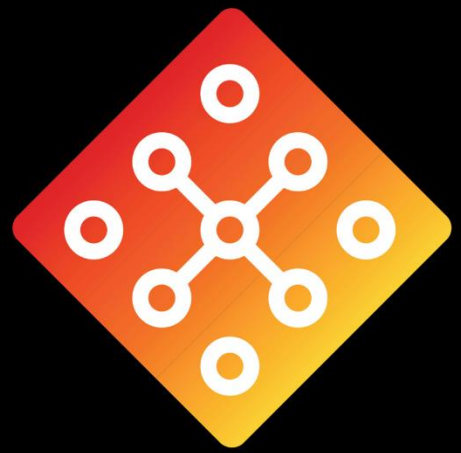


CAEML

CENTER FOR ADVANCED ELECTRONICS THROUGH MACHINE LEARNING

Georgia Tech | ILLINOIS | NC STATE UNIVERSITY

AN NSF INDUSTRY/UNIVERSITY COOPERATIVE RESEARCH CENTER

Machine Learning for Electronic Design Automation

ELYSE ROSENBAUM

November 6, 2018

Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

Limits of Modern EDA

- **Design respins have not been eliminated**
- **Many of the observed failures during qualification testing are the direct result of an insufficient modeling capability**
  - Sources of such failures include mistuned analog circuits, signal timing errors, reliability problems, and crosstalk [1]
  - Variability cannot be modeled in a manner that is both accurate and computationally efficient
- **Simulation-based design optimization has had only limited success**
  - Simulation "in-the-design-loop" often too slow and leads to impractical designs
- **Proposal: use machine learning algorithms to overcome those hurdles!**

[1] Harry Foster, "2012 Wilson Research Group Functional Verification Study,"
http://www.mentor.com/products/fv/multimedia/the-2012-wilson-research-group-functional-verification-studyview

Center for Advanced Electronics through Machine Learning (CAEML)

- **Vision statement**
  - To enable fast, accurate design and verification of microelectronic circuits and systems by creating machine-learning algorithms to derive models used for electronic design automation
- **An NSF Industry/University Cooperative Research Center**
  - University of Illinois at Urbana-Champaign
    - Lead site. Prof. Elyse Rosenbaum, Center Director
  - Georgia Tech
    - Prof. Madhavan Swaminathan, Site Director
  - North Carolina State University
    - Prof. Paul Franzon, Site Director

CAEML Research Thrusts

- Theory and Machine Learning Efficiency
- Design and System Optimization
- Modeling and Simulation
- Verification
- Reliability and Security

Solvers and Optimizers

- **CAEML researchers utilize open source and commercial software to get the job done**
- **In particular, several Matlab toolboxes**
  - System identification
  - Global optimization
  - Machine learning
  - Neural networks

CAEML Activities

- **Design Optimization**
  - Example: Clock skew minimization for 3D-IC
- **Generative Modeling**
  - Example: Stochastic models of PCB interconnects
- **Circuit macro-modeling**
  - Example: Recurrent neural network
  - Example: System identification
- **Other …**
  - Deep networks (for DRC)
  - Causal inference (for hardware failure models)
  - Security threat detection

Introduction to Bayesian Optimization

$$P(f|\boldsymbol{D}) = \frac{P(\boldsymbol{D}|f)P(f)}{P(\boldsymbol{D})} \propto P(\boldsymbol{D}|f)P(f)$$

Posterior — Likelihood — Prior Model

- **Model *f* as a random process**
  - Usually a GP
- **Obtain next training sample $f(X_i)$, where $X_i = \underbrace{argmax}_{X}\{P(f(X_i) > f(X^+))\}$**
  - Maximizes PI, probability of improvement
- **Other acquisition functions**
  - EI, expected improvement
  - UCB / LCB, and more …
  - Designed to balance exploration (high variance) and exploitation (high mean)

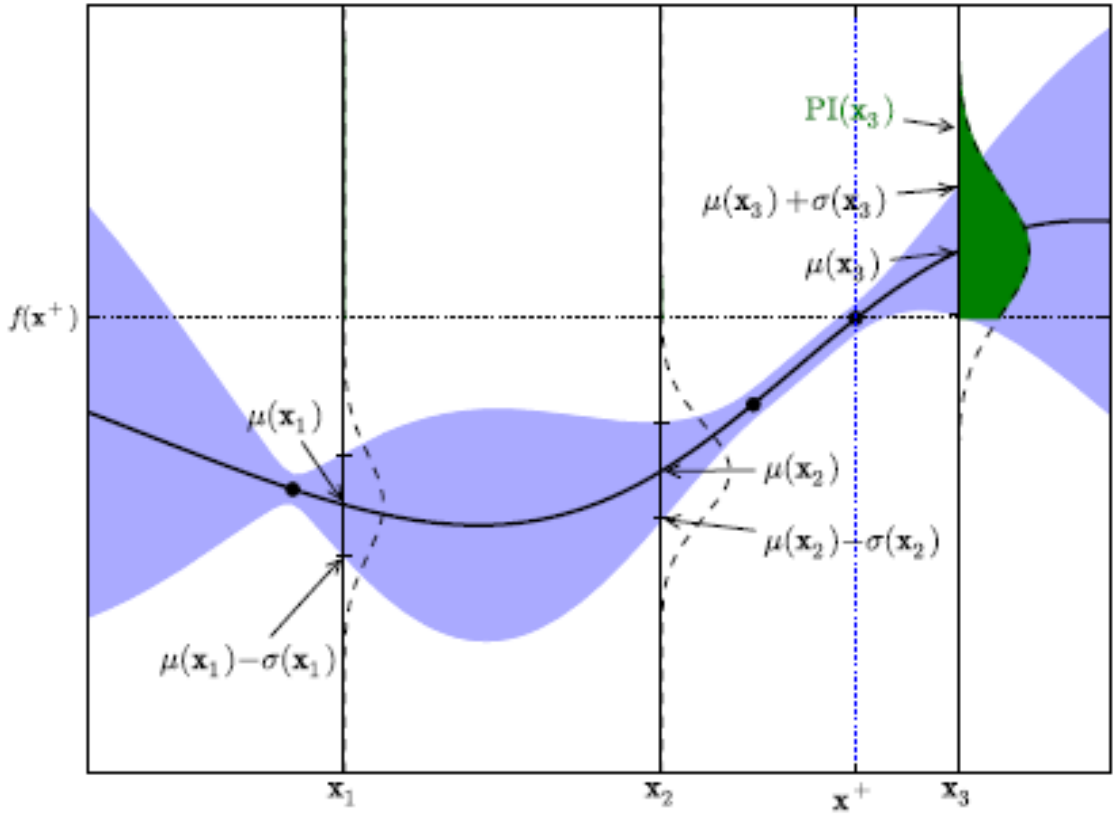
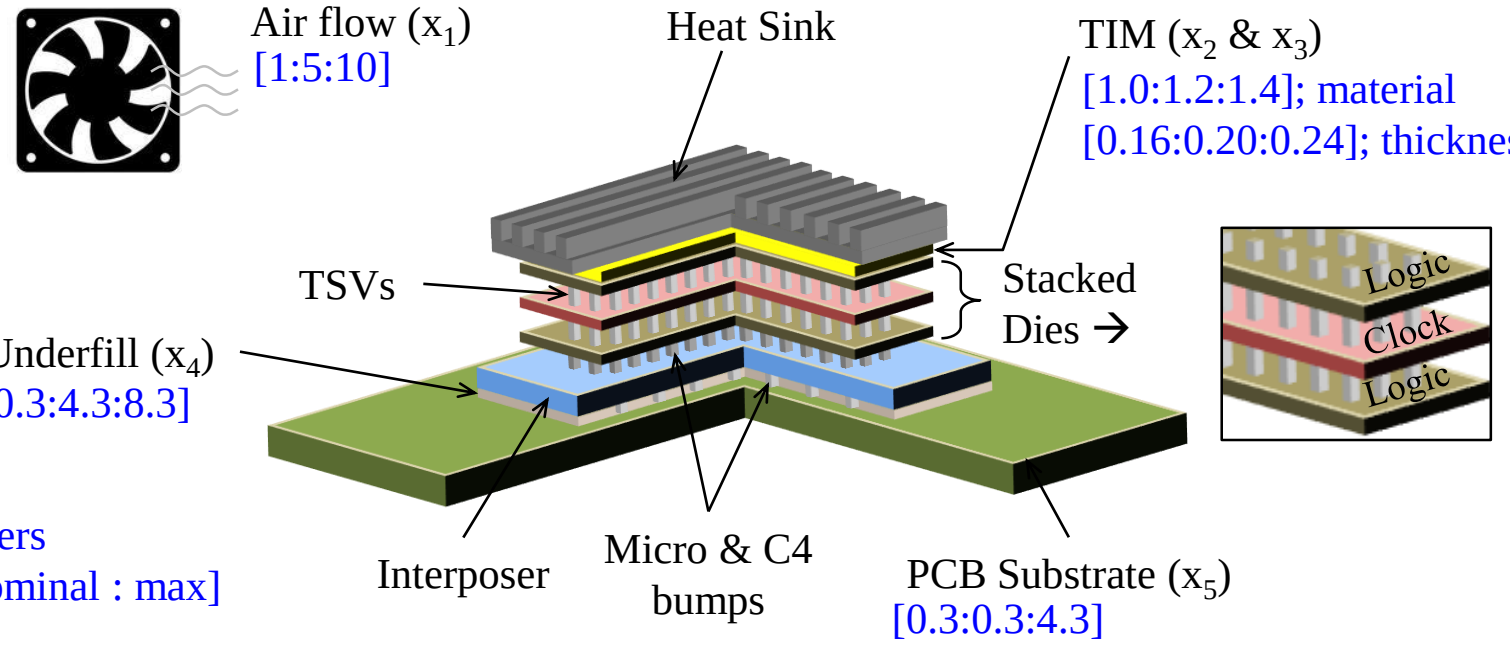


