is paper describes objective technical results and analysis. Any subjective views or opinions that might be expres
the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Governm

SAND2019-9139C

Time-series Machine Learning Error Models for Appproximate Solutions to Dynamical Systems

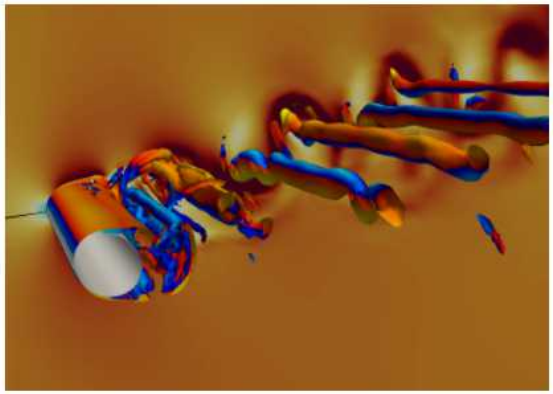
E. Parish and K. Carlberg
Sandia National Labs
USNCCM 15

July 31, 2019

oratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia,
neywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract

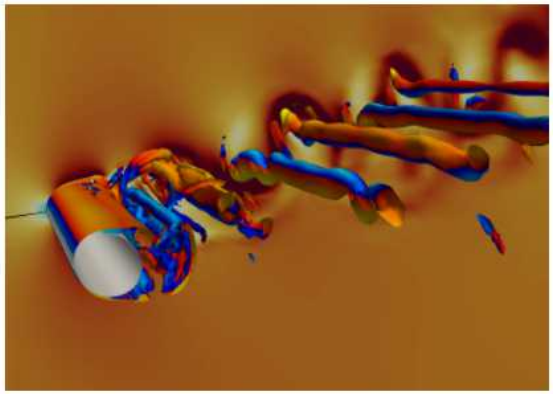
Motivation

- Parameterized dynamical systems are ubiquitous in science and engineering



Motivation

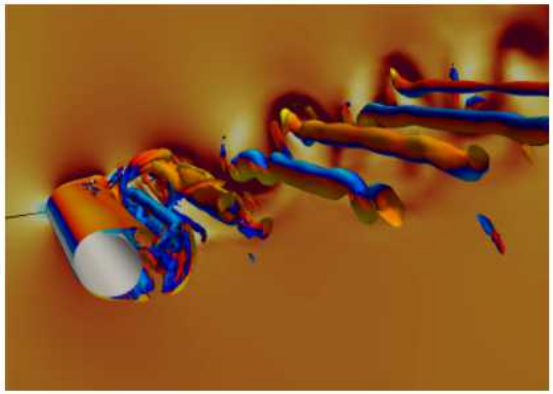
- Parameterized dynamical systems are ubiquitous in science and engineering



  - Approximate models are often employed to reduce cost

Motivation

- Parameterized dynamical systems are ubiquitous in science and engineering



  - Approximate models are often employed to reduce cost
  - **Quantifying errors in approximate solutions is the objective of this work**

Full-Order Model

- Work focuses on the FOM ODE

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{\mu})$$

  - Parameters, $\boldsymbol{\mu} \in \mathcal{D} \subseteq \mathbb{R}^{N_\mu}$

Full-Order Model

- Work focuses on the FOM ODE

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{\mu})$$

  - Parameters, $\boldsymbol{\mu} \in \mathcal{D} \subseteq \mathbb{R}^{N_\mu}$

- Time discretization: FOM ODE $\mapsto$ FOM OΔE

$$\boldsymbol{r}^n(\boldsymbol{x}^n; \boldsymbol{x}^{n-1}, \boldsymbol{\mu}) = \boldsymbol{0}, \qquad n = 1, \ldots, N_t$$

  - Discrete residual, $\boldsymbol{r}^n$
  - Discrete approximation, $\boldsymbol{x}^n \approx \boldsymbol{x}(t^n)$
  - Number of time-step instances, N_t

Full-Order Model

- Work focuses on the FOM ODE

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{\mu})$$

  - Parameters, $\boldsymbol{\mu} \in \mathcal{D} \subseteq \mathbb{R}^{N_\mu}$

- Time discretization: FOM ODE $\mapsto$ FOM OΔE

$$\boldsymbol{r}^n(\boldsymbol{x}^n; \boldsymbol{x}^{n-1}, \boldsymbol{\mu}) = \boldsymbol{0}, \qquad n = 1, \ldots, N_t$$

  - Discrete residual, $\boldsymbol{r}^n$
  - Discrete approximation, $\boldsymbol{x}^n \approx \boldsymbol{x}(t^n)$
  - Number of time-step instances, N_t

- Often interested in a scalar quantity of interest (QoI):

$$s^n : \boldsymbol{\mu} \mapsto g(\boldsymbol{x}^n(\boldsymbol{\mu}); t^n, \boldsymbol{\mu}), \quad n = 0, \ldots, N_t,$$

Approximate Models

- Generate a sequence of **approximate solutions**

$$\hat{\boldsymbol{x}}^n, n = 0, \ldots, N_t$$

- We assume the existence of a prolongation operator $\mathbf{p}$ that maps to the FOM state space

$$\tilde{\boldsymbol{x}}^n = \mathbf{p}(\hat{\boldsymbol{x}}^n) \approx \boldsymbol{x}^n$$

- Approximate QoIs

$$\tilde{s}^n, n = 0, \ldots, N_t$$

- **Challenge: The approximate model contains error**

$$\delta_{\boldsymbol{x}}^n := \|\boldsymbol{x}^n - \tilde{\boldsymbol{x}}^n\|_2 \neq 0$$
$$\delta_s^n := s^n - \tilde{s}^n \neq 0$$

Traditional Approaches for Error Quantification

- *A posteriori* error bounds

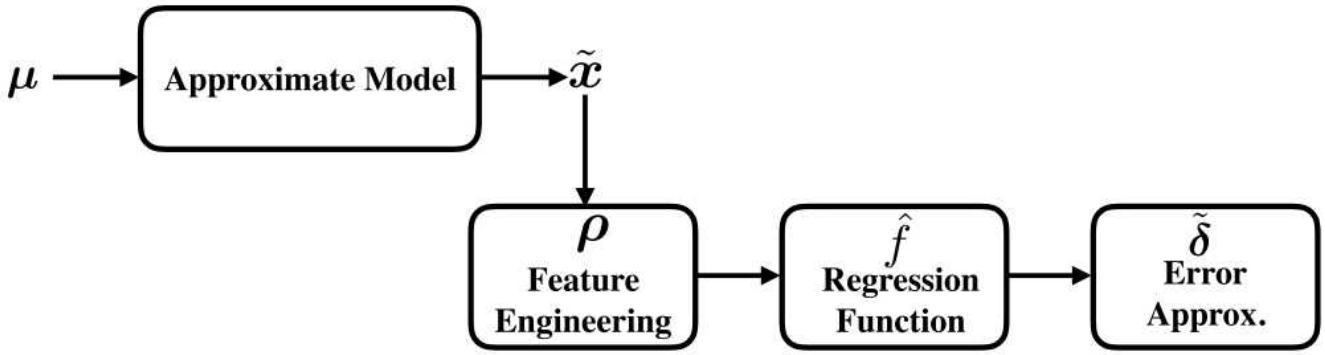
$$\delta_{\boldsymbol{x}}^{n}(\boldsymbol{\mu}) \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}(\boldsymbol{\mu}); \tilde{\boldsymbol{x}}^{n-1}(\boldsymbol{\mu}), \boldsymbol{\mu} \right) \right\|_{2} + \gamma \delta_{\boldsymbol{x}}^{n-1}(\boldsymbol{\mu})$$

  - **Provide guaranteed bounds**
  - **Lack of sharpness/inaccuracy**
  - **Difficulty to compute**
- Dual weighted residuals
  - **First order approximation for QoI errors**
  - **Challenging to implement**
  - **Can be expensive**

- **Motivates the use of a posteriori error models**

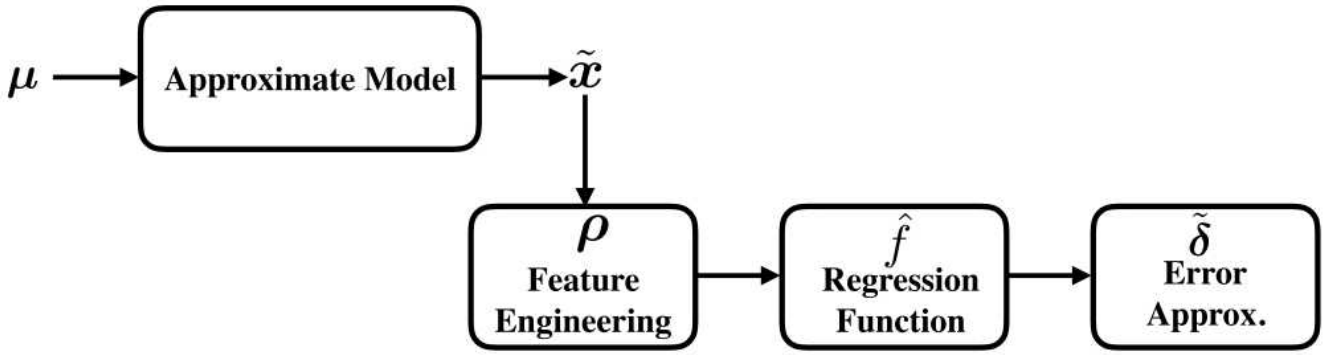
A posteriori Error Models for Static Systems

- Build a **feature-error** mapping via a regression function



A posteriori Error Models for Static Systems

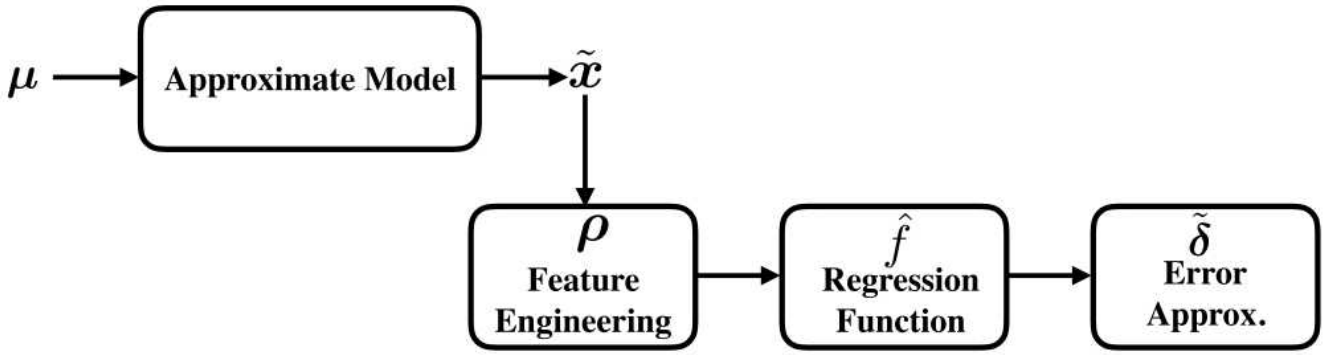
- Build a **feature-error** mapping via a regression function



$$\hat{\delta} = \hat{f}(\boldsymbol{\rho})$$

A posteriori Error Models for Static Systems

- Build a **feature-error** mapping via a regression function

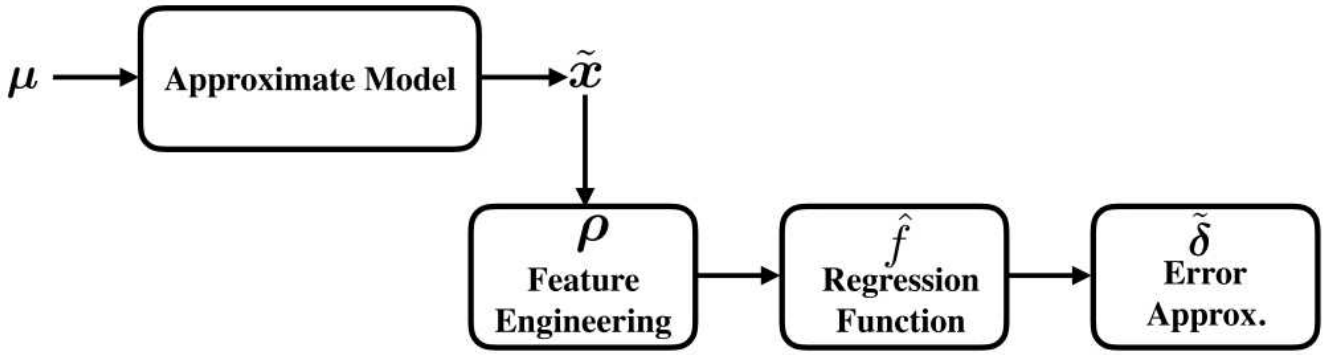


$$\hat{\delta} = \hat{f}(\boldsymbol{\rho})$$

- Existing work on error models:
  1. Kennedy and O'Hagan (2001): Model discrepancy
     - **Features:** $\rho = \mu$
     - **Uses a GP regression function**

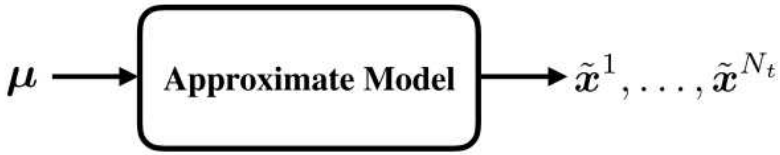
A posteriori Error Models for Static Systems

- Build a **feature-error** mapping via a regression function

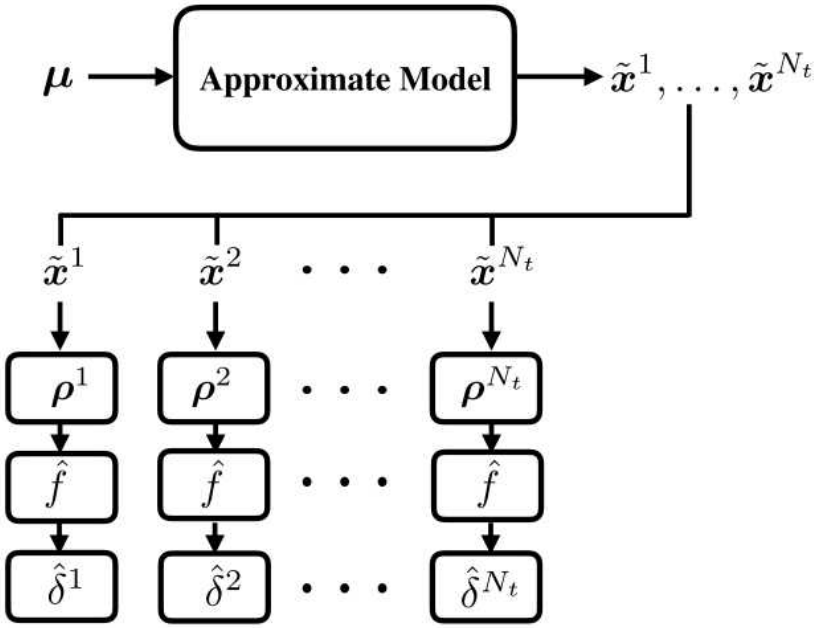


- Existing work on error models:
  1. Kennedy and O'Hagan (2001): Model discrepancy
  2. Drohmann and Carlberg (2015): Reduced-order modeling error surrogates approach (ROMES)
  3. Freno and Carlberg (2019): Machine-learning error models (MLEM)

