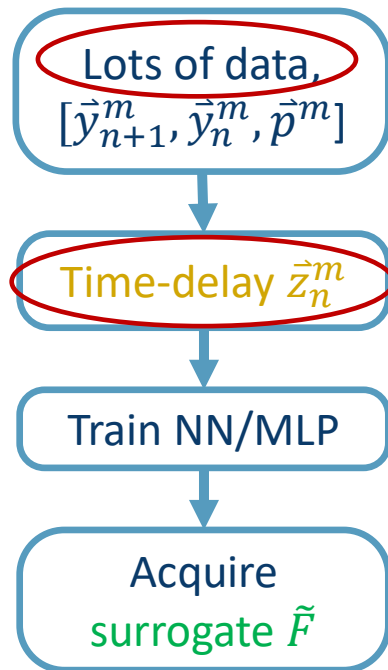
measurements

Solved for (previously)

fixed/prescribed

Technical Summary

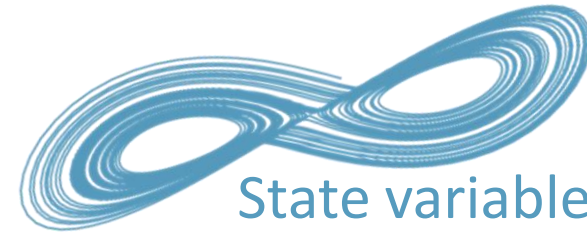
Training Phase: Estimating Parameters:



1. No model? **No problem.**
2. Don't know what to measure? **That's fine too** (within reason).

Parameter estimation without models or state variables.

Results on Lorenz System

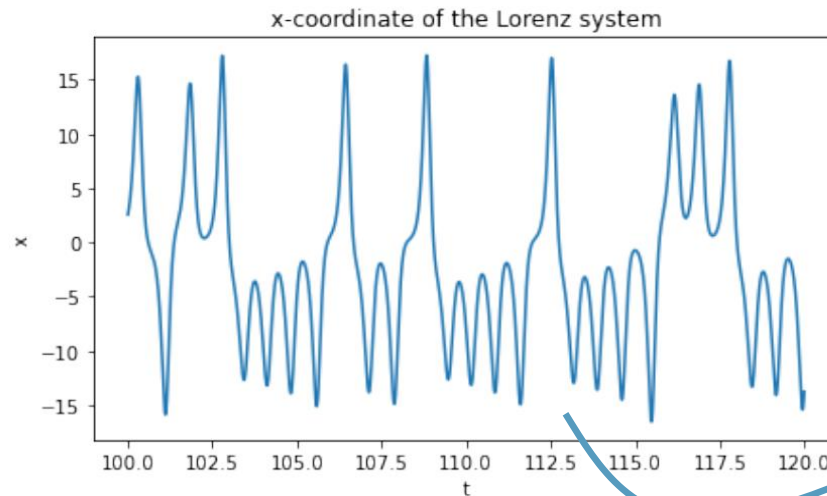
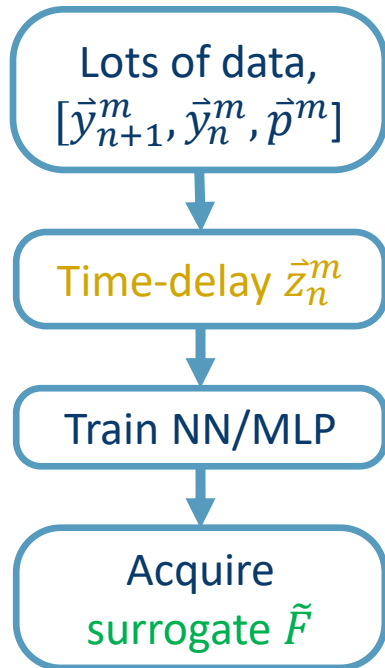