Figure: E. Brochu. arXiv:1012.2599

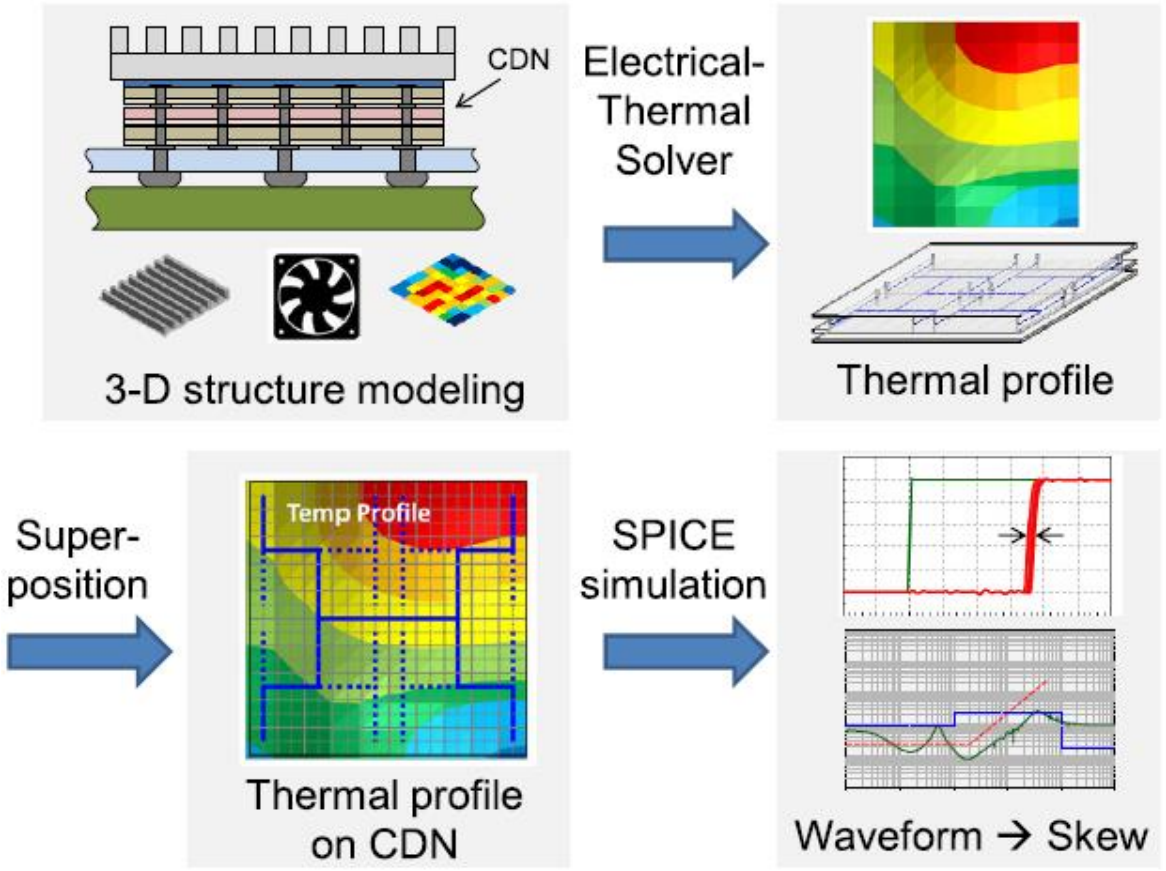
Thermal Design Optimization for 3D-IC



S. J. Park, B. Bae, J. Kim and M. Swaminathan, "Application of Machine-Learning for Optimization of 3-D Integrated Circuits and Systems," *IEEE Trans. VLSI,* June 2017.

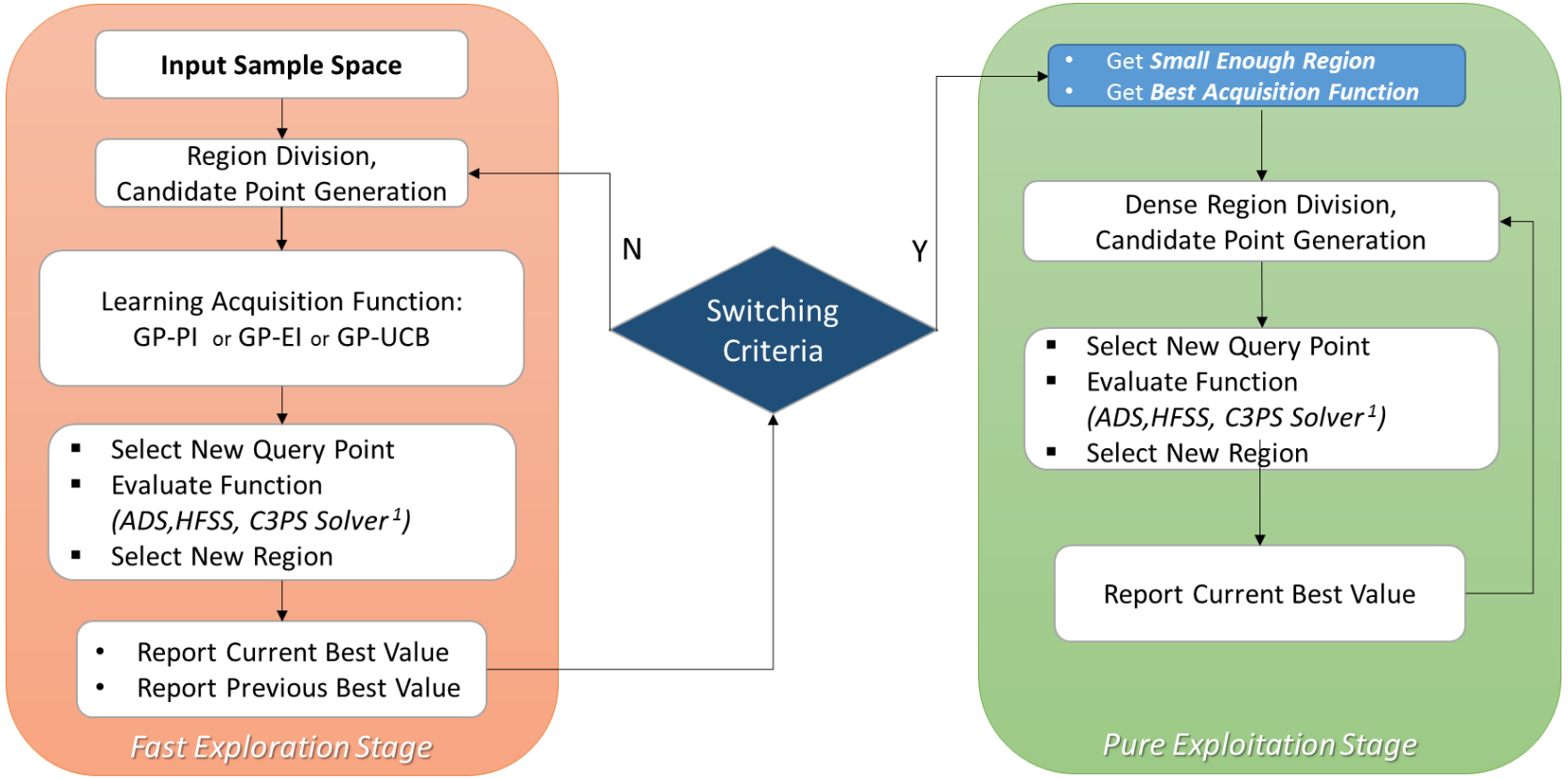
- Clock Skew is affected by Temperature (magnitude and gradient)
- Temperature is controlled by FIVE input (or control) parameters
- Control parameters have certain constraints
- Objective is therefore to TUNE these parameters to minimize Skew
- More generally, seek to find $X_{opt} = \underbrace{argmin}_{x}((f(X))$. Accurate modeling of f(X) needed only near min
- Use ML-based Bayesian Optimization

Full Design Space Exploration too Costly



- 3D (finite volume) simulations + SPICE-type circuit simulation
- Need to limit the number of designs that are simulated