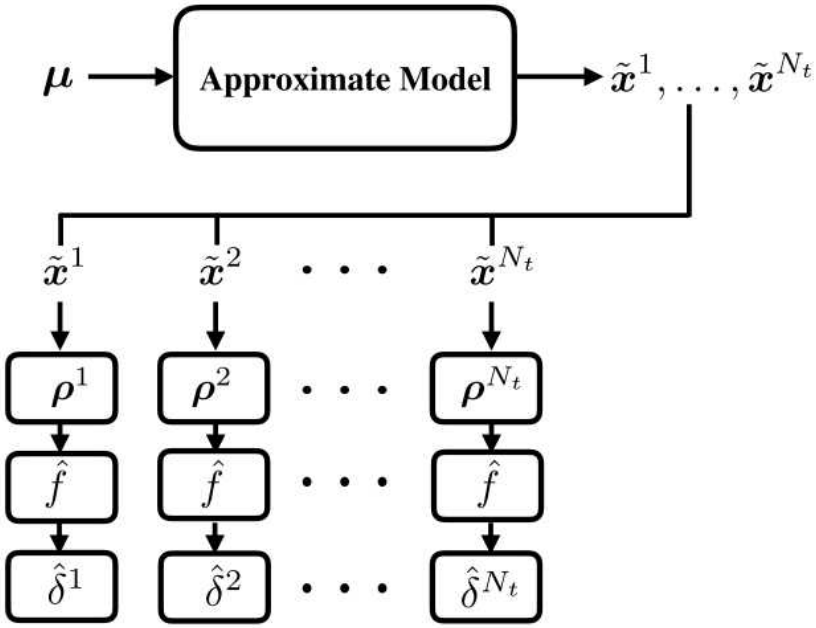
A posteriori Error Models for Dynamic Systems



A posteriori Error Models for Dynamic Systems

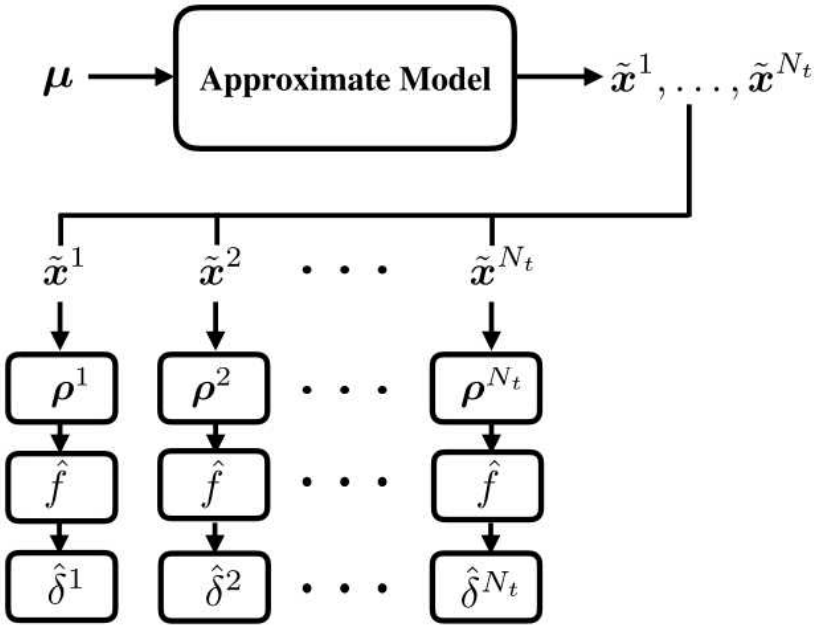


A posteriori Error Models for Dynamic Systems



$$\hat{\delta}^n = \hat{f}(\boldsymbol{\rho}^n)$$

A posteriori Error Models for Dynamic Systems



$$\hat{\delta}^n = \hat{f}(\boldsymbol{\rho}^n)$$

- **Physically inconsistent**
- Error depends on the **history** of the system

$$\delta_{\boldsymbol{x}}^n(\boldsymbol{\mu}) \leq \frac{1}{h} \left\| \boldsymbol{r}^n \left(\tilde{\boldsymbol{x}}^n(\boldsymbol{\mu}); \tilde{\boldsymbol{x}}^{n-1}(\boldsymbol{\mu}), \boldsymbol{\mu} \right) \right\|_2 + \gamma \delta_{\boldsymbol{x}}^{n-1}(\boldsymbol{\mu})$$

Existing work on Error Modeling for Dynamical Systems

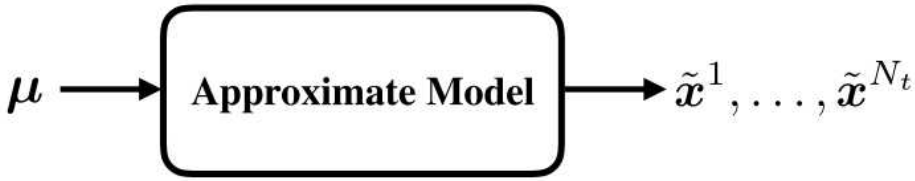
- Model discrepancy approach with time-local Error Models
  - e.g., Pagani, Manzoni, and Quarteroni (2017)
  - Constructs a **separate** regression function for each point in time
  - Can lead to accurate error models
  - Can not generalize to new time intervals
  - Uses only the parameters as input features

- Error Modeling via Machine Learning (EMML)
  - Trehan, Carlberg, and Durlofsky (2017)
  - Constructs regression functions with time-lagged input features
  - Considers past states as inputs
  - Considers high dimension regression functions
  - Prediction at each time-step is independent of previous prediction

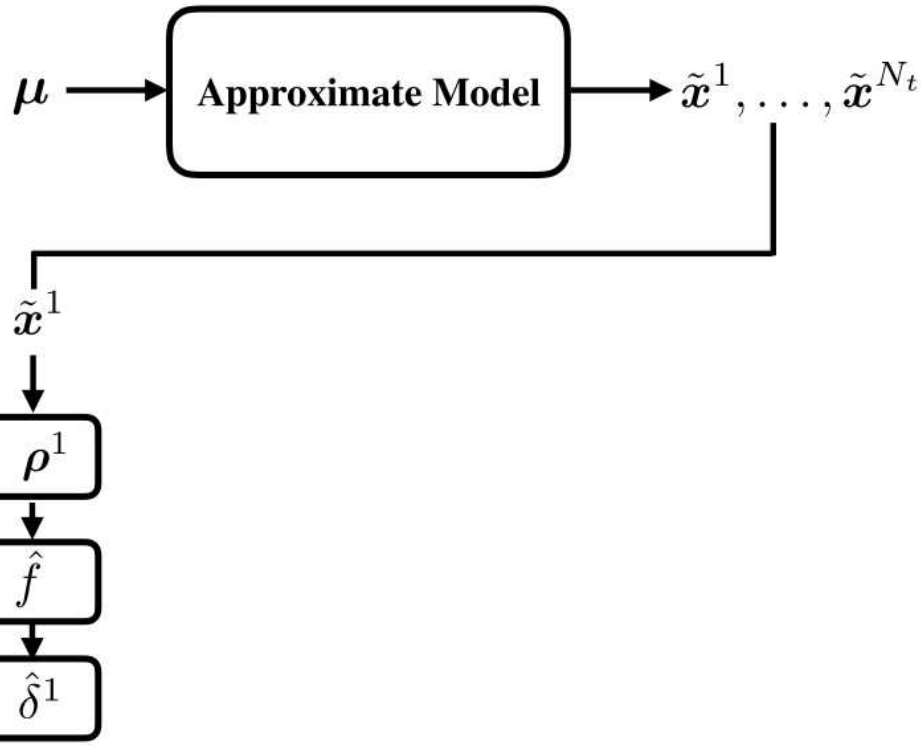
Time-Series Machine Learning Error Models

- We seek to address these short comings:
  - Inaccuracy + cost of dual weighted residuals/a posteriori bounds
  - Physically inconsistency of existing error models for dynamical systems
- We develop a new error modeling framework for dynamical systems
  - Time Series Machine Learning Error Models (T-MLEM)
  - Inspired from a posteriori error analysis and dual weighted residuals
  - Physically consistent
  - Easy to implement
  - Accurate
  - Requires data
- Extends the MLEM framework to dynamical systems

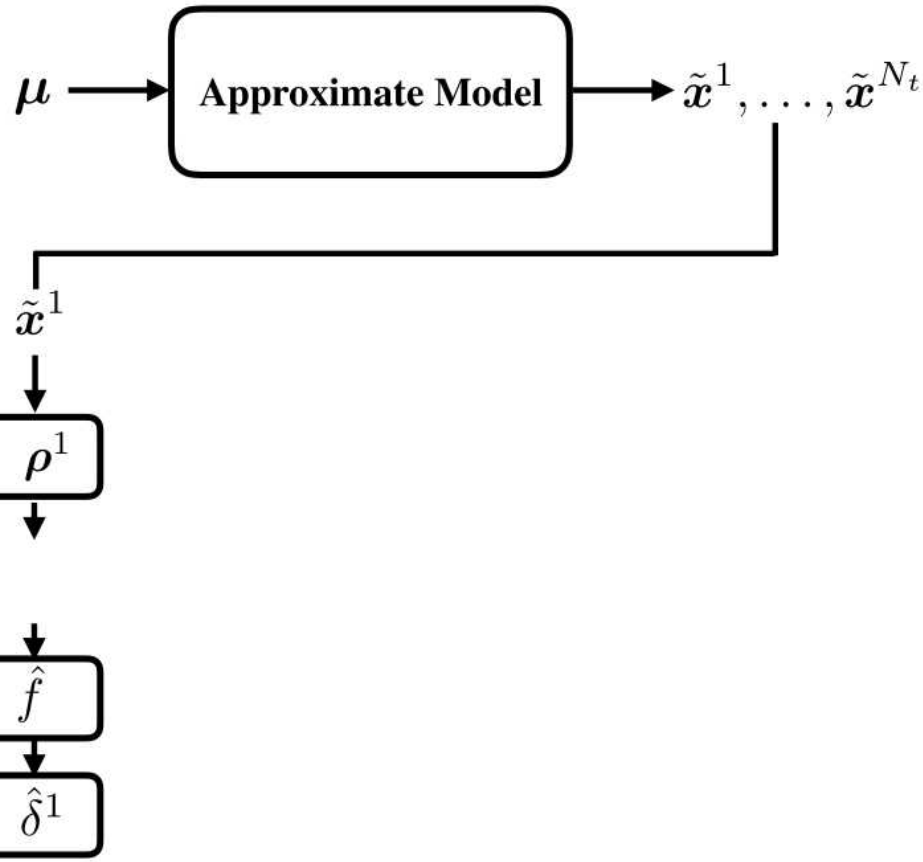
Time Series Machine Learning Error Models (T-MLEM)