State variables

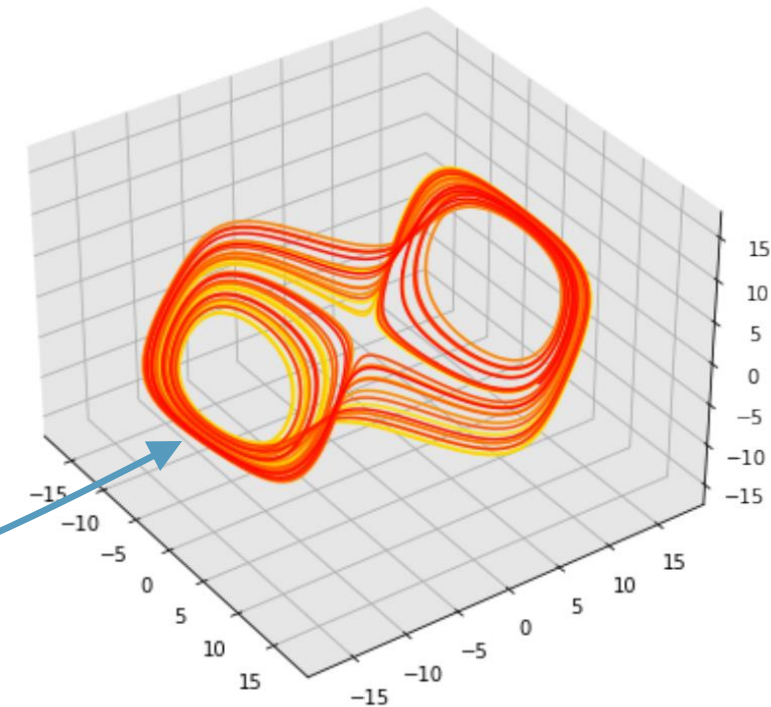
$$\dot{x} = \sigma(y - x)$$

$$\dot{y} = x(\rho - z) - y$$

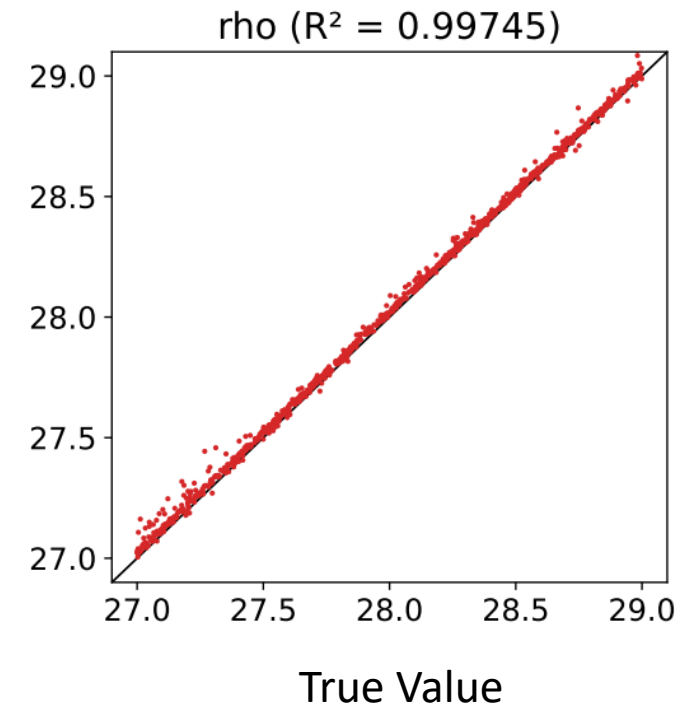