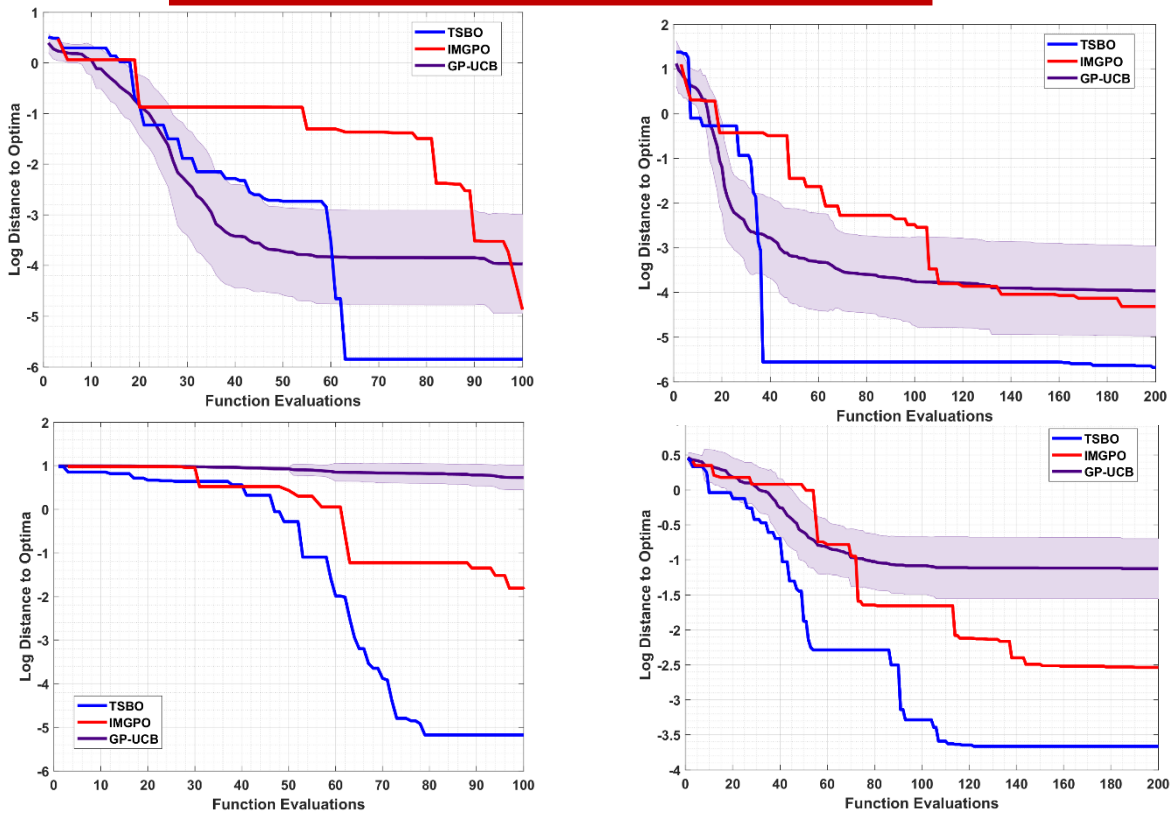
Two-stage Bayesian Optimization



- **First, Fast Exploration; Second, Pure Exploitation**
  - Coarse and fine tuning
- **Hierarchical Partitioning Scheme**
  - Reduces number of simulations required
- **Active learning of acquisition function**

Not the Fastest Algorithm, but …

Fewer iterations to convergence



Note: IMGPO is a known algorithm that is publicly downloadable

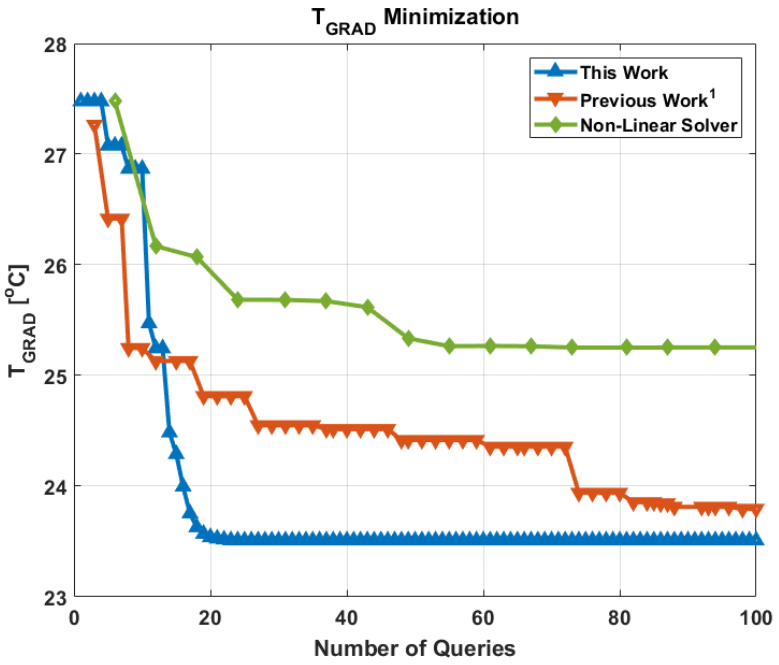
Algorithm Run Times for 100 iterations

	2D BRANIN	3D HARTMANN	4D SHEKEL	6D HARTMANN
TSBO	6.15s	4.35s	6.79s	6.51s
IMGPO	1.28s	1.35s	1.83s	1.91s
GP-UCB	10.4s	10.5s	11.4s	12.2s

- With the cost of few seconds of algorithm run time, TSBO reduces number of simulations required for accurate optimization

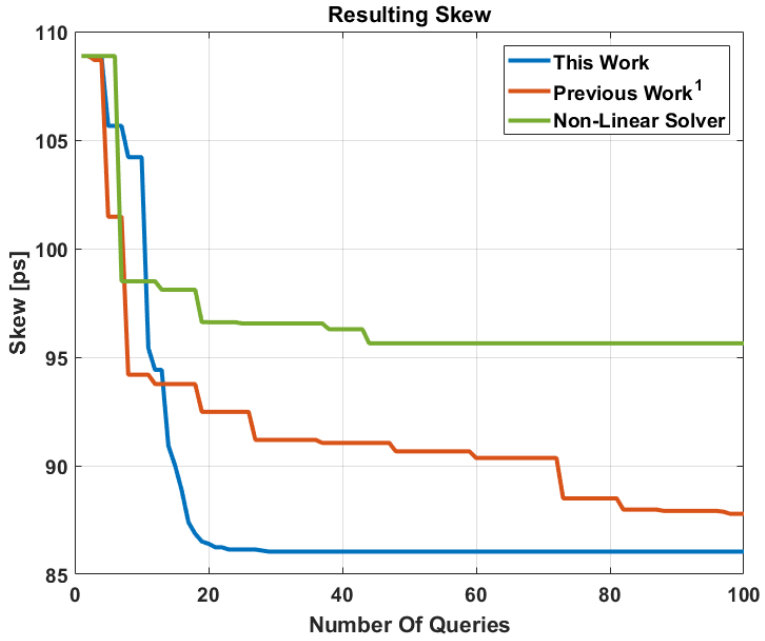
- **For EDA applications in which simulations are expensive, minimizing the number of physical simulations is the goal**

Results: Clock Skew Minimization



~4X Faster

[1] S. J. Park et al., TVLSI '17.



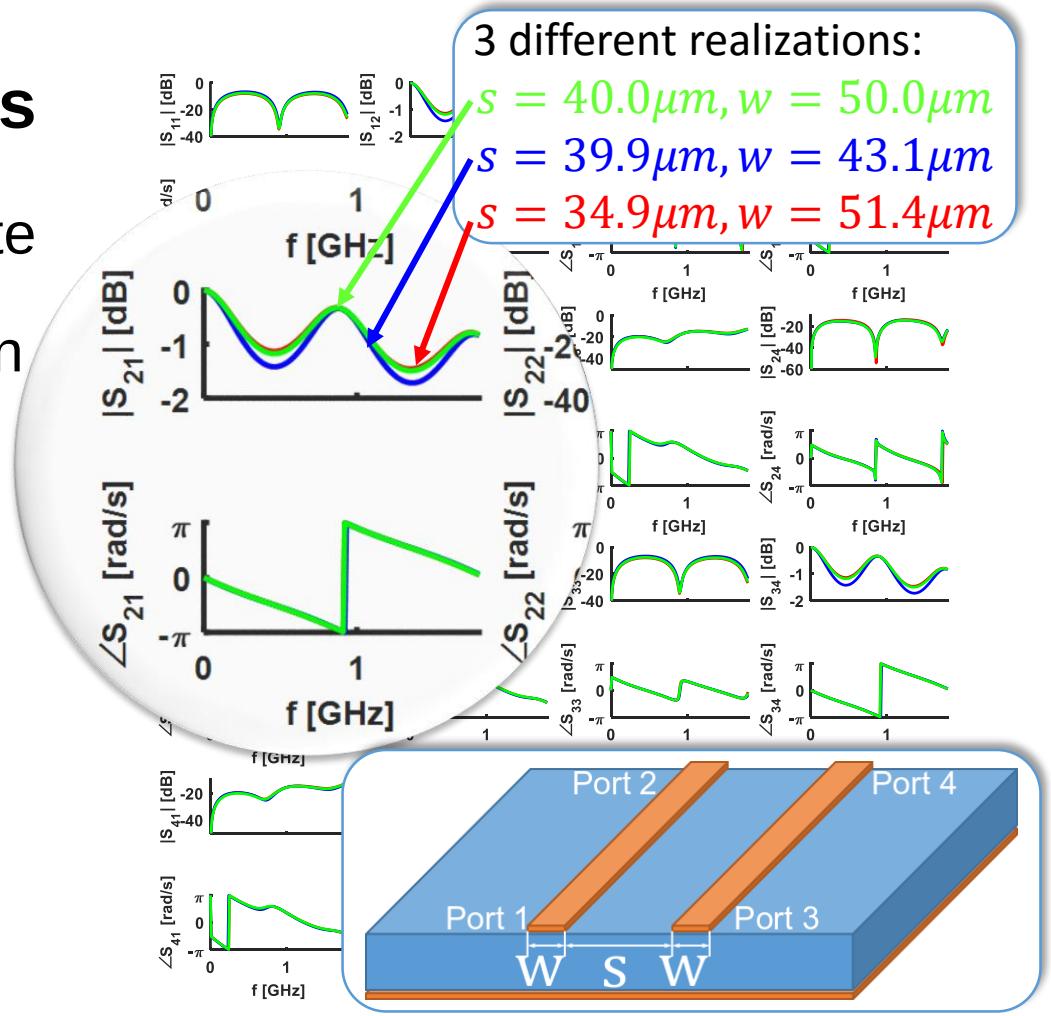
	Non Linear Solver	Previous Work[1]	This Work
T_{GRAD} [°C]	25.2 (+%9.4)	23.8 (+%4.7)	**23.5**
Skew [ps]	96.6 (+%12.3)	88 .0 (+%2.3)	**86.0**
CPU Time (Normalized) *	3.96	3.76	**1.00**