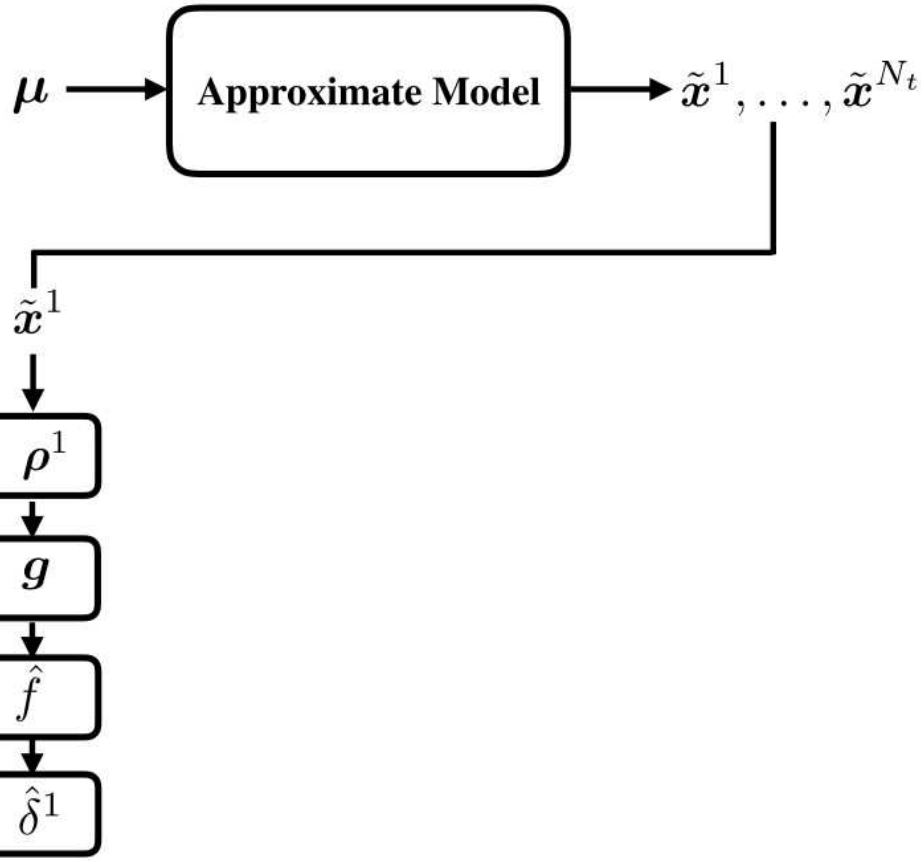
T-MLEM



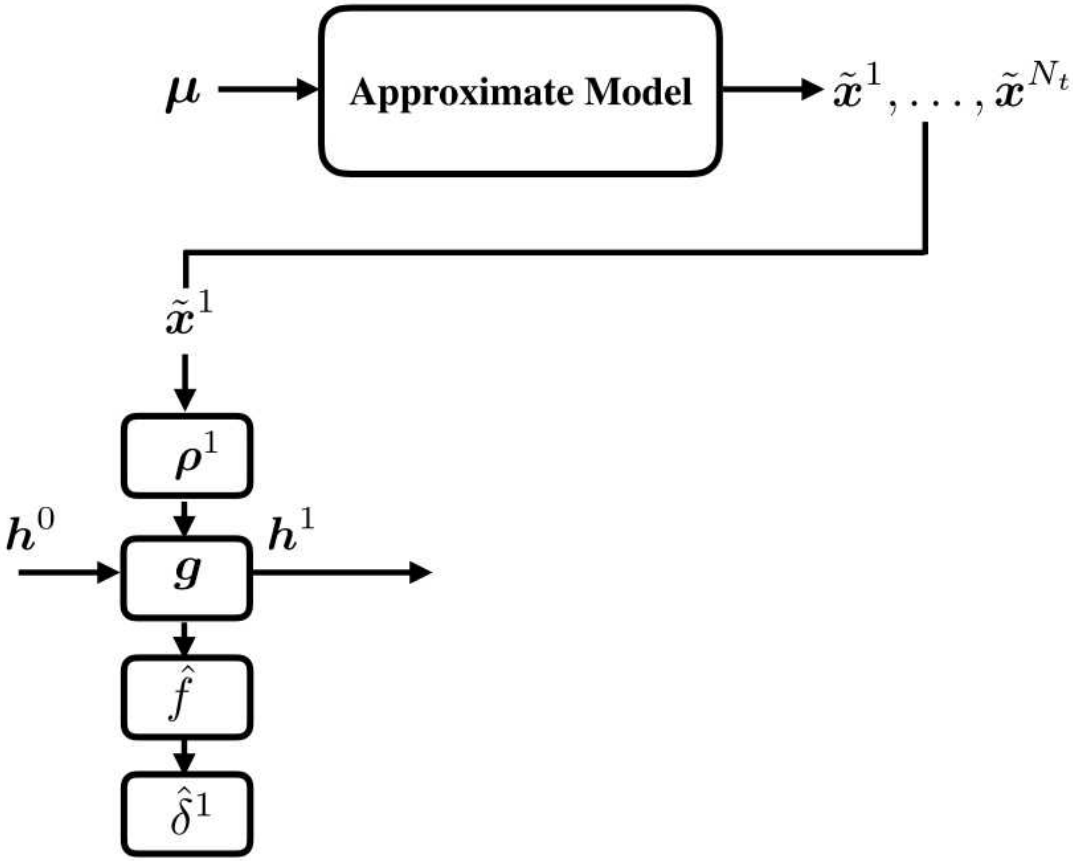
T-MLEM



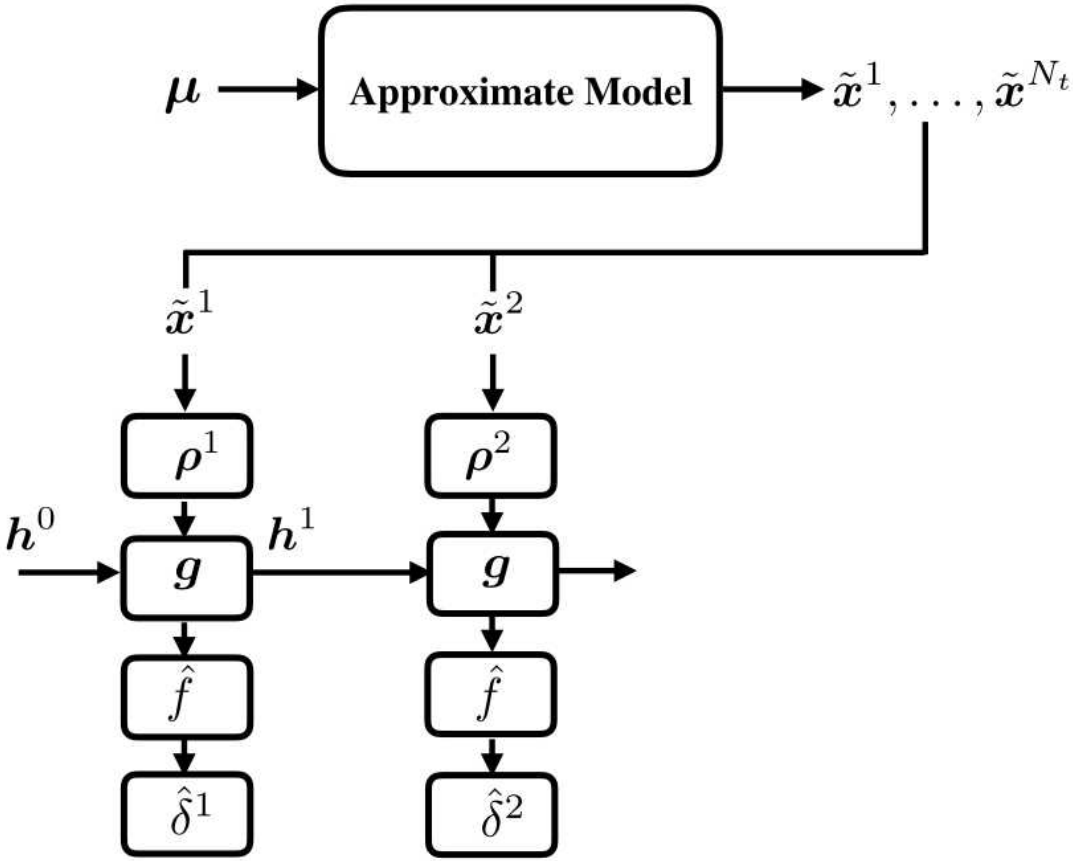
T-MLEM



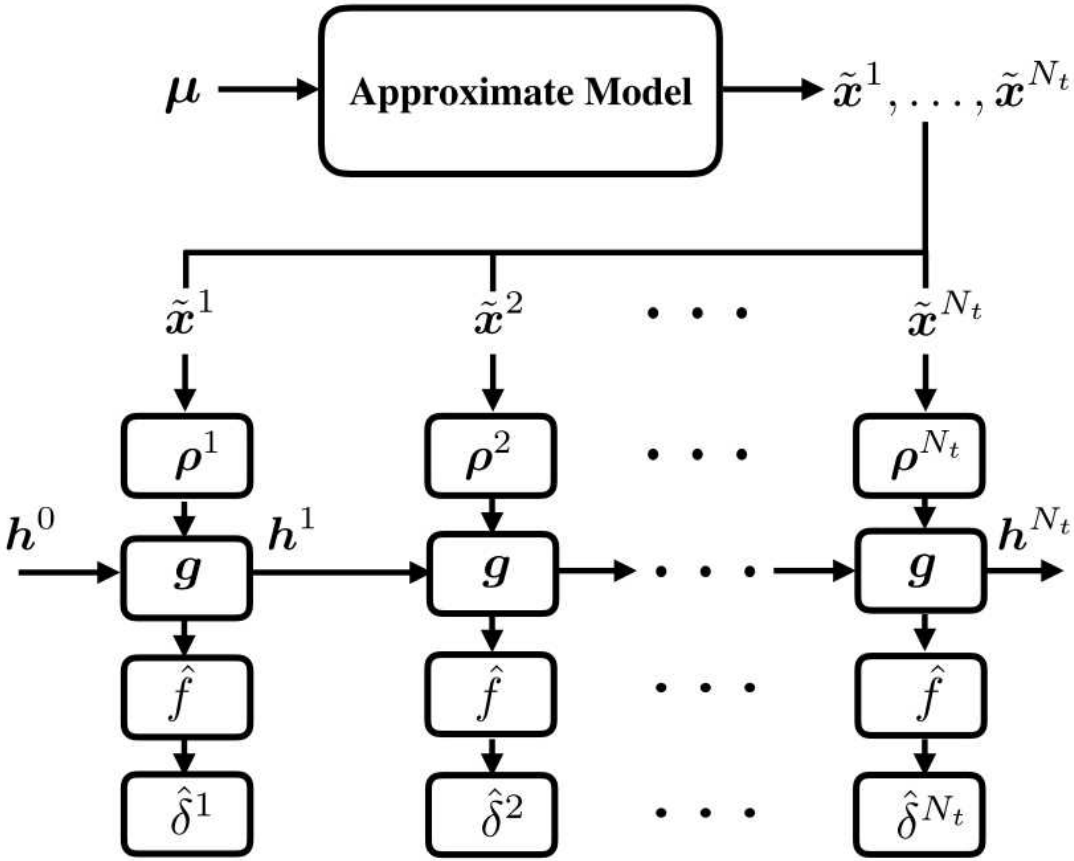
T-MLEM



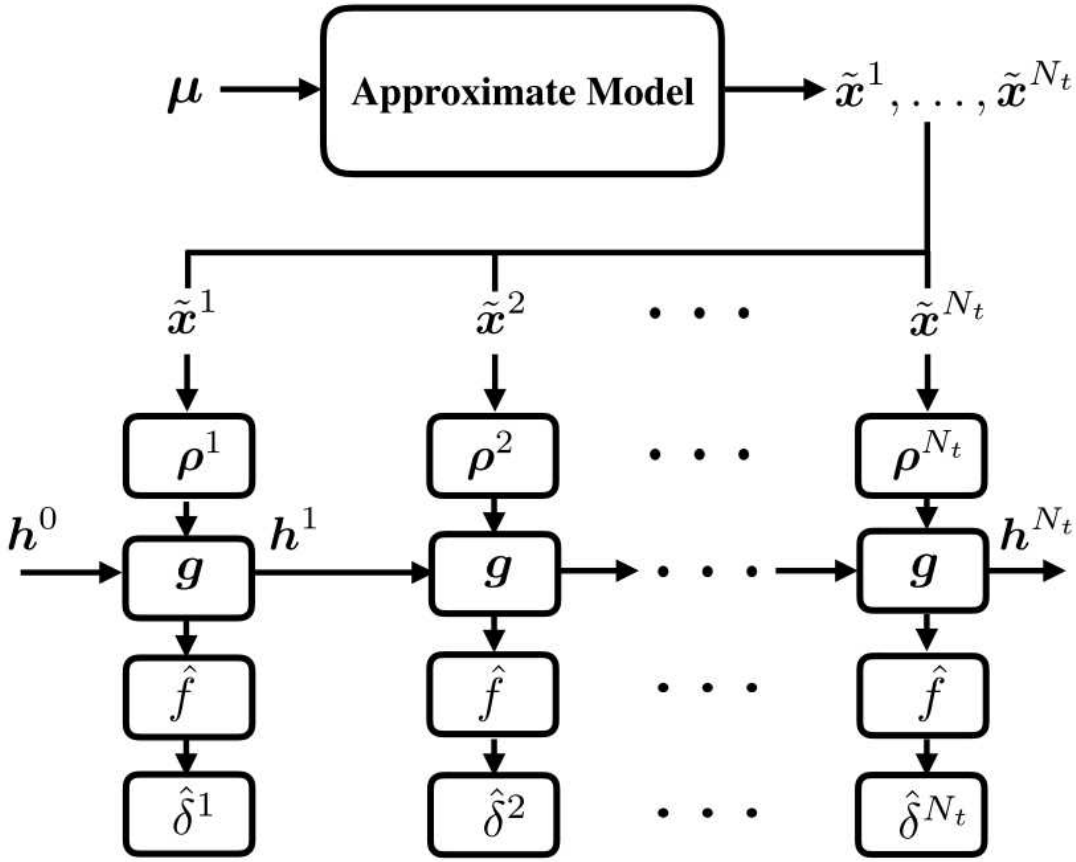
T-MLEM



T-MLEM

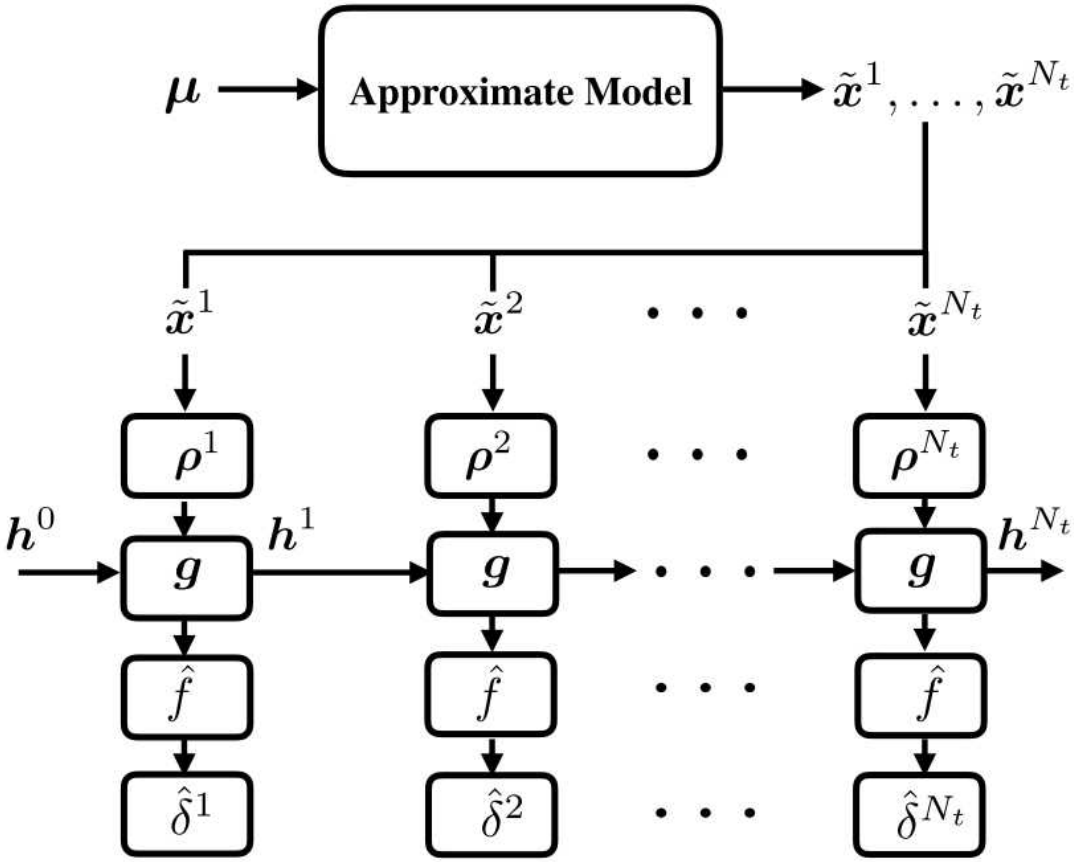


T-MLEM



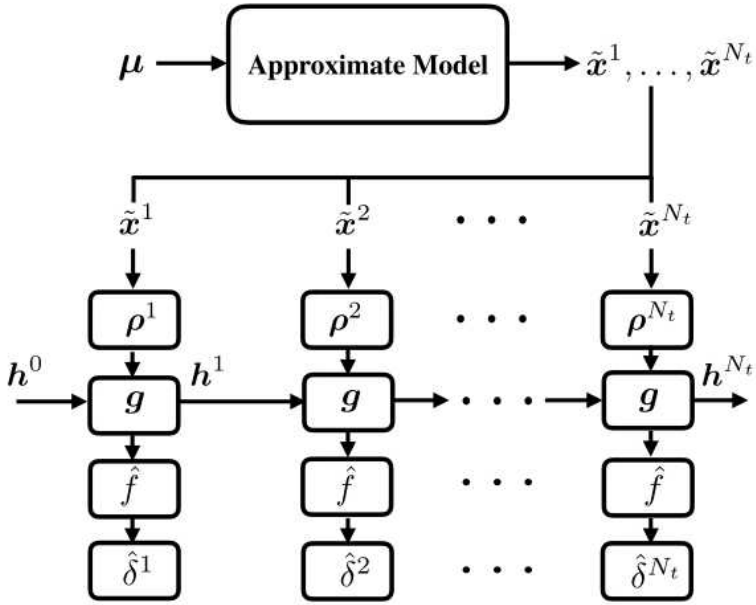
$$\hat{\delta}^n = \hat{f}(\boldsymbol{\rho}^n, \boldsymbol{h}^n),$$
$$\boldsymbol{h}^n = \mathbf{g}(\boldsymbol{\rho}^n, \boldsymbol{h}^{n-1})$$