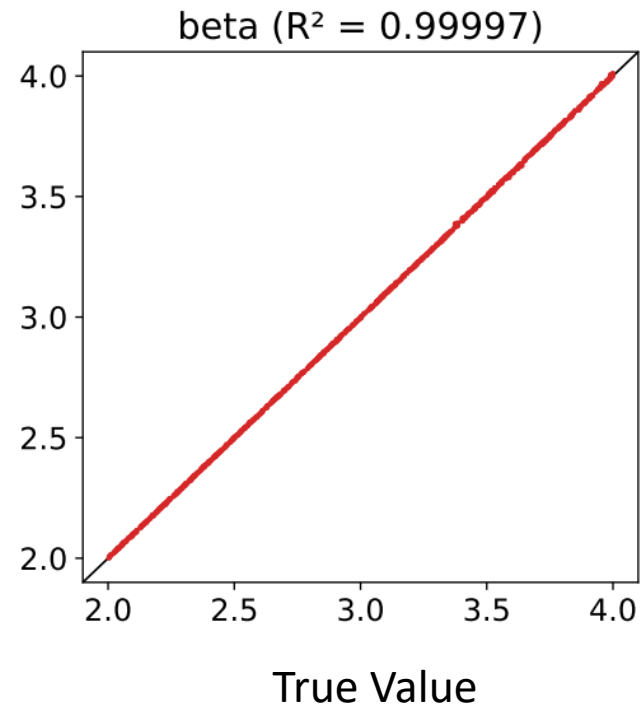
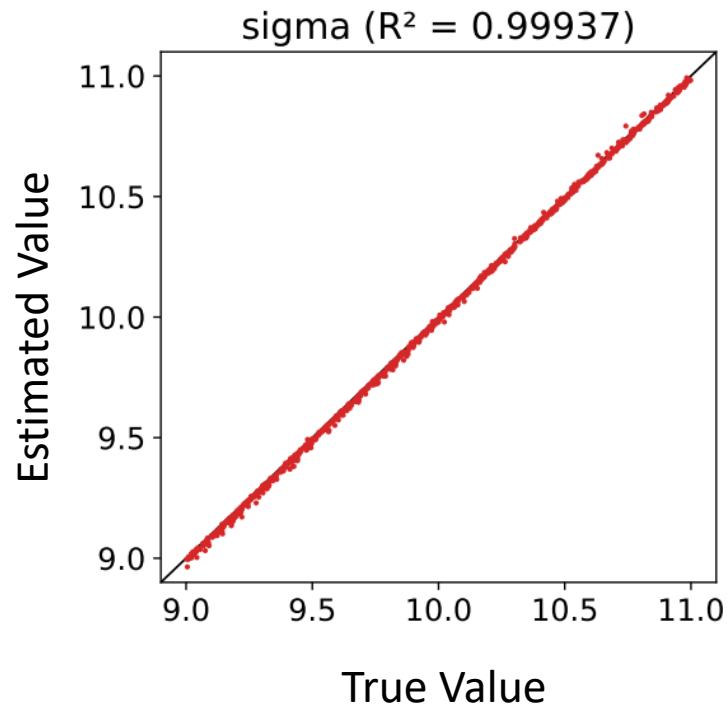
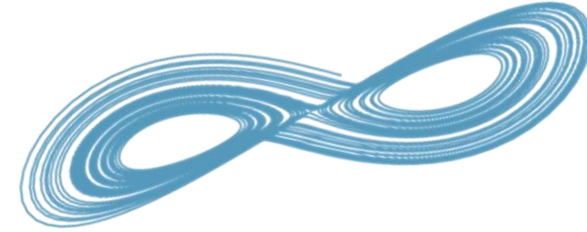
$$\dot{z} = xy - \beta z$$