CAEML Activities

- **Design Optimization**
  - Example: Clock skew minimization for 3D-IC
- **Generative Modeling**
  - Example: Stochastic models of PCB interconnects
- **Circuit macro-modeling**
  - Example: Recurrent neural network
  - Example: System identification
- **Other …**
  - Deep networks (for DRC)
  - Causal inference (for hardware failure models)
  - Security threat detection

Interconnect Modeling

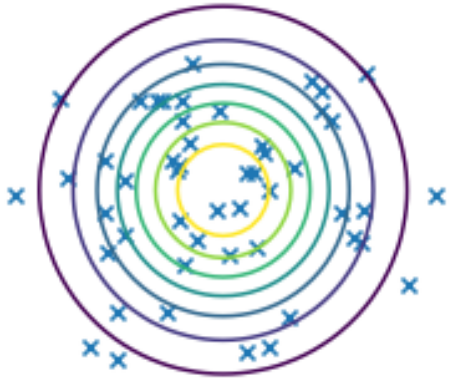
- **Scattering parameters characterize the electrical behavior of circuit interconnects**
  - Complex, matrix-valued function of frequency
  - Stored as frequency-tabulated matrices at discrete frequency points within band of interest
  - Can be used as generic multiport network block in circuit simulation
  - Dense frequency sampling and multiport interconnects produce high-dimensional data
- **S-parameters of manufactured interconnects exhibit variability**
  - Manufacturing process introduces variability in material and geometric properties
  - Deviation of those properties from design value alters electrical behavior
  - Probabilistic model is necessary to capture the variability



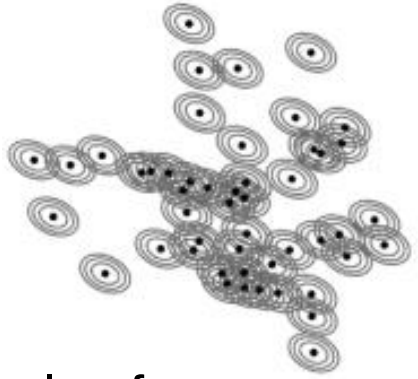
Overlaid scattering parameters

Generative Models

- **Goal: Model probability distribution of S-parameters in a form that lends itself to sampling**
  - Generate S-parameters for stochastic circuit simulations
- **Current state-of-the-art**
  - S. Ridder et al., "A generative modeling framework for statistical link analysis based on sparse data," *IEEE Trans. CPMT* (2017)
    - Vector fitting (VF) and principal component analysis (PCA) based dimensionality reduction; kernel density estimation (KDE) based non-parametric statistical model
  - S. Ridder et al., "Generation of stochastic interconnect responses via Gaussian process latent variable models" *IEEE Trans. EMC* (2018)
    - VF and latent variable model (LVM) based dimensionality reduction; Gaussian process (GP) regression based non-parametric statistical model
- **Drawback**
  - Inefficient for large amount of high-dimensional data
    - Need additional dimensionality reduction steps, which may compromise model accuracy



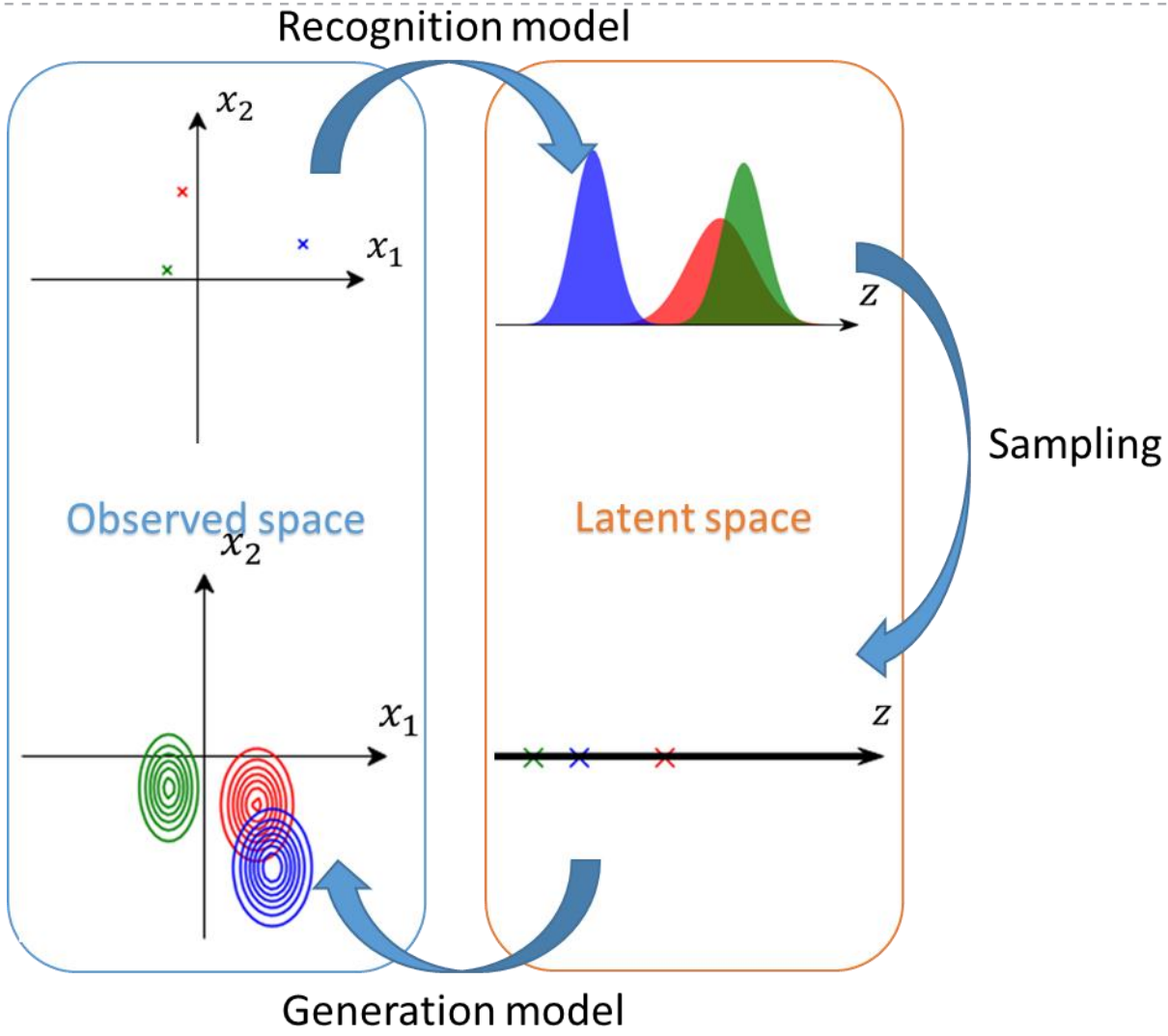
Example of parametric statistical model: Gaussian fitting



Example of non-parametric statistical model: KDE

Variational Autoencoder (VAE)

- **Explain variation of observed data with latent variables and non-linear mapping**
  - High-dimensional data often concentrates near low-dimensional manifold
  - Latent space is low-dimensional and easy to model
- **Recognition model and generation model**
  - Probability distributions parameterized by outputs of neural network
- **Optimize variational bound on combined model**
  - Identify the most appropriate latent space
  - Maximize reconstruction accuracy
- **Advantages**
  - High memory efficiency
    - Parametric statistical model offers bounded model complexity
    - Iterative training algorithm
  - Improved accuracy
    - No additional dimensionality-reduction step needed