T-MLEM



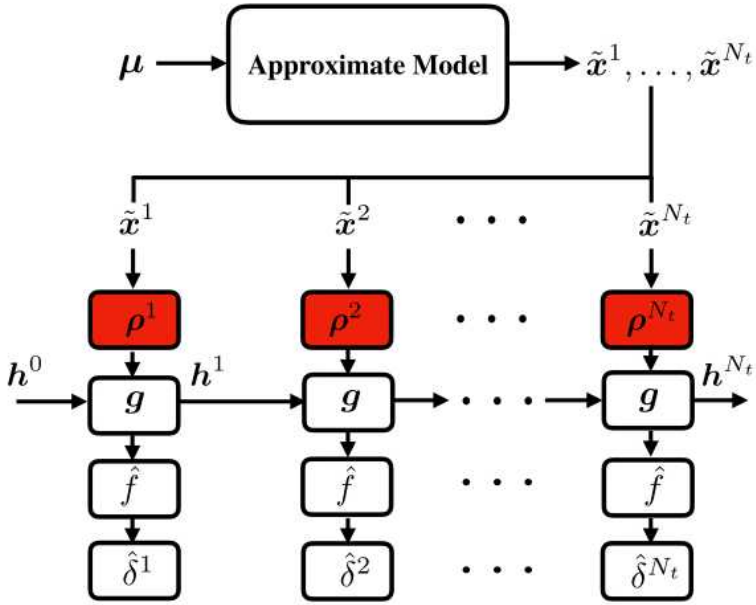
- T-MLEM introduces **latent variables**
- Governed by **recursive** relation
- Allows T-MLEM to capture recursive error

T-MLEM



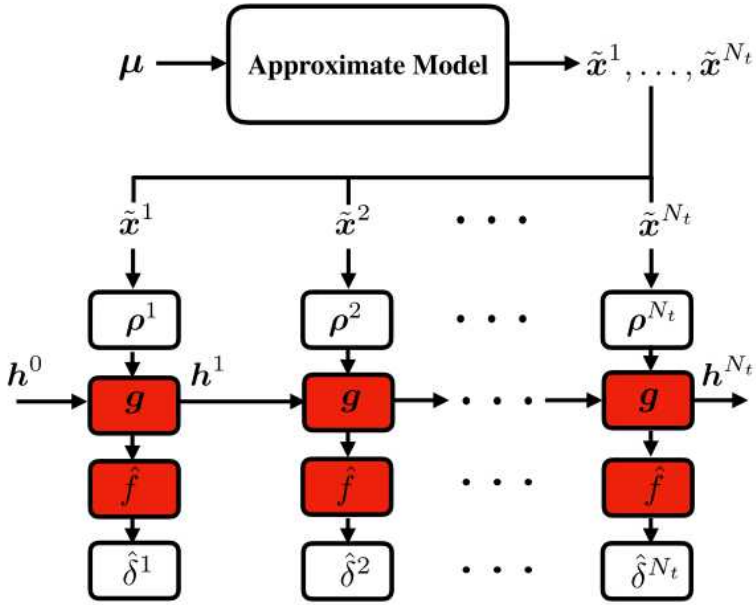
- T-MLEM requires:

T-MLEM



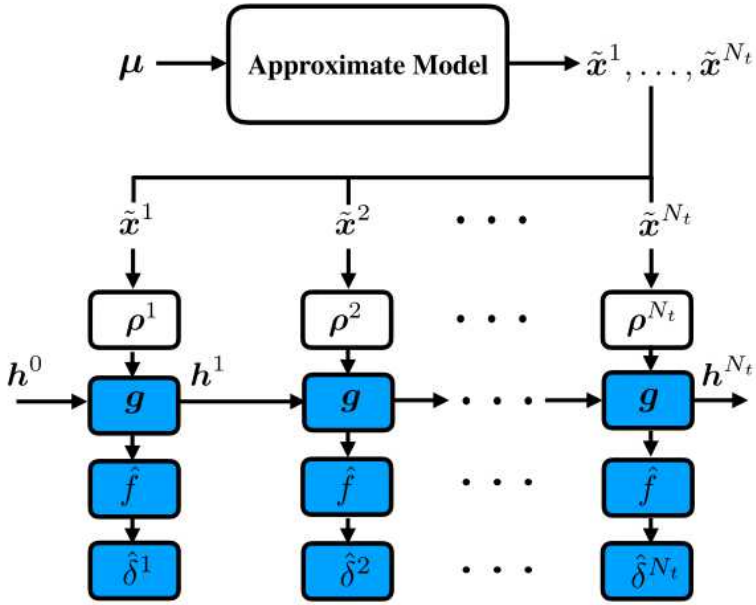
- T-MLEM requires:
  - **Feature engineering**

T-MLEM



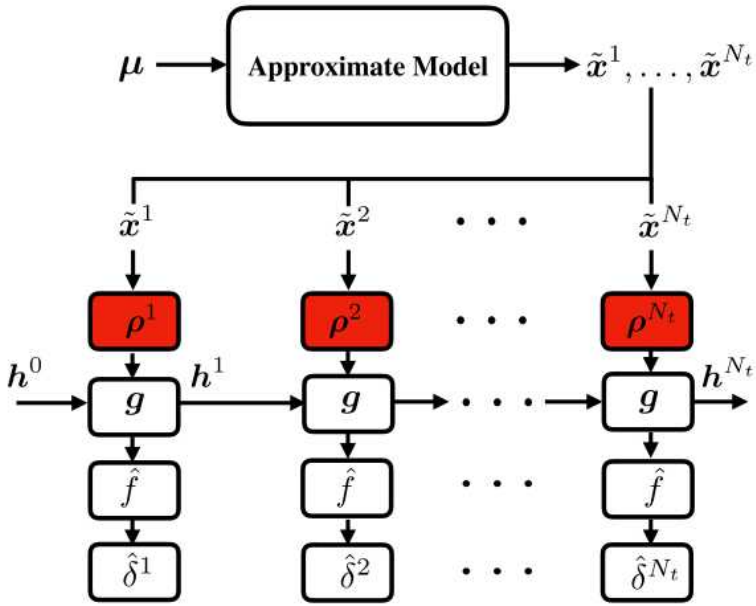
- T-MLEM requires:
  - Feature engineering
  - **Specification of regression functions**

T-MLEM



- T-MLEM requires:
  - Feature engineering
  - Specification of regression functions
  - **Training the regression function model**

Feature Engineering



- Need to define candidate features, $\boldsymbol{\rho}^n$
- We examine features inspired by *a posteriori* error bounds and dual weighted residuals

Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_{2} + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
  - Used in the Kennedy and O'Hagan "model discrepancy" approach
  - Free to compute
  - Not informative of error
  - Low-dimensional

Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_2 + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
- Feature 2: **Residual norm**, $\|\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})\|_2$
  - Directly appears in *a posteriori* error bounds
  - Low-dimensinal
  - Only informative of magnitude (not sign)

Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_{2} + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
- Feature 2: **Residual norm**, $\|\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})\|_{2}$
- Feature 3: **Residual**, $\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
  - Directly appears in *a posteriori* error bounds
  - High dimensional input feature
  - May require many data for training
  - Expensive to compute

Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_2 + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
- Feature 2: **Residual norm**, $\|\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})\|_2$
- Feature 3: **Residual**, $\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 4: **Residual principal components**, $\hat{\boldsymbol{r}}^{n}(\boldsymbol{\mu}) := \boldsymbol{\Phi}_{\boldsymbol{r}}^{T}(\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu}) - \bar{\tilde{\boldsymbol{r}}})$
  - Same advantageous of the full residual but is lower dimensional
  - Requires less data for training
  - Still expensive to compute

Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_2 + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
- Feature 2: **Residual norm**, $\|\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})\|_2$
- Feature 3: **Residual**, $\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 4: **Residual principal components**, $\hat{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 5: **Residual gappy principal components**, $\hat{\boldsymbol{r}}_{g}^{n}(\boldsymbol{\mu}) := [\mathbf{P}\boldsymbol{\Phi}_{\boldsymbol{r}}]^{+}\mathbf{P}(\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu}) - \bar{\tilde{\boldsymbol{r}}})$.
  - Same advantages of Feature 5, but is cheaper to compute!

Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_2 + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
- Feature 2: **Residual norm**, $\|\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})\|_2$
- Feature 3: **Residual**, $\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 4: **Residual principal components**, $\hat{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 5: **Residual gappy principal components**, $\hat{\boldsymbol{r}}_{g}^{n}(\boldsymbol{\mu})$
- Feature 6: **Sampled residual**, $\mathbf{P}\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
  - Same advantages of Feature 6

Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_2 + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
- Feature 2: **Residual norm**, $\|\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})\|_2$
- Feature 3: **Residual**, $\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 4: **Residual principal components**, $\hat{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 5: **Residual gappy principal components**, $\hat{\boldsymbol{r}}_{g}^{n}(\boldsymbol{\mu})$
- Feature 6: **Sampled residual**, $\mathbf{P}\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$

- Methods can be combined to create many candidate feature sets

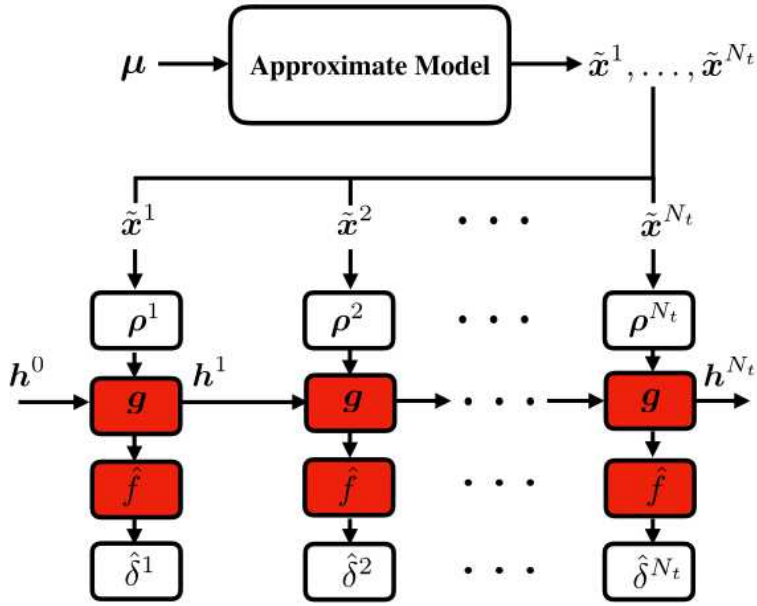
Feature Engineering

$$\delta_{\boldsymbol{x}}^{n} \leq \frac{1}{h} \left\| \boldsymbol{r}^{n} \left(\tilde{\boldsymbol{x}}^{n}; \tilde{\boldsymbol{x}}^{n-1}, \boldsymbol{\mu} \right) \right\|_2 + \gamma \delta_{\boldsymbol{x}}^{n-1}$$

- Feature 1: **Parameters**, $\boldsymbol{\mu}$
- Feature 2: **Residual norm**, $\|\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})\|_2$
- Feature 3: **Residual**, $\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 4: **Residual principal components**, $\hat{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$
- Feature 5: **Residual gappy principal components**, $\hat{\boldsymbol{r}}_{g}^{n}(\boldsymbol{\mu})$
- Feature 6: **Sampled residual**, $\mathbf{P}\tilde{\boldsymbol{r}}^{n}(\boldsymbol{\mu})$

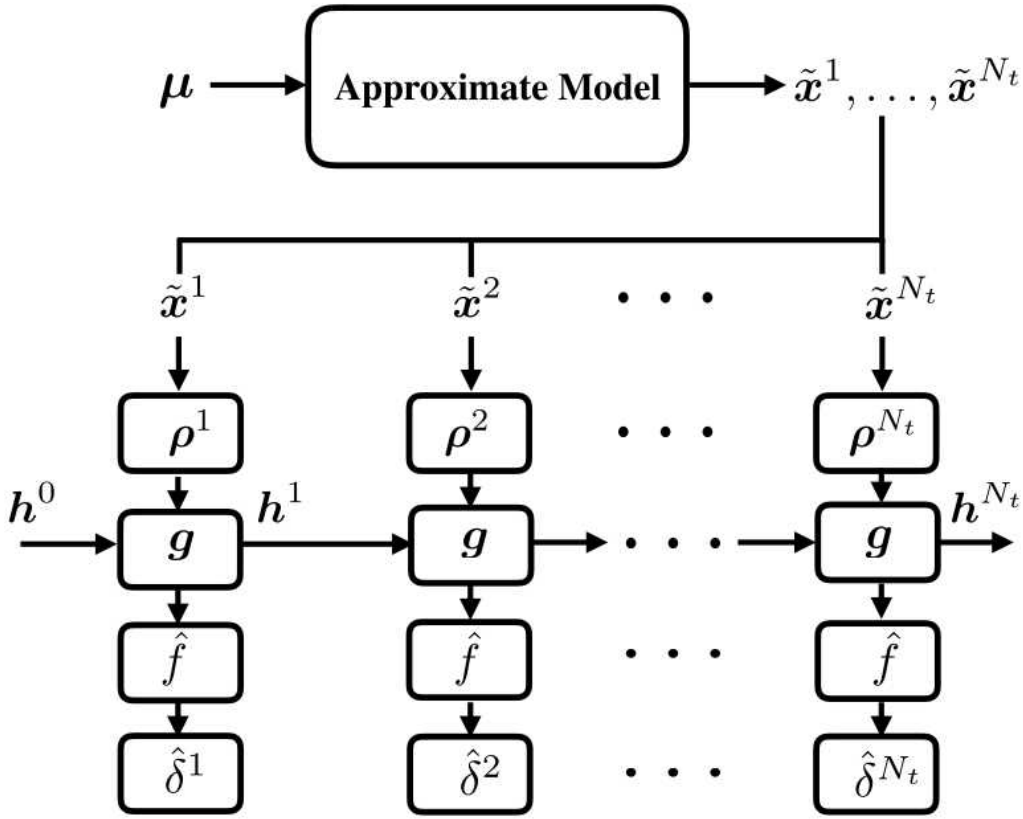
- Methods can be combined to create many candidate feature sets
- **Takeaway: We consider residual-based features**