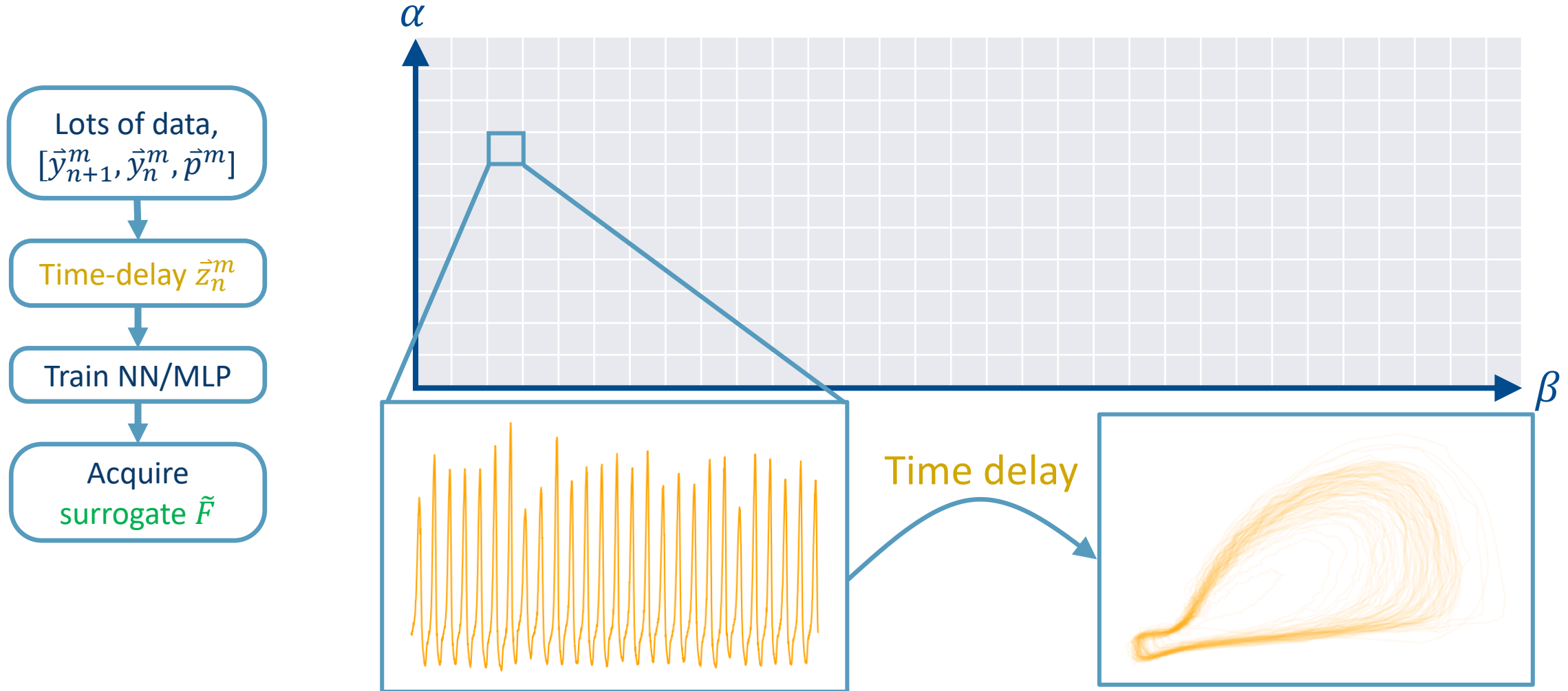
Time delay



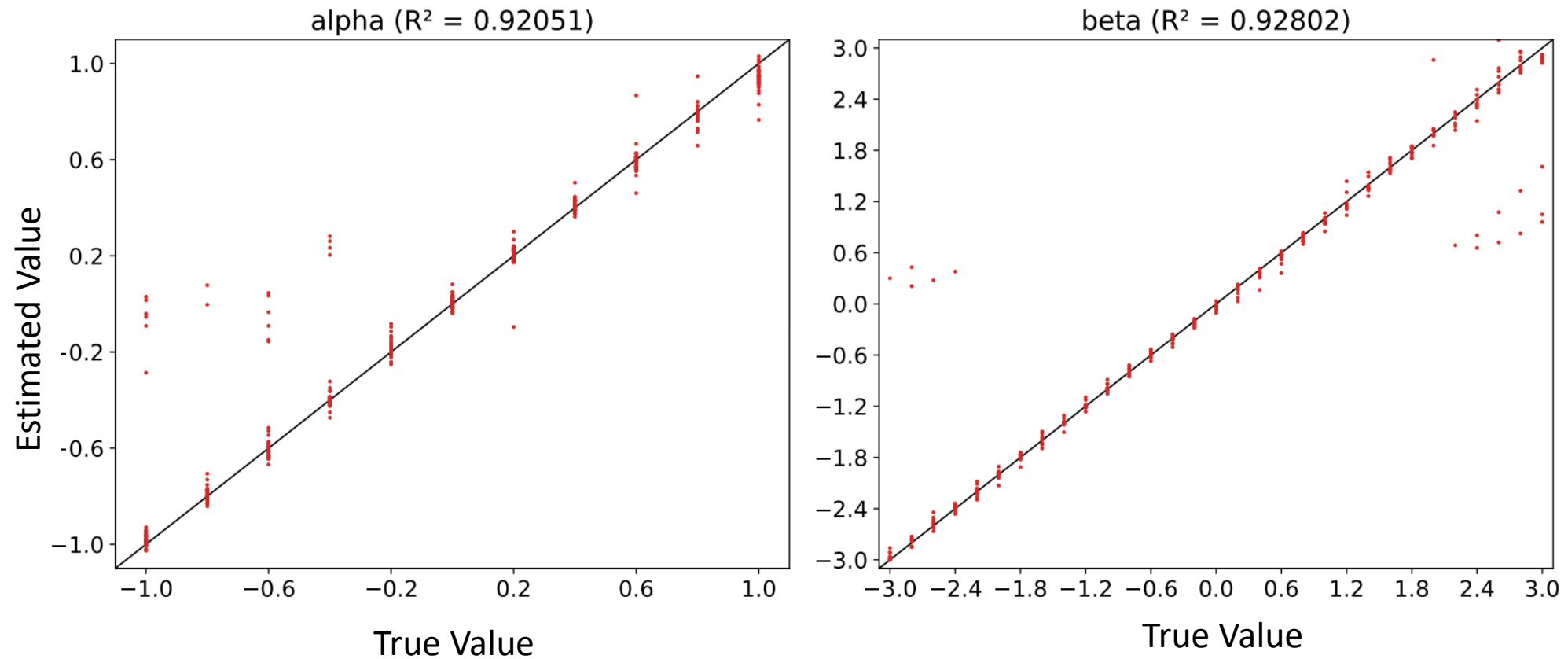
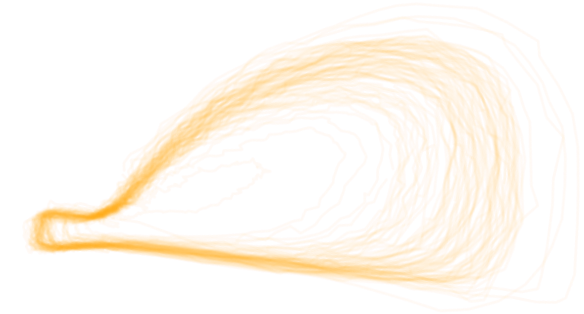
Results on Lorenz System



Results on Hall Thruster System



Results on Hall Thruster System



Reservoir Computing

- Yet another model-free method
 - No need for state variables
 - No need for physical model
- Prediction and Inference
 - Prediction – Data in $0 < t < T$, predict states $t > T$
 - Inference – Present partial measurements, infer missing ones
- Less “black box” than others
 - Synchronization-based (Pecora and Carroll 1990)

Reservoir Computing in Equations

no access to this knowledge

$$x_{m+1} = F(x_m; p)$$

State variables x , parameters p

$$y_m = h(x_m) + \eta$$

Measurements y , noise η ($0 < t < T$)

$$r_{m+1} = G(r_m, y_m; q)$$

Reservoir state r , hyperparameters q

(Run out of signal, $t > T$)

“Used all the data to get $r(t)$ ”