Physical Consistency

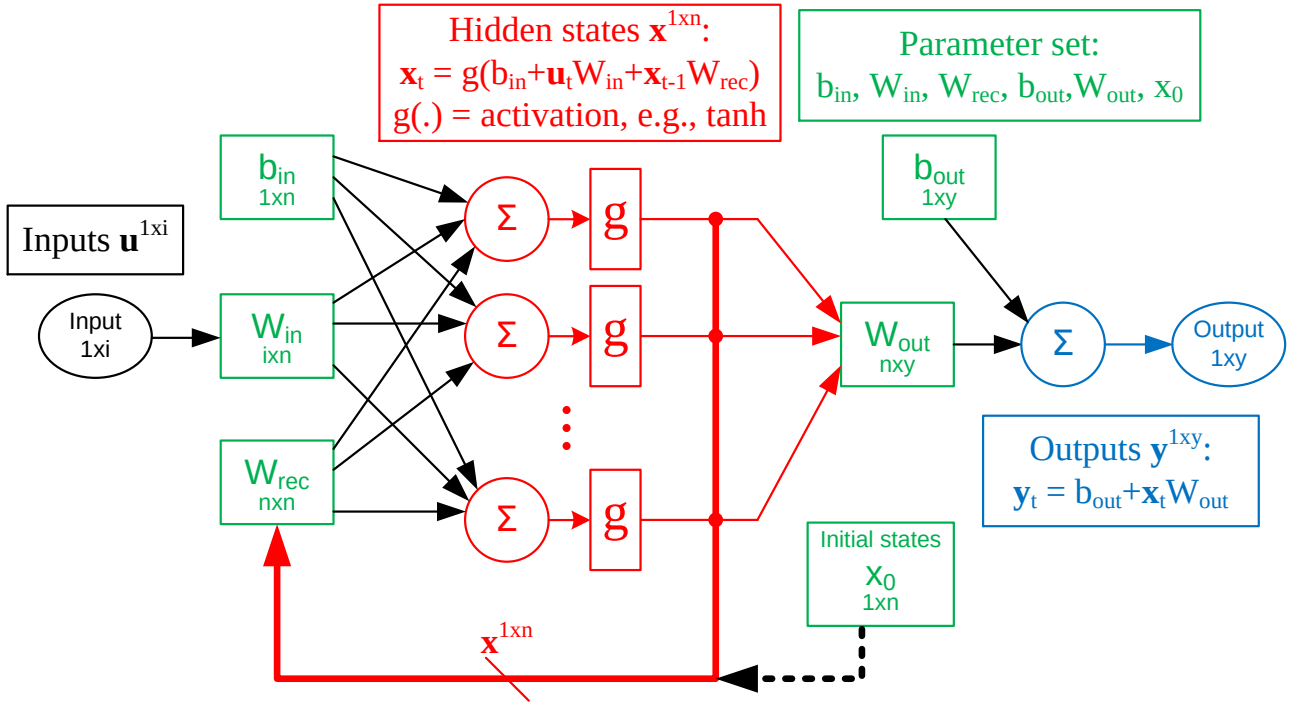
- **Stability, causality, and passivity**
  - Fundamental properties that interconnect model must satisfy
  - Violations potentially cause erroneous circuit simulation results
  - Difficult to build these constraints into the generative model
- **Our solution: Vector fitting (VF) of generated scattering parameters (post-processing step)**
  - Producing a rational function fit of system response (macro-model) based on frequency-tabulated data
  - Straightforward enforcement of stability and causality as well as perturbative passivity enforcement result in a physically consistent macro-model
  - Matlab has VF functionality and many circuit simulators already have VF functionality built-in

X. Ma, *2018 SPI*

CAEML Activities

- **Design Optimization**
  - Example: Clock skew minimization for 3D-IC
- **Generative Modeling**
  - Example: Stochastic models of PCB interconnects
- **Circuit macro-modeling**
  - Example: Recurrent neural network
  - Example: System identification
- **Other …**
  - Deep networks (for DRC)
  - Causal inference (for hardware failure models)
  - Security threat detection

Transient Modeling Using RNN



- RNN approximate any system represented by a state space model

$$\begin{cases} \dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{y} = g(\boldsymbol{x}, \boldsymbol{u}) \end{cases}; \; u \text{ is input, } y \text{ is output, } x \text{ is state}$$

- Many circuits and devices can be represented by state space models

Modified RNN for Zero-in Zero-out

Z. Chen et al., *2017 EPEPS*

- **"Vanilla" RNN equations**

$$\boldsymbol{x}_i = \tanh\left[\boldsymbol{b}_u + W_r \boldsymbol{x}_{i-1} + W_u \boldsymbol{u}_i\right]$$
$$\boldsymbol{y}_i = \boldsymbol{b}_y + W_y \boldsymbol{x}_i$$

- **Electrical circuits at thermal equilibrium**
  - Response to zero input and zero initial state is zero
    - Relevant when simulating response of an unpowered circuit to system-level ESD
- **Implementation of zero-in zero-out (ZIZO) RNN**

$$\boldsymbol{x}_i = \tanh\left[W_r \boldsymbol{x}_{i-1} + W_u \boldsymbol{u}_i + \boldsymbol{\Omega_0}\right] - \tanh \boldsymbol{\Omega_0}$$
$$\boldsymbol{y}_i = W_y \boldsymbol{x}_i$$

  - If $\boldsymbol{u} = \boldsymbol{0}$ (zero input) and $\boldsymbol{x}_0 = \boldsymbol{0}$ (zero initial state), then $\boldsymbol{y} = \boldsymbol{0}$ (zero output)

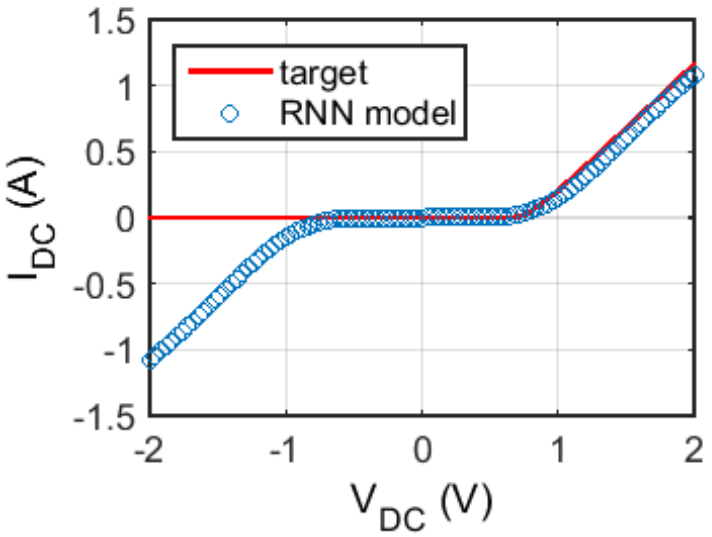
Discussion: ZIZO Equations

- **Alternatively, ZIZO can be obtained using the vanilla RNN equations with the bias terms set to zero**

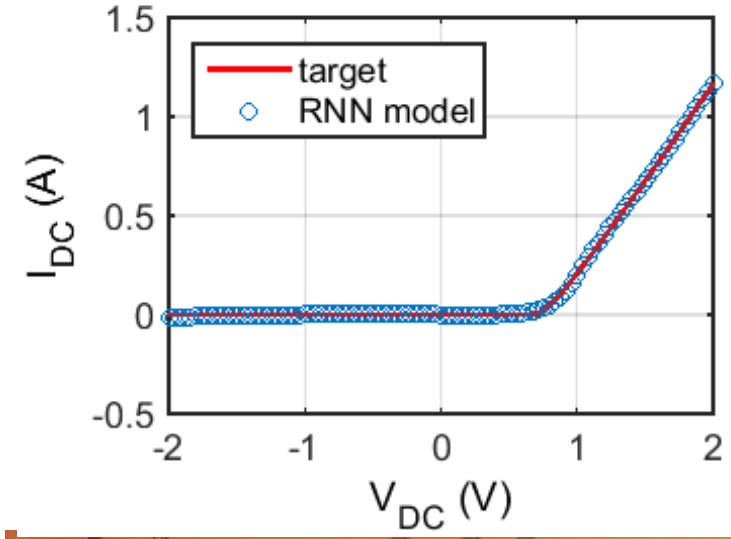
$$\begin{aligned} \boldsymbol{x}_i &= \tanh\left[\boldsymbol{b}_u + W_r \boldsymbol{x}_{i-1} + W_u \boldsymbol{u}_i\right] \\ \boldsymbol{y}_i &= \boldsymbol{b}_y + W_y \boldsymbol{x}_i \end{aligned} \longrightarrow \begin{aligned} \boldsymbol{x}_i &= \tanh\left[W_r \boldsymbol{x}_{i-1} + W_u \boldsymbol{u}_i\right] \\ \boldsymbol{y}_i &= W_y \boldsymbol{x}_i \end{aligned}$$

- **Issue: This RNN provides a good fit only to circuits whose I-V characteristic is an odd function**

Modified vanilla RNN



New ZIZO RNN



RNN for Circuit Simulation

- **Transform discrete time RNN model to continuous time**
- **Hidden states x as node voltages**
- **Implement differential equation in Verilog-A**
  - Differential Equation:

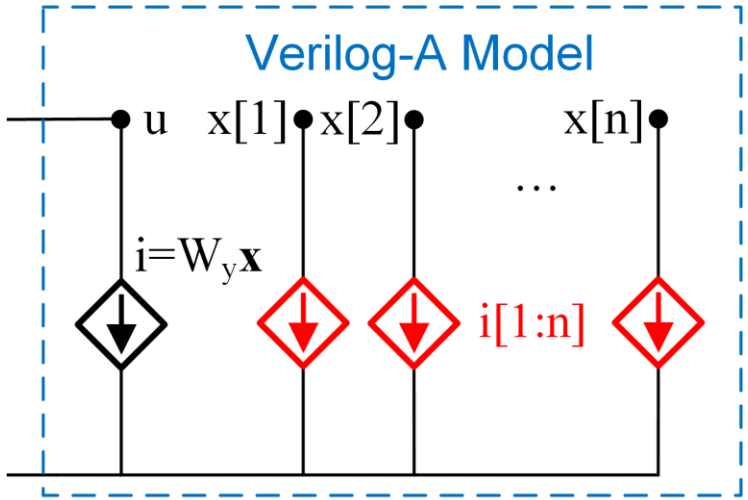
$$\underbrace{\boldsymbol{x} + \beta h \dot{\boldsymbol{x}}}_{LHS(x)} = \underbrace{\tanh\left[W_r(\boldsymbol{x} - \alpha h \dot{\boldsymbol{x}}) + W_u(\boldsymbol{u} + \beta h \dot{\boldsymbol{u}}) + \boldsymbol{\Omega_0}\right] - \tanh \boldsymbol{\Omega_0}}_{RHS(x,u)}$$

    - Create current sources

$$i[1:n] = LHS(x) - RHS(x,u)$$

    - Due to KCL, simulator solves for

$$i[1:n] = 0$$



Z. Chen et al., *2017 EPEPS*

Model Stability

- **The behavioral model should preserve the stability of the circuit being represented**
- **For arbitrary steady-state input u_s and any initial condition, the solution of the differential equation must satisfy**

$$\lim_{t\to\infty} \boldsymbol{x}(t) = \boldsymbol{x_s},$$

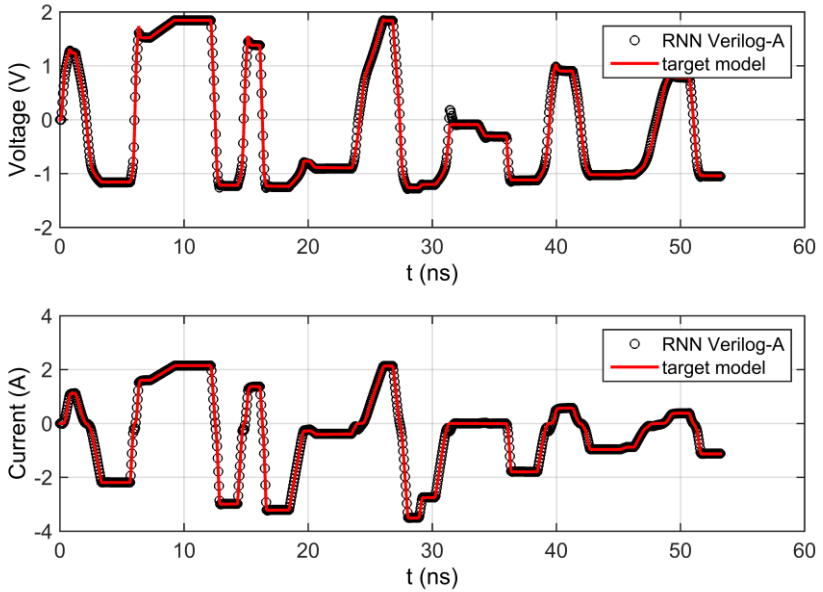
where x_s is the solution of the static equation

$$\boldsymbol{x_s} = \boldsymbol{\sigma}[W_r \boldsymbol{x_s} + W_u \boldsymbol{u_s} + \boldsymbol{\Omega_0}] - \boldsymbol{\sigma}(\boldsymbol{\Omega_0})$$

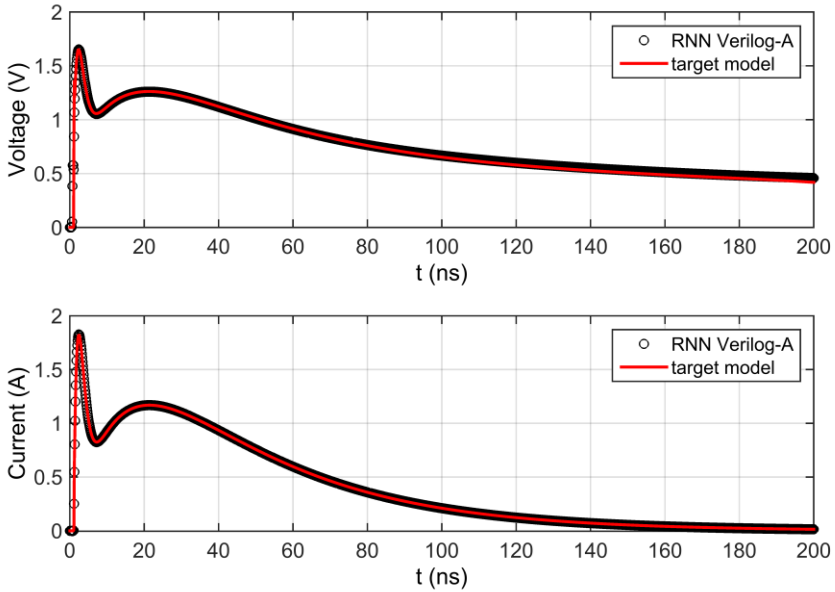
- **RNN weights are optimized using stochastic gradient descent → introduce a term into the loss function to penalize unstable models**
  - Penalize positive eigenvalues of $T_0 = W_r + W_r^T - 2I$

Verilog-A Model Evaluation

- **Apply previously unencountered stimuli to Verilog-A RNN model of an Active Rail Clamp circuit**



Randomly generated piecewise-linear waveform

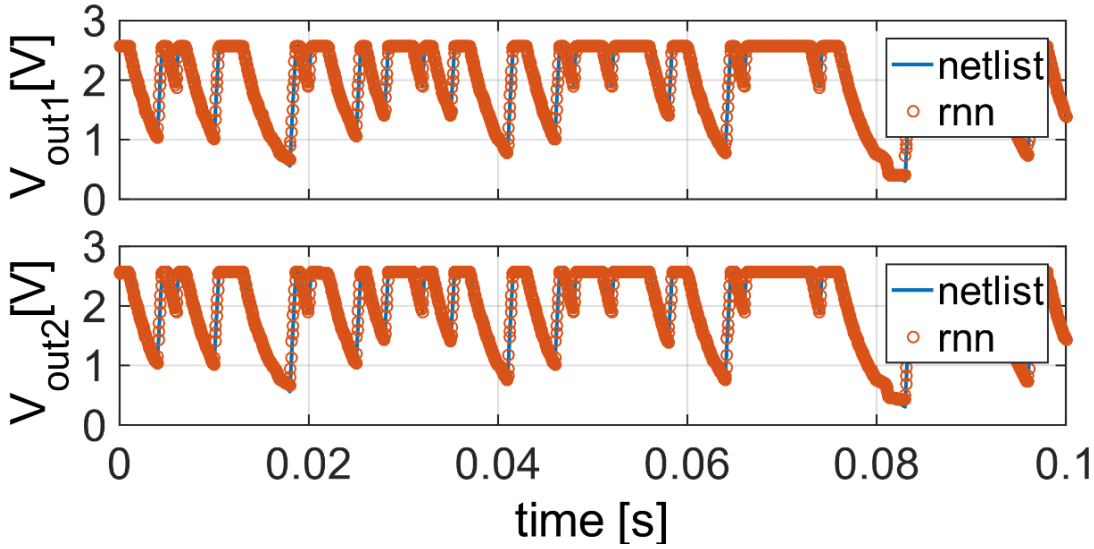


IEC 61000-4-2 system-level ESD waveform

- **Average error less than 2% of peak-to-peak amplitude in all cases**

Computational Efficiency

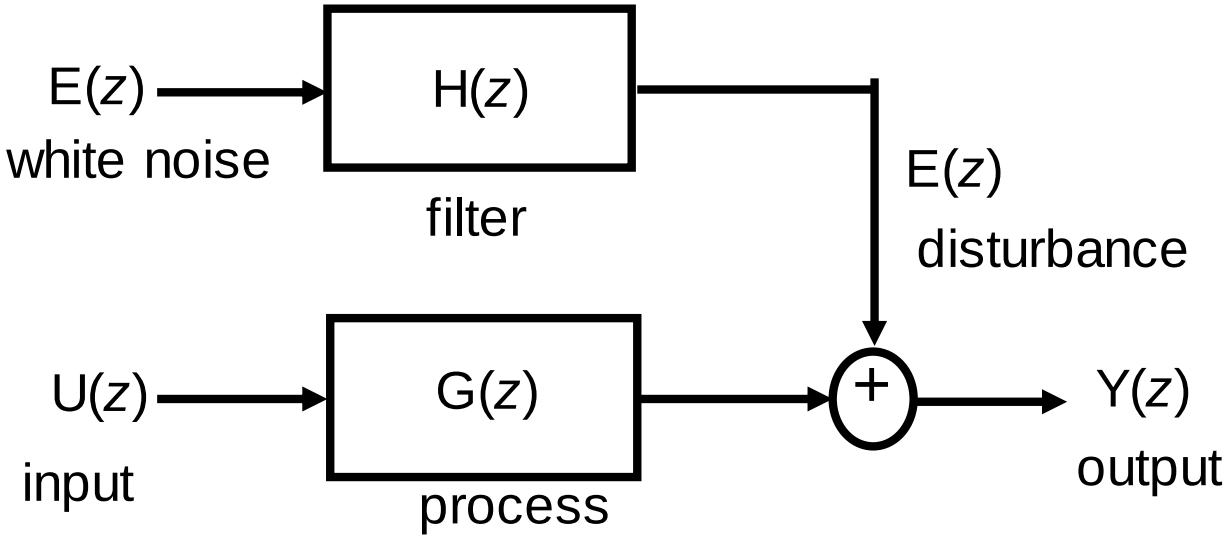
- **Test case: Encrypted netlist for unknown circuit**
  - Over 1000 transistors
  - 7 voltage inputs, 2 voltage outputs
  - Stimuli: 2-level digital input with V_H=5V, V_L=0V, T_{bit}=1ms, t_r=t_f=1ns
- **Training data generation**
  - Randomly assign 1 or 0 to each input for each bit (1 ms)
  - Transient circuit simulation with full netlist
- **Result**
  - Basic RNN model
  - Waveform shown at right
  - Model Error < 1%
  - Simulation Time
    - **Netlist 254.5 s**
    - **RNN 5.7 s**