Regression-function models



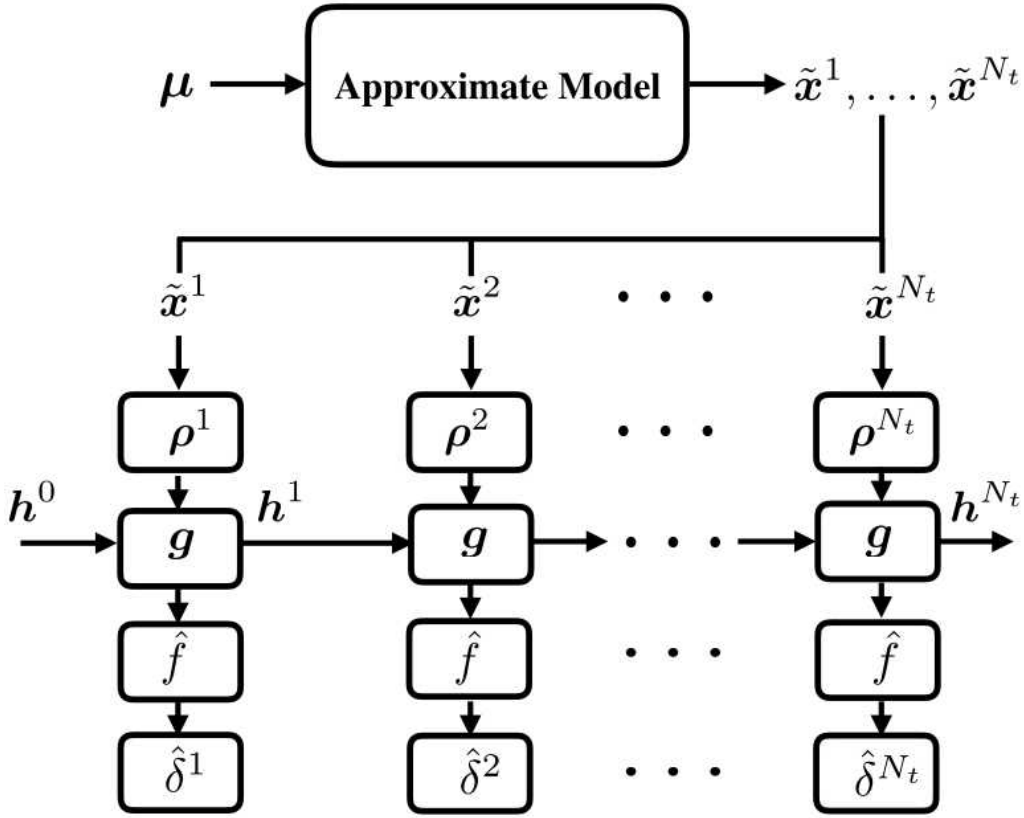
- Now need to specify the regression functions $\hat{f}$ and $\mathbf{g}$
  - Provides the mapping from input features to the error
- Examine **three** categories of methods

Category 3 Models



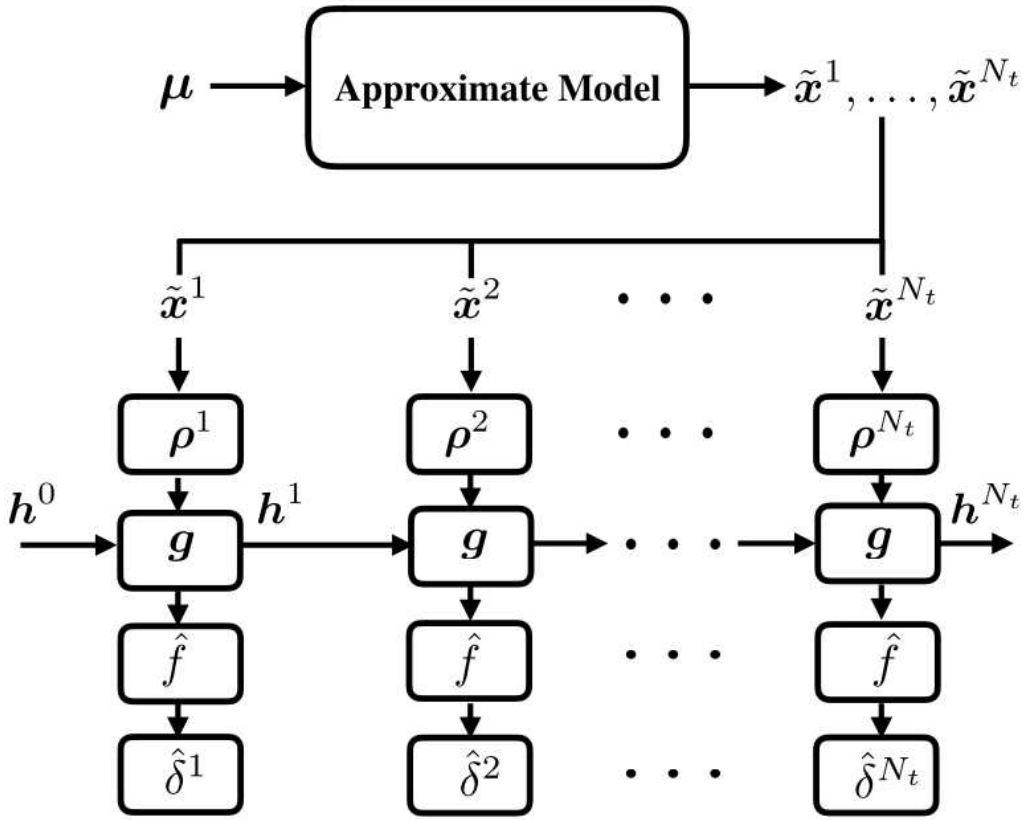
$$\hat{\delta}^n = \hat{f}(\boldsymbol{\rho}^n, \boldsymbol{h}^n),$$
$$\boldsymbol{h}^n = \mathbf{g}(\boldsymbol{\rho}^n, \boldsymbol{h}^{n-1})$$

Category 3 Models

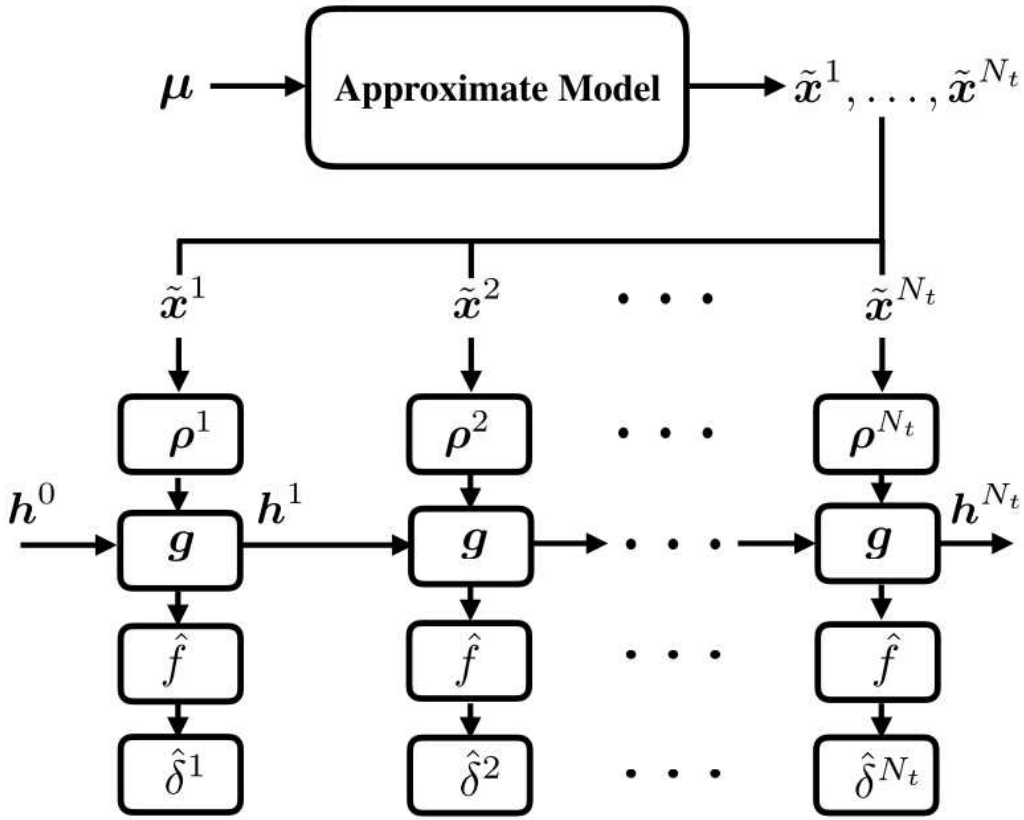


- Examine three Category 3 models:
  - Latent Auto-regressive model: $\hat{f}$ and $\mathbf{g}$ are linear
  - Recurrent neural network: $\hat{f}$ and $\mathbf{g}$ are non-linear
  - Long short-term memory network: $\hat{f}$ and $\mathbf{g}$ are non-linear