$$y_m \simeq \hat{y}_m = W_{out} r_m ; \forall m$$

Solve for best projection W_{out}

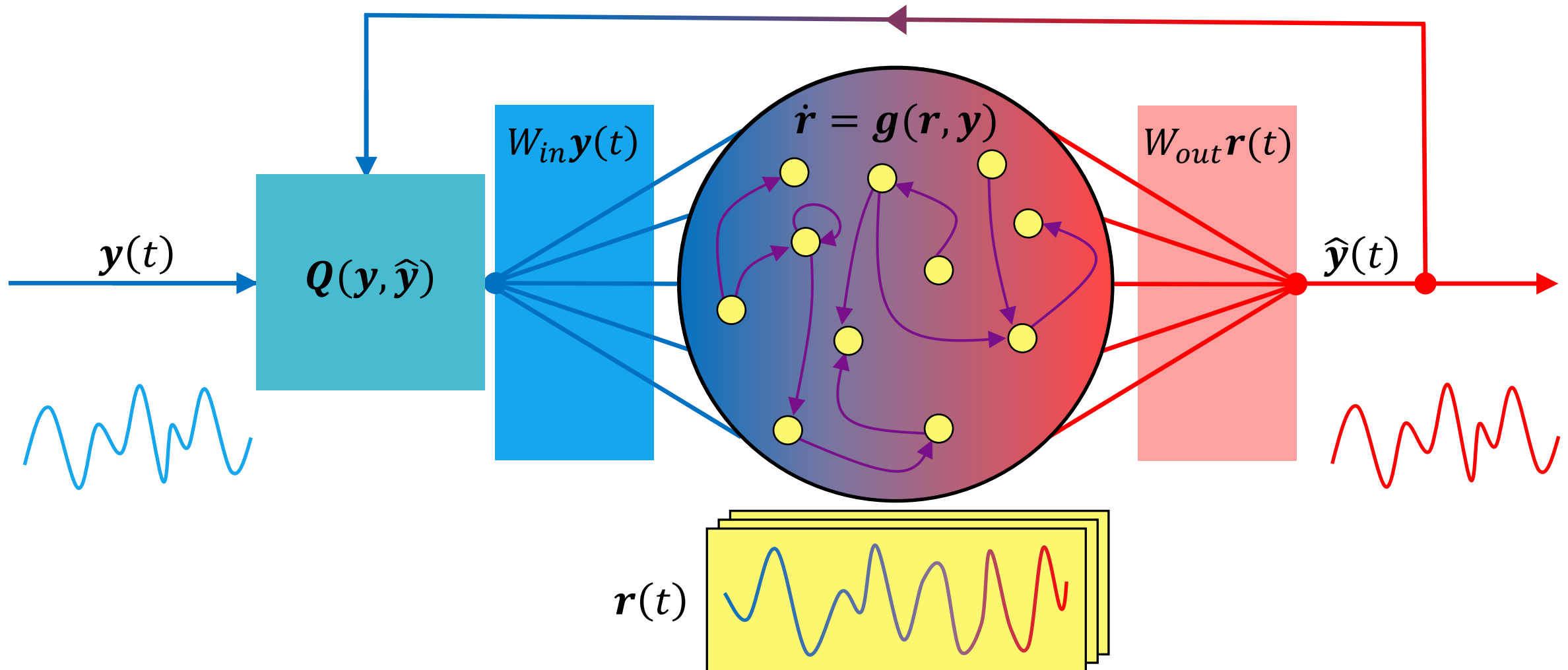
$$r_{m+1} \simeq G(r_m, \hat{y}_m; q)$$

Substitute in the estimated state \hat{y}

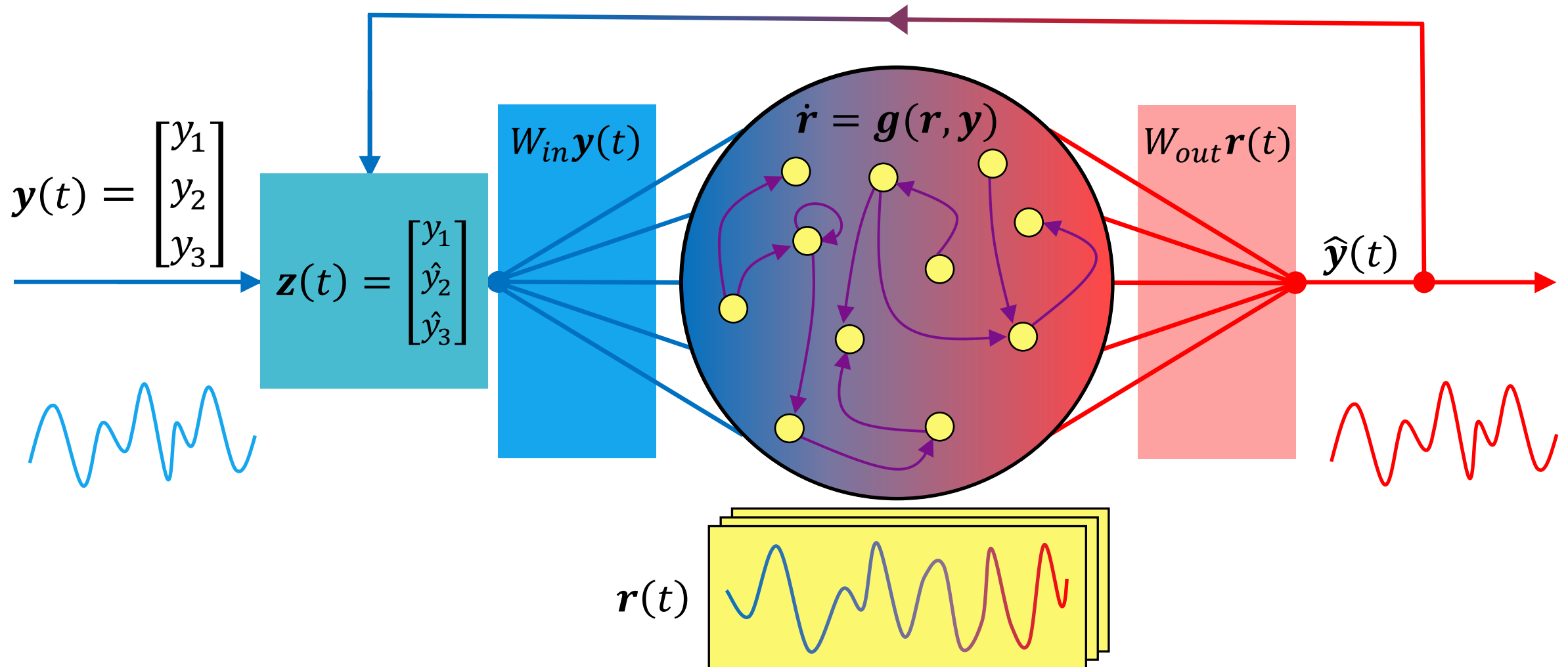
$$= \tilde{G}(r_m; W_{out}, q)$$

Predict forward without needing y

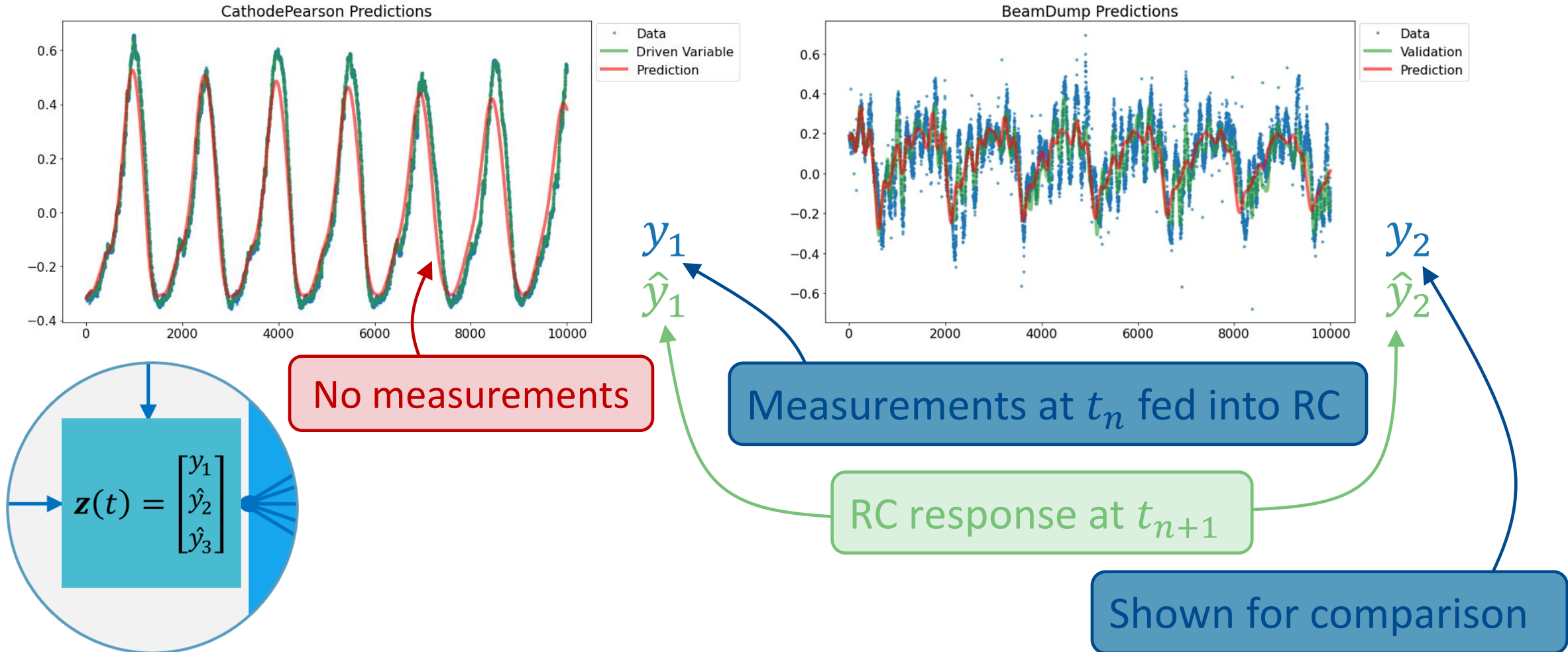