System Identification

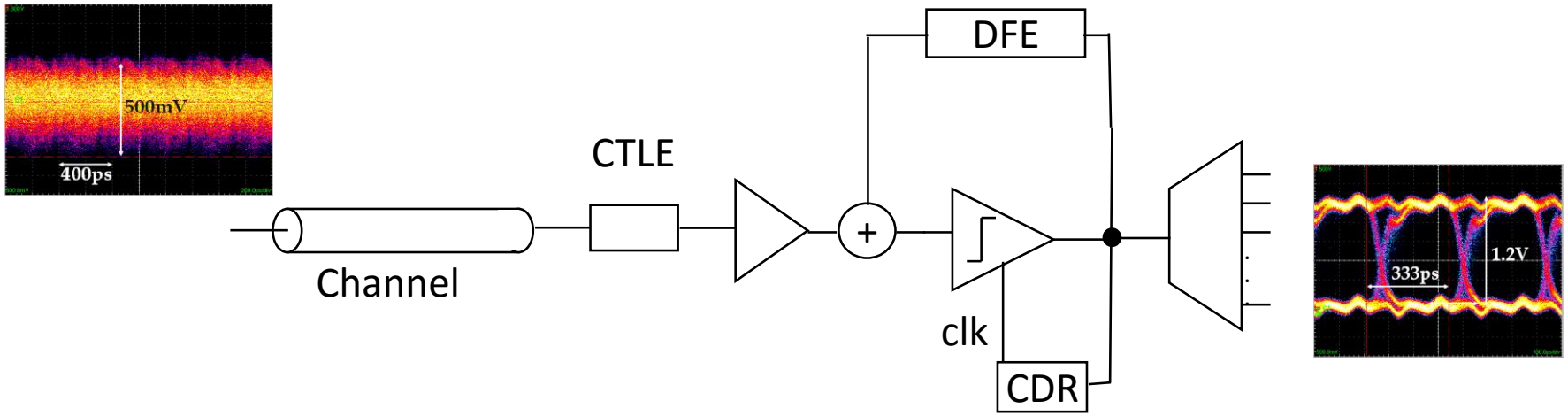
System identification is used to build models for dynamic systems based on the input and output pairs. It solves the black-box problem.

Two general approaches: linear system identification modeling and nonlinear system identification modeling.



Example: Receiver Modeling

- **Problem:**
  - How to predict eye opening at output of receiver when all you can measure the closed eye at the input
  - How to determine best settings for DFE, CTLE
- **Solution: Black box modeling using System Identification**
  - Matlab toolbox



AutoRegressive eXternal input (ARX)

- **In ARX model, the current output is related to previous delayed inputs and the previous outputs. It doesn't consider white-noise disturbance. The ARX linear model predictor is of the form:**

$$A(q)y(t) = B(q)u(t - n_k)$$

Where $A(q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}$

$B(q) = b_1 + b_2 q^{-1} + \cdots + b_{n_b} q^{-n_b+1}$

Usage in Matlab:

>> sys = arx(data,[na nb nc nk])

- data: training data
- na: is the order of the polynomial A(q)
- nb: is the order of the polynomial B(q) + 1
- nk: is the input-output delay expressed as fixed leading zeros of the B polynomial

ARX Model

m = arx(ztrain,[2 2 0]);

compare(ztest, m);

Description:

ztrain: training data

m: arx model

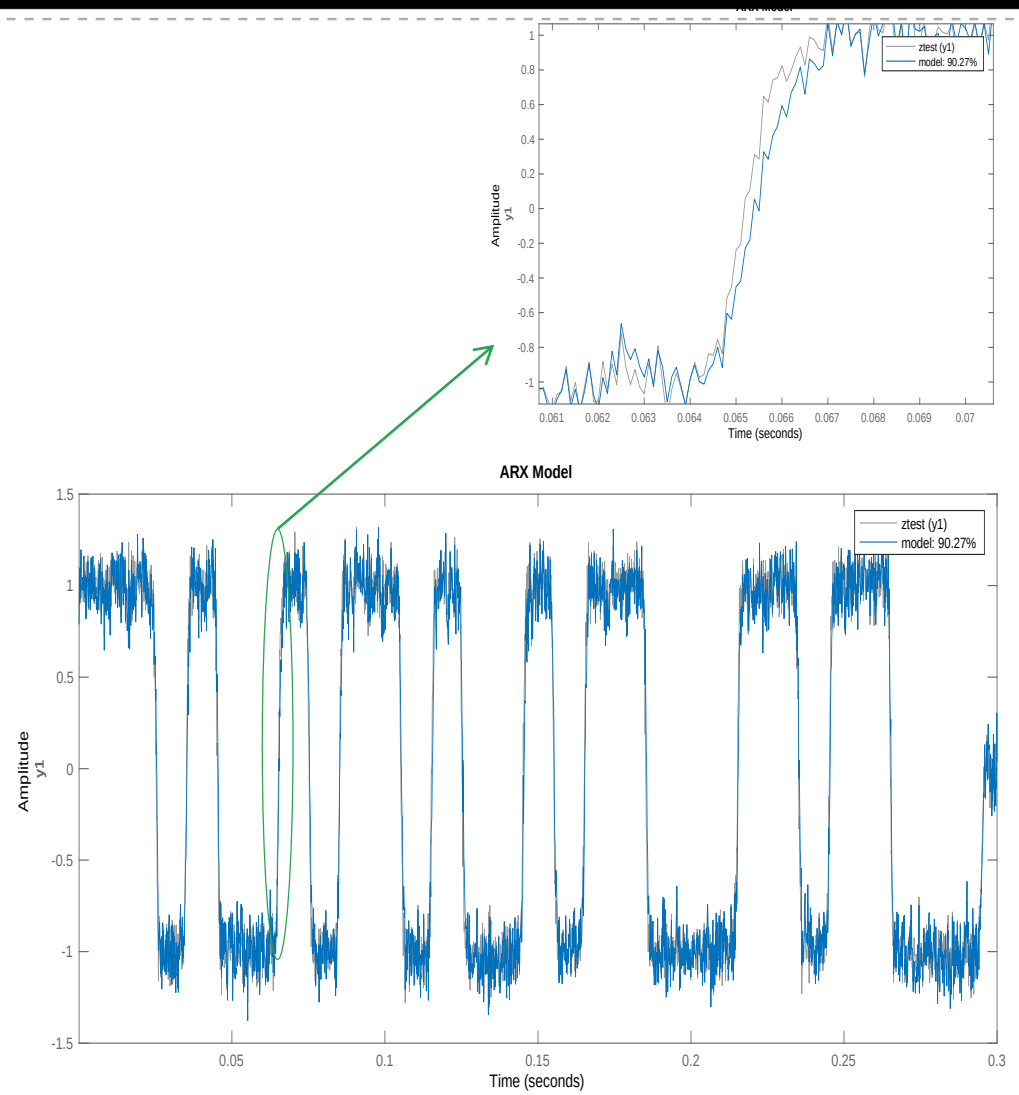
Order of A (how many previous outputs): 2

Order of B (how many previous inputs): 2

Delay (delay between input and output): 0

ztest: testing data

compare(): test model accuracy in time domain.