Category 2 Models

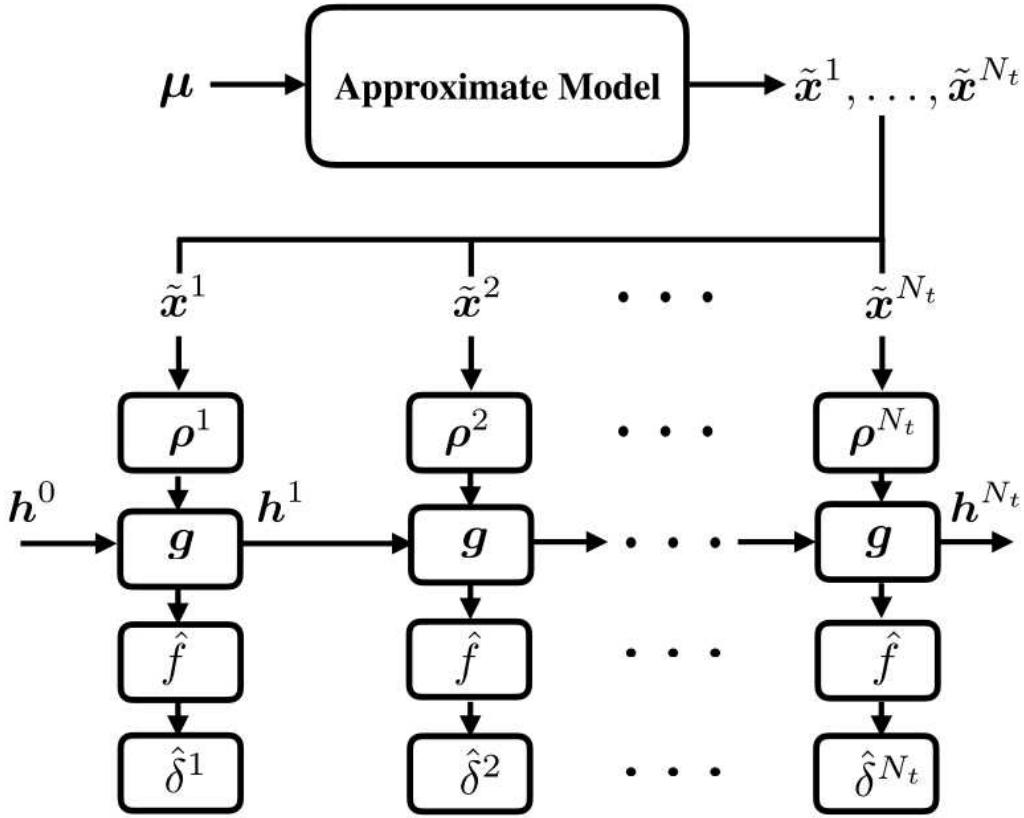


Category 2 Models



- Set latent state to be equal to previous prediction

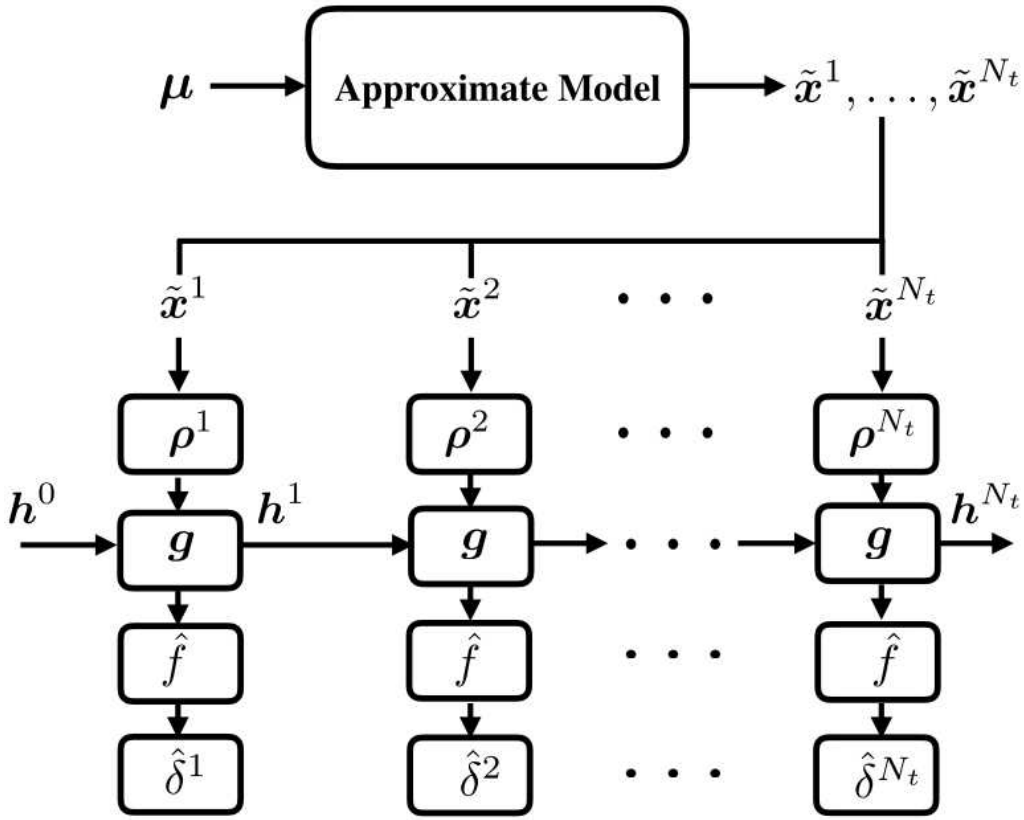
Category 2 Models



$$\hat{\delta}^n = \hat{f}(\boldsymbol{\rho}^n, \boldsymbol{h}^n),$$

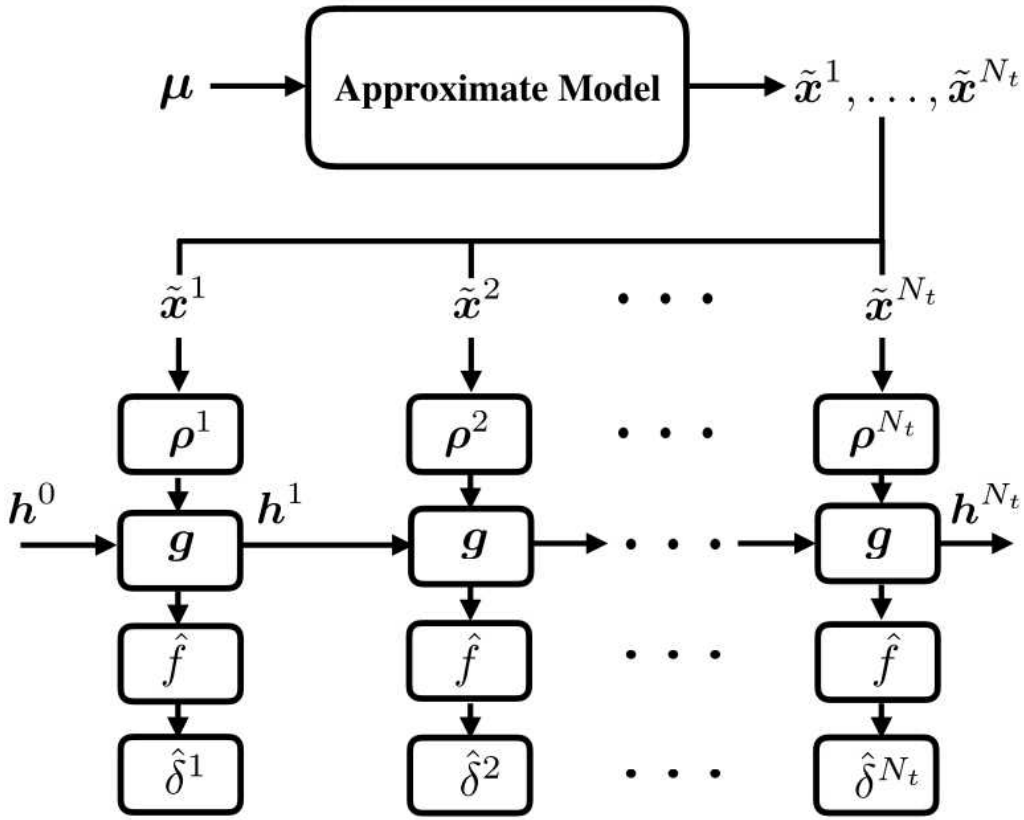
$$\boldsymbol{h}^n \equiv \hat{\delta}^{n-1}$$

Category 2 Models

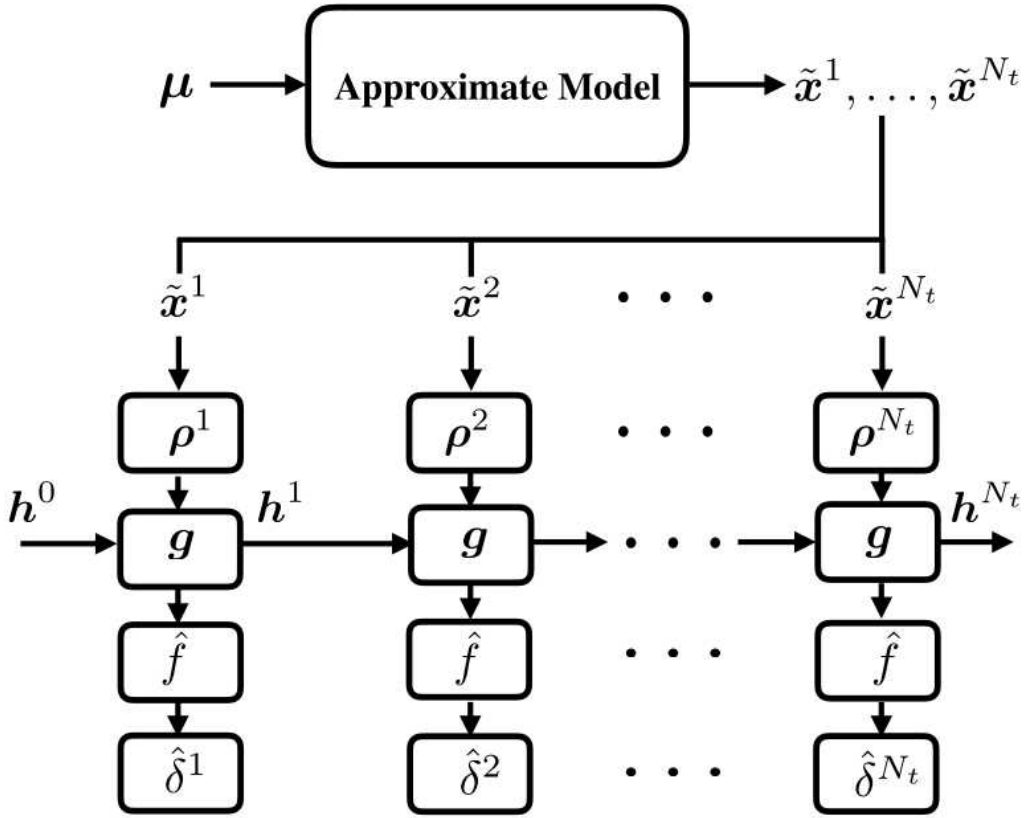


- Examine two Category 2 models:
  - Auto-regressive model: $\hat{f}$ is linear
  - Integrated Neural Network: $\hat{f}$ is a neural network

Category 1 Models

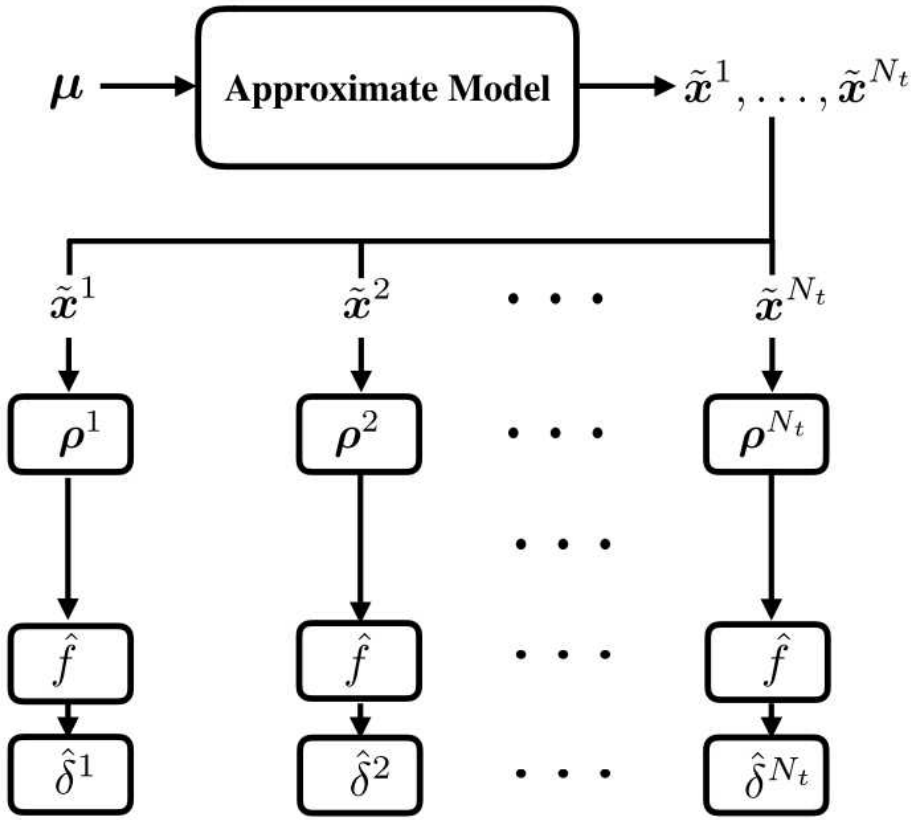


Category 1 Models



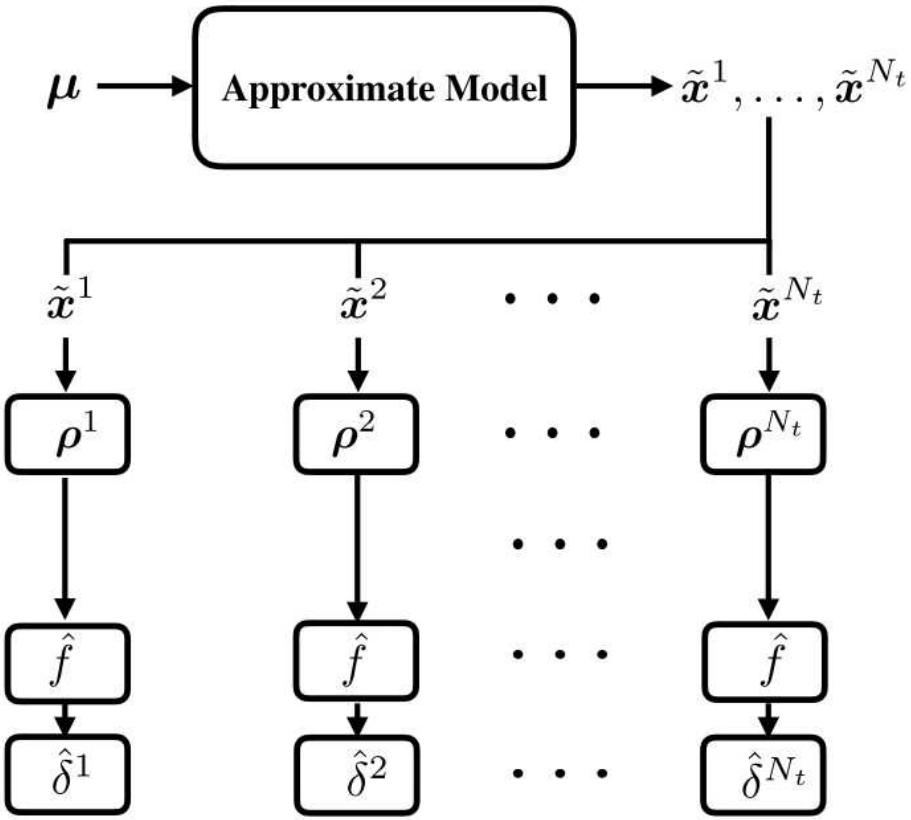
- Turn off the latent state
- Turn off recursion

Category 1 Models



$$\hat{\delta}^n = \hat{f}(\rho^n)$$

Category 1 Models



- Examine two Category 1 models:
  - kNN: $\hat{f}$ is a k-nearest neighbors method
  - ANN: $\hat{f}$ is a neural network

Model Summary

- **Category 1:**
  - No rercursive dynamics
- **Category 2:**
  - Latent state of dimension 1

$$\boldsymbol{h}^n = \hat{\delta}^{n-1}$$

  - Linear recursive dynamics
- **Category 3:**
  - Arbitrary latent state dimension
  - Linear/non-linear recursive dynamics

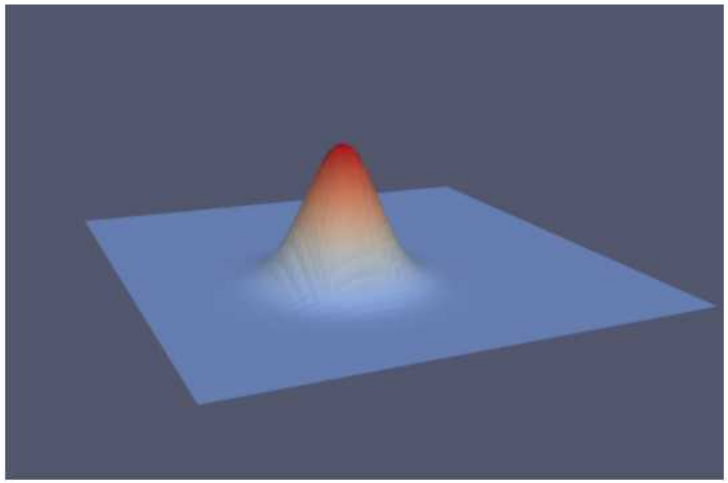
Data Generation

- Lastly, we generate the **training** data, **validation** data, and **testing** data
  - Training data: data used to train (optimize the models)
  - Validation data: independent data-set used for hyper-parameter and model selection
  - Test data: independent data-set used to assess the performance of the models
- These data are generated by executing the FOM and approximate models for samples of the parameter instances

Numerical Experiments

Numerical Example: ROM of Shallow Water Equations

- Solve the shallow water equations parameterized by
  - Gravity: $\mu_1 = [3, 9]$
  - Transient water height: $\mu_2 = [0.05, 0.2]$



$$\frac{\partial h}{\partial t} + \frac{\partial}{\partial x}(hu) + \frac{\partial}{\partial y}(hv) = 0,$$

$$\frac{\partial hu}{\partial t} + \frac{\partial}{\partial x}(hu^2 + \frac{1}{2}\mu_1 h^2) + \frac{\partial}{\partial y}(huv) = 0,$$