Reservoir Computing Visual



Reservoir Computing Visual



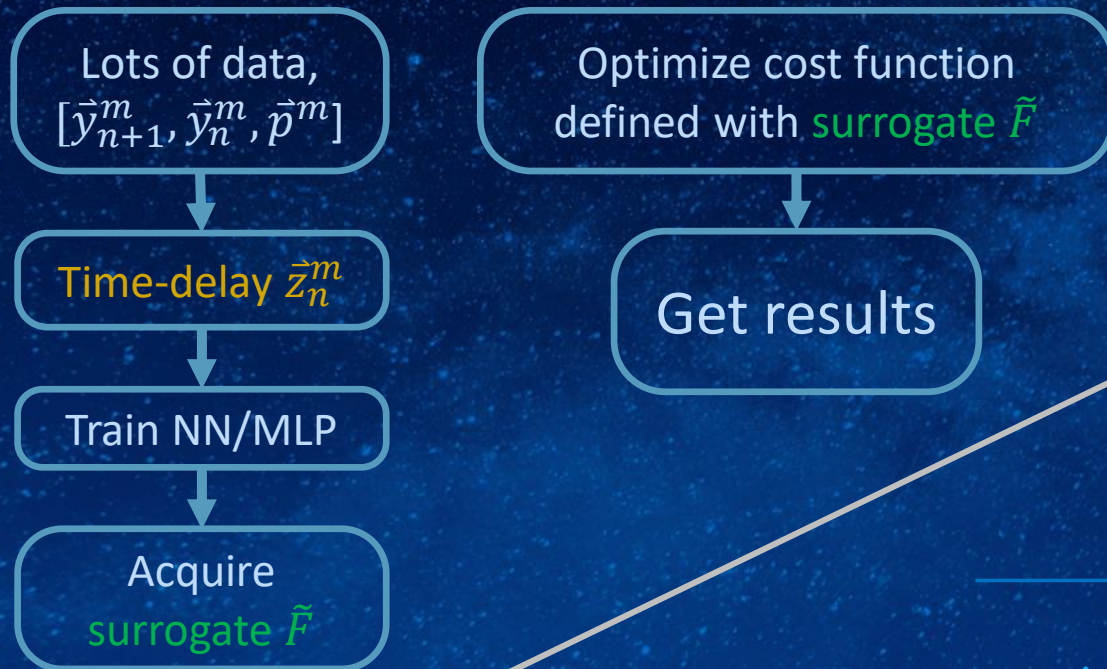
Tracking of Unmeasured States



Conclusions

- Data-Driven/Model-Free methods work well with Time-Delays
 - *Parameter Estimation* with Shallow Feed-Forward Network
 - *Inference* with Reservoir Computing
- ***Parameter Estimation:***
 - Experimental data across parameters → train surrogate models
 - Coupled Nonlinear Dynamics → Time Delay vs. State Space
 - Smoothness in parameters space (derivatives)
- ***Inference:***
 - RC Synchronization → Topological Equivalence (Diffeomorphic)

Overview



$$r_{m+1} = G(r_m, y_m; q)$$

$$y_m \simeq \hat{y}_m = W_{out} r_m ; \forall m$$

$$r_{m+1} \simeq G(r_m, \hat{y}_m; q)$$

$$= \tilde{G}(r_m; W_{out}, q)$$