AutoRegressive Moving Average eXternal input(ARMAX)

- **In ARMAX model, the current output is related to previous delayed inputs, the previous outputs, current and previous white-noise disturbance values. The ARMAX model is more general than the ARX model. The ARMAX linear model structure is shown below:**

$$A(q)y(t) = B(q)u(t - n_k) + C(q)e(t)$$

Where $A(q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}$

$B(q) = b_1 + b_2 q^{-1} + \cdots + b_{n_b} q^{-n_b+1}$

$C(q) = 1 + c_1 q^{-1} + \cdots + c_{n_c} q^{-n_c}$

- **Usage in Matlab:**

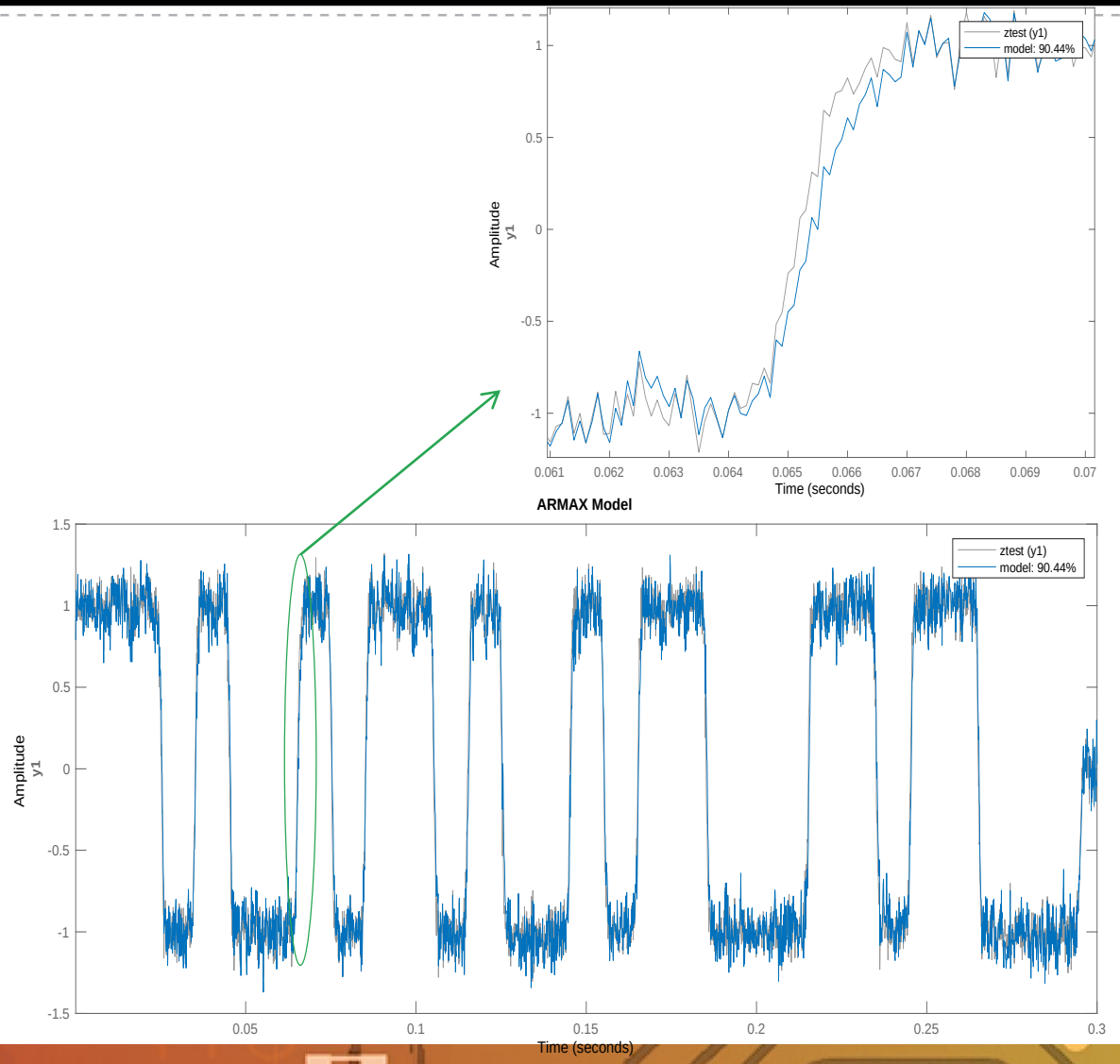
  >> sys = armax(data,[na nb nc nk])

  - data is the training data; na is the order of the polynomial A(q), nb is the order of the polynomial B(q) + 1; nc: is the order of the polynomial C(q); nk: is the input-output delay expressed as fixed leading zeros of the B polynomial

ARMAX Model

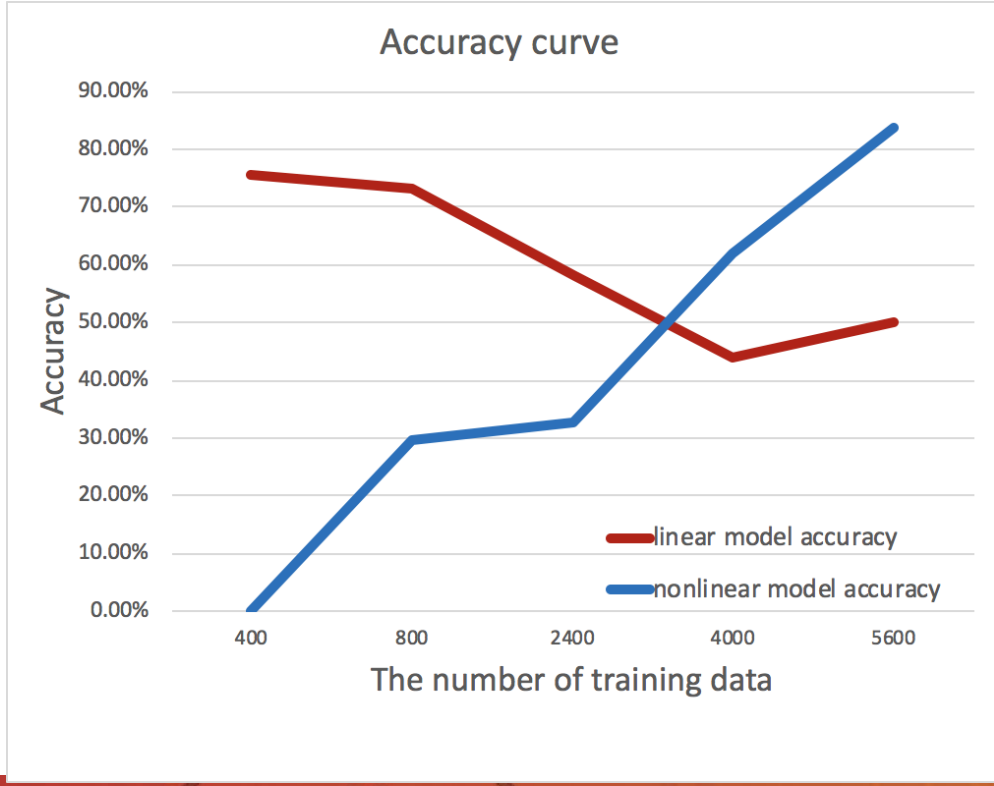
>> m = armax(ztrain,[2 3 3 0]);
>> compare(ztest, m);

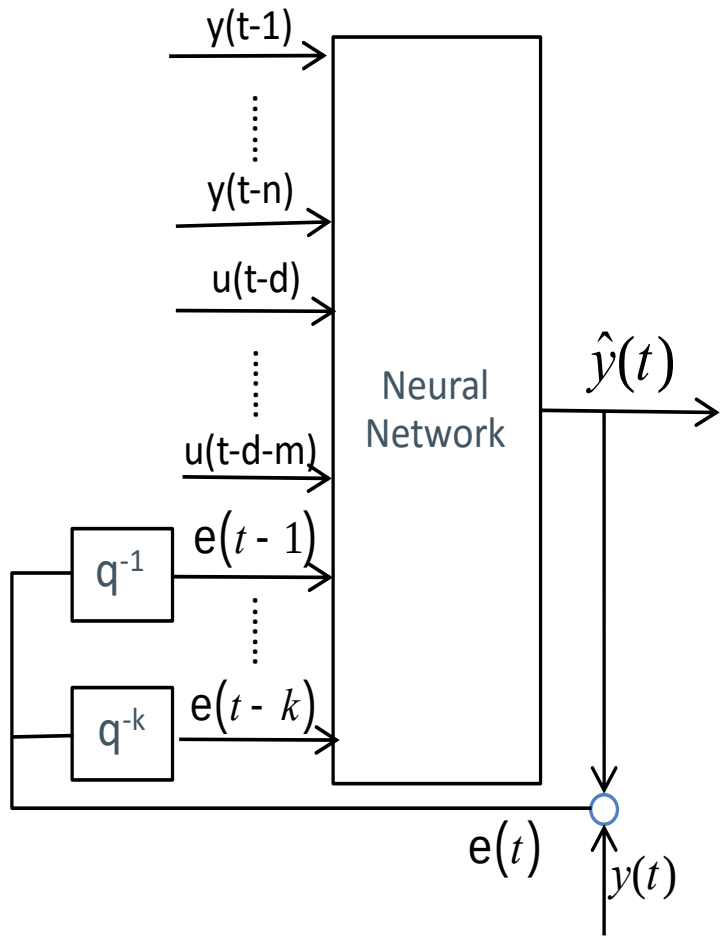
Description:
ztrain: training data
m: armax model
Order of A (how many previous outputs): 2
Order of B (how many previous inputs): 3
Order of C (previous white-noise disturbance value): 3
Delay (delay between input and output): 0
ztest: testing data
compare(): test model accuracy in time domain.



Solution Space

- **Nonlinear network autoregressive external moving average input gives best accuracy with a large enough training set**





Extra Slides

Industry-University Cooperative Research Center (I/UCRC) Program

Mission:

- To contribute to the nation's research infrastructure base by developing long-term partnerships among industry, academe and government
- To leverage NSF funds with industry funds to support graduate students performing industrially-relevant research

Vision:

- To expand the innovation capacity of our nation's competitive workforce through partnerships between industries