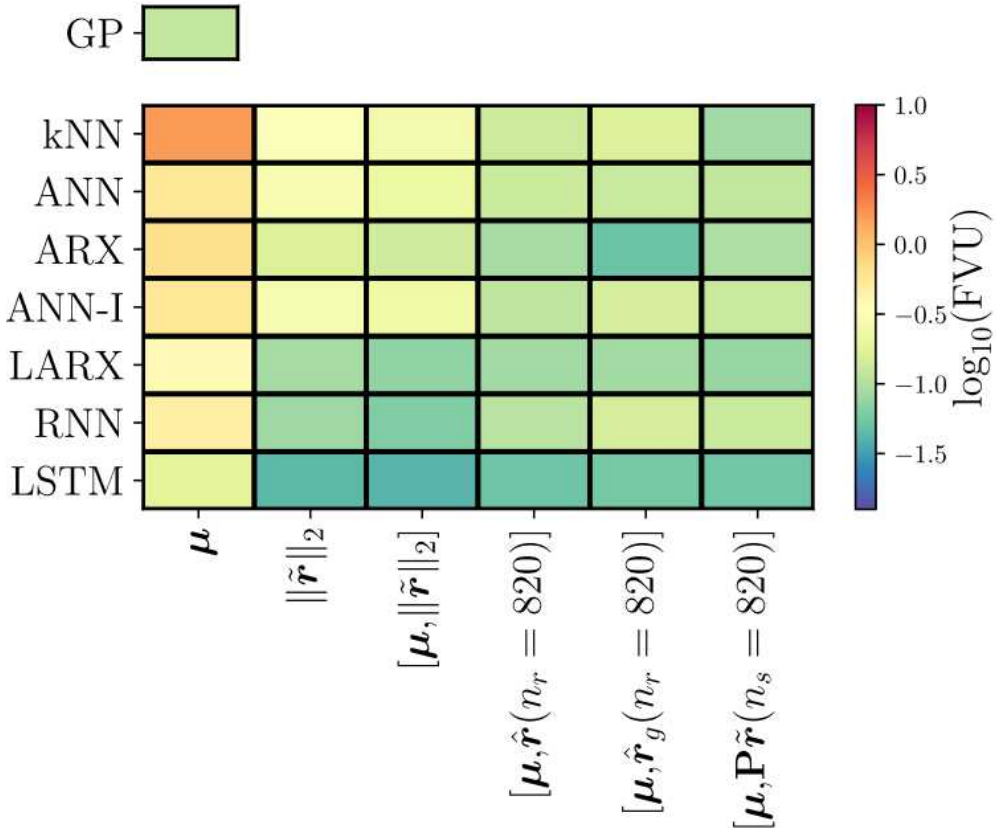
$$\frac{\partial hv}{\partial t} + \frac{\partial}{\partial x}(huv) + \frac{\partial}{\partial y}(hv^2 + \frac{1}{2}\mu_1 h^2) = 0,$$

$$h(t=0) = h_0 + \mu_2 e^{(x-1)^2+(y-1)^2}, u_0 = 0, v_0 = 0,$$

- FOM is a 4th order discontinuous Galerkin scheme
  - Contains 12k degrees of freedom
  - Approximate model is a ROM of dimension $K = 78$

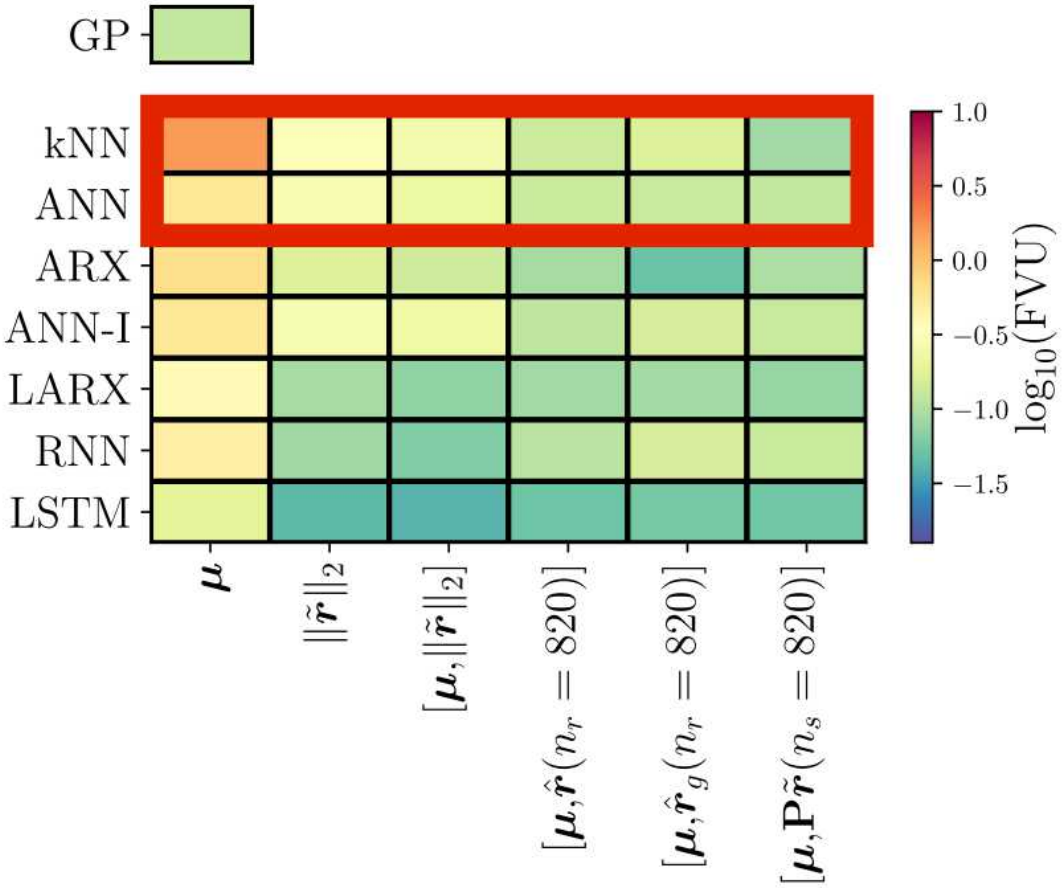
Regression Results

- Prediction for normed state error, $\delta_{\boldsymbol{x}}^n = \left\| \tilde{\boldsymbol{x}}^n - \boldsymbol{x}^n \right\|_2$



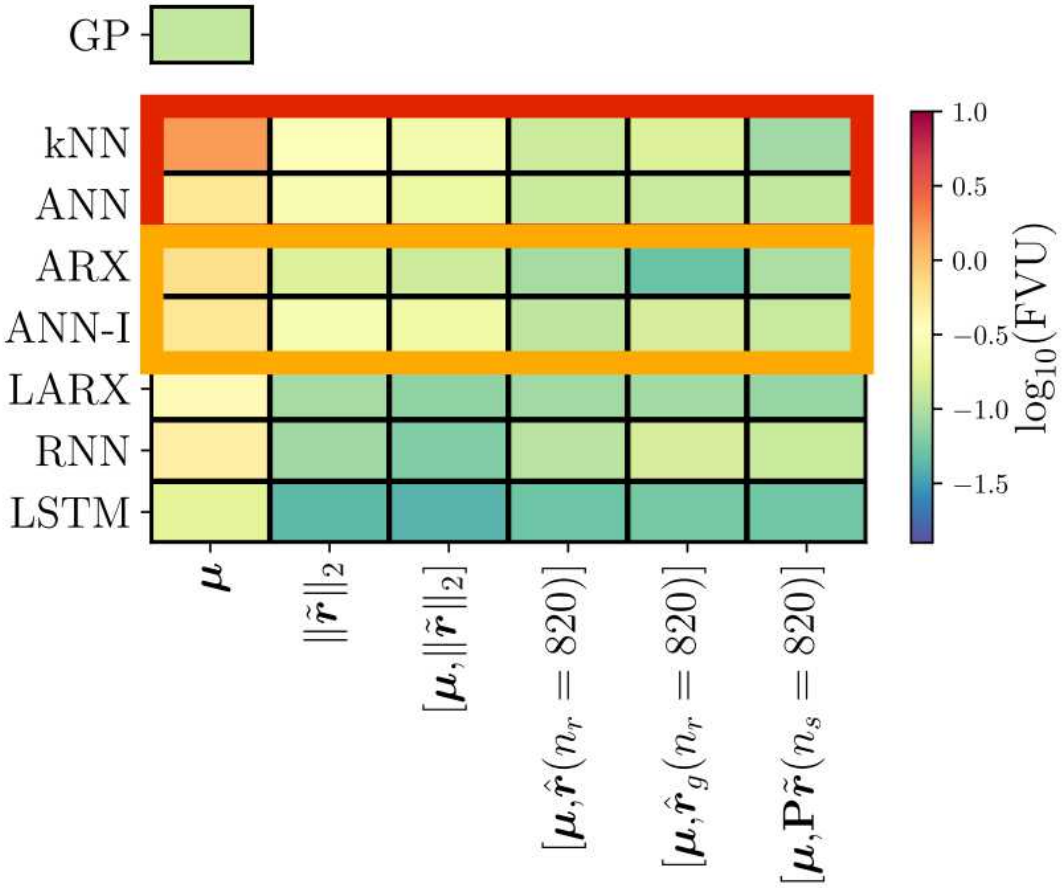
Regression Results

- Prediction for normed state error, $\delta_{\boldsymbol{x}}^n = \|\tilde{\boldsymbol{x}}^n - \boldsymbol{x}^n\|_2$



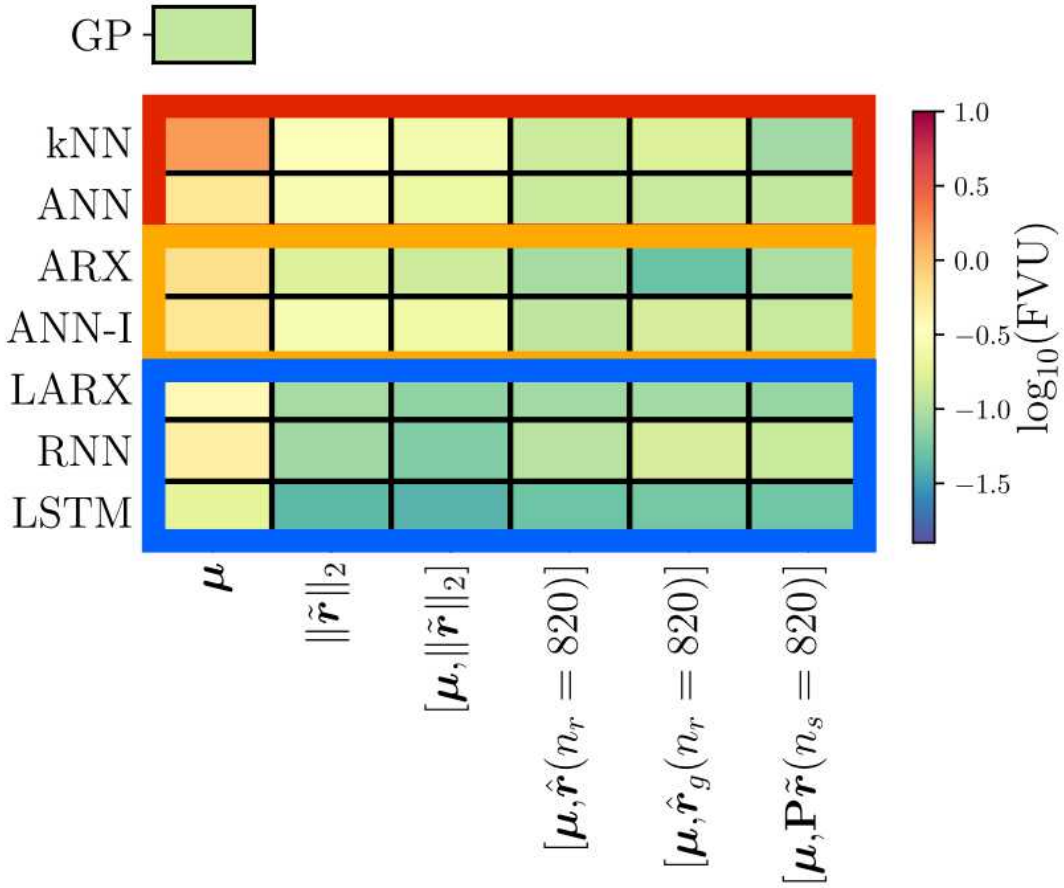
Regression Results

- Prediction for normed state error, $\delta_{\boldsymbol{x}}^n = \left\| \tilde{\boldsymbol{x}}^n - \boldsymbol{x}^n \right\|_2$



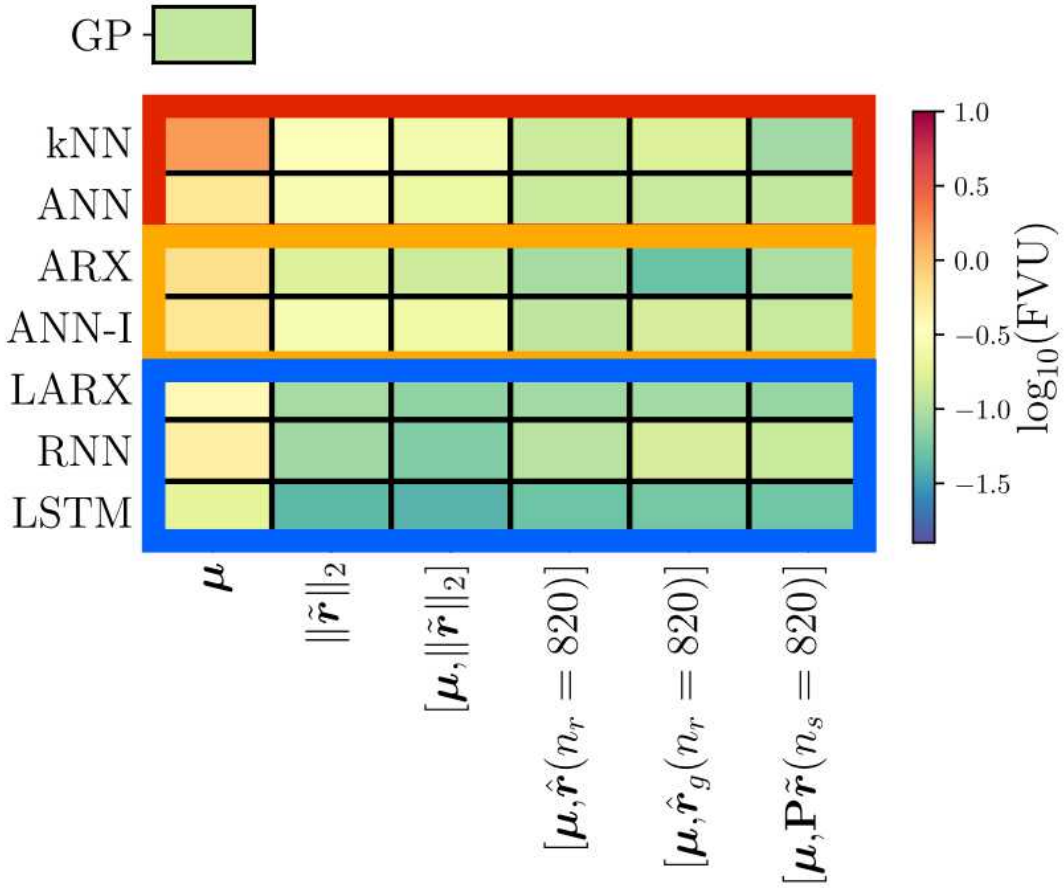
Regression Results

- Prediction for normed state error, $\delta_{\boldsymbol{x}}^n = \|\tilde{\boldsymbol{x}}^n - \boldsymbol{x}^n\|_2$



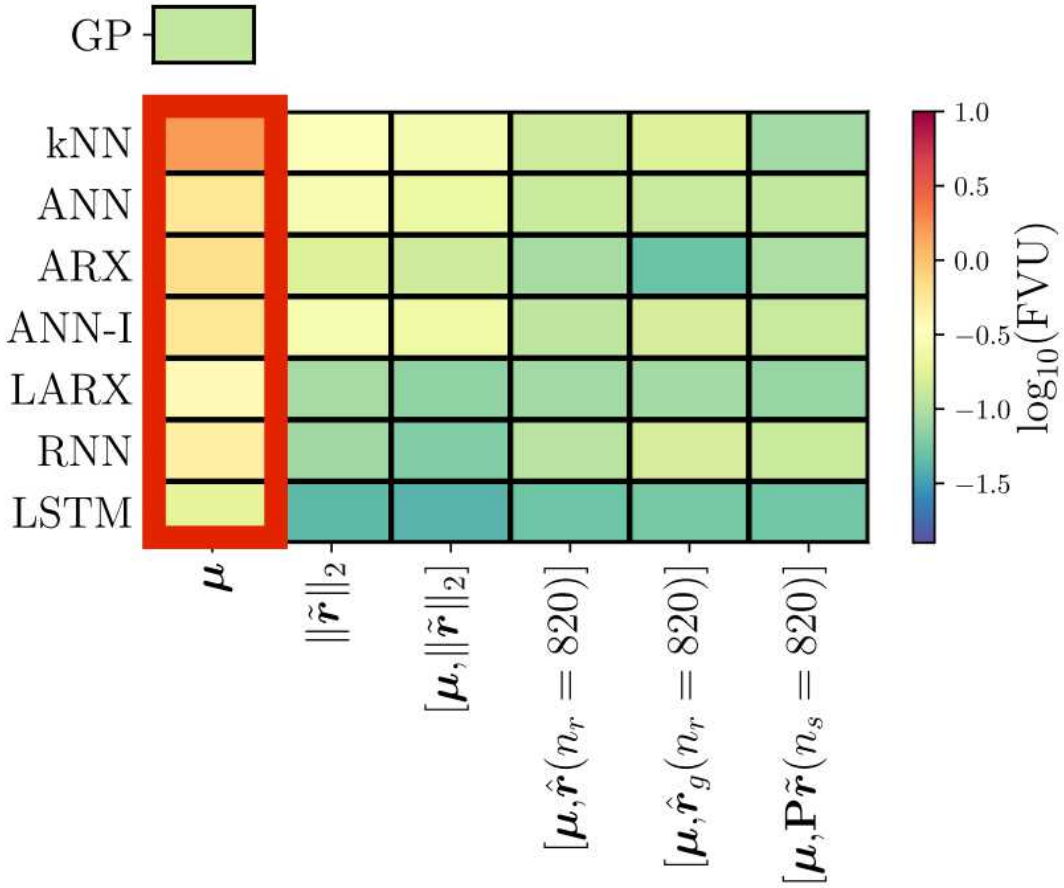
Regression Results

- Prediction for normed state error, $\delta_{\boldsymbol{x}}^n = \|\tilde{\boldsymbol{x}}^n - \boldsymbol{x}^n\|_2$



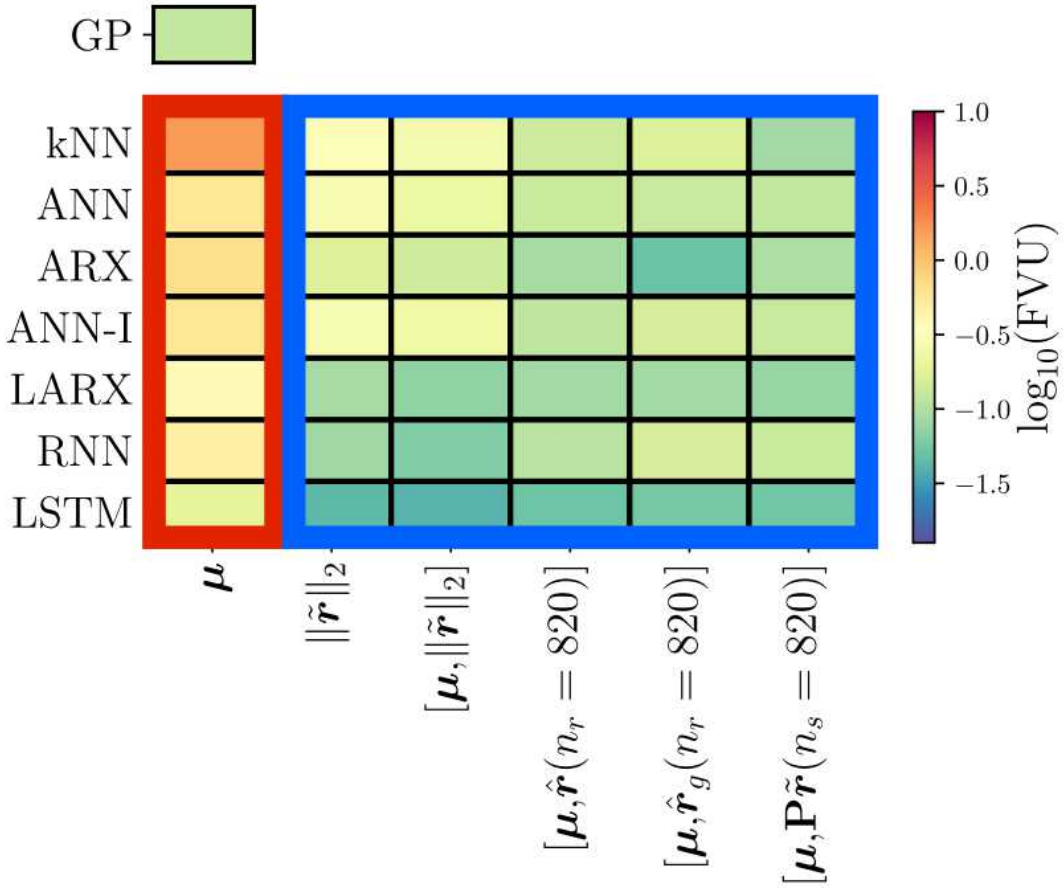
Regression Results

- Prediction for normed state error, $\delta_{\boldsymbol{x}}^n = \|\tilde{\boldsymbol{x}}^n - \boldsymbol{x}^n\|_2$



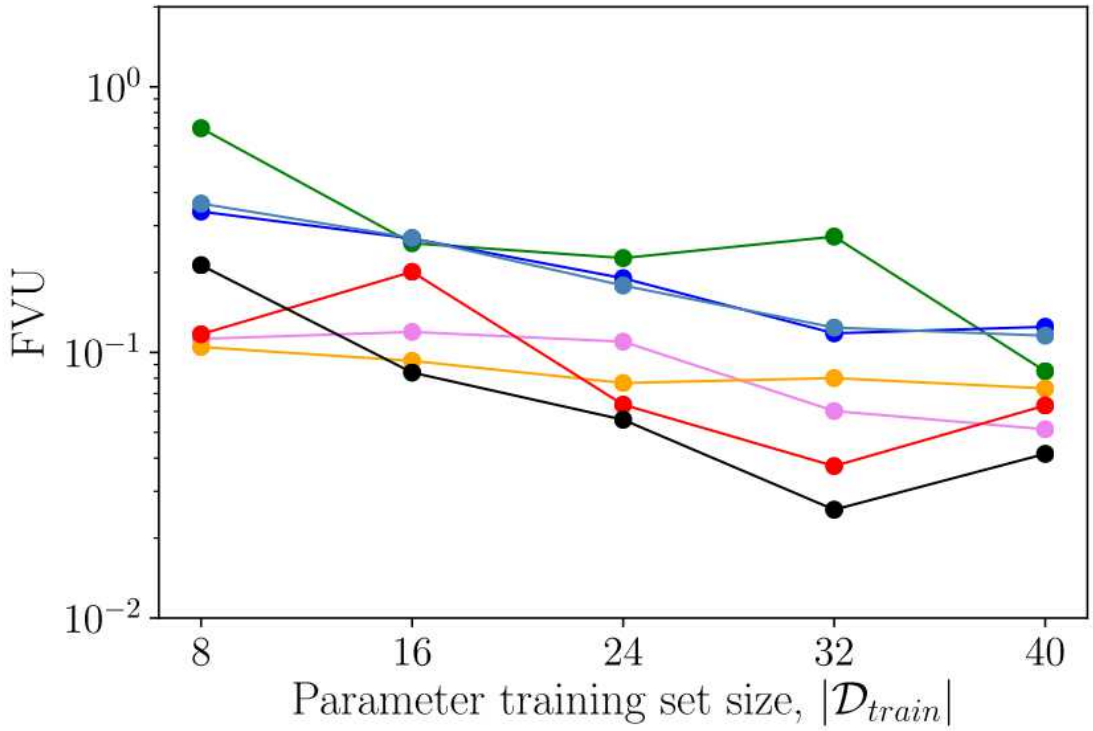
Regression Results

- Prediction for normed state error, $\delta_{\boldsymbol{x}}^n = \left\| \tilde{\boldsymbol{x}}^n - \boldsymbol{x}^n \right\|_2$



Regression Results

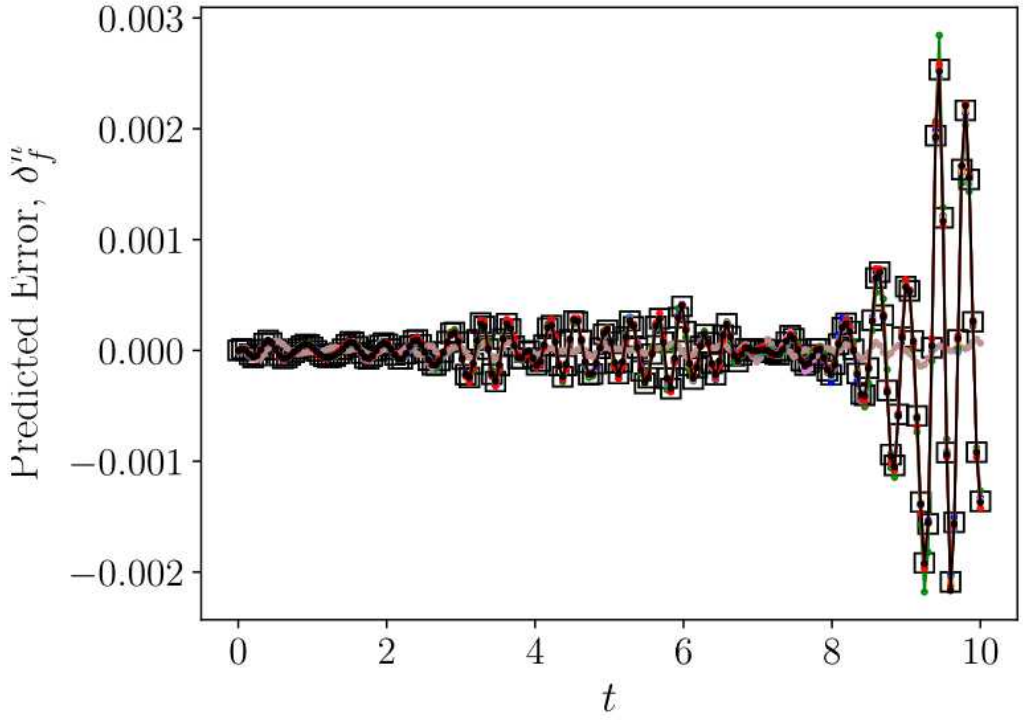
- Next assess the impact of dataset size:



- **RNN and LSTM again perform best**

Regression Results

- Error response as a function of time:



$$|\mathcal{D}_{\text{train}}| = 40$$

Error response $\delta_{\boldsymbol{x}}$.

Summary

- Outlined an error modeling framework for approximate solutions to parameterized dynamical systems
  - Framework comprises an extension of Ref.[1] to dynamical systems
- Key components of framework:
  1. **Feature Engineering**: Engineer features from classical error analysis techniques
  2. **Recursive regression function construction**: Constructs a recursive regression method that provides the feature–response mapping
- Numerical experiments demonstrate:
  - The LSTM network yielded the best performance
  - Residual-based terms were the best performing features

Selected References

- Parish, E.J., and Carlberg, K.T., *"Time-series machine-learning error models for approximate solutions to parameterized dynamical systems"*, arXiv:1907.11822

- Freno, B. and Carlberg, K.T., *Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations*, (2019)
- Trehan, S., Carlberg, K., and Durlofsky, L.J., *Error modeling for surrogates of dynamical systems using machine learning*, (2017)
- Pagani, S., Manzoni, A., and Quarteroni A., *Efficient State/Parameter Estimation in Nonlinear Unsteady PDEs by a Reduced Basis Ensemble Kalman Filter*, (2017)
- Drohmann, M. and Carlberg, K., *The ROMES method for statistical modeling of reduced-order-model error*, (2015)
- Kennedy, M., and O'Hagan, A., *Bayesian calibration of computer models*, (2002)

Thank you for your time!

This work was supported by Sandia's John von Neumann Postdoctoral Fellowship. This presentation describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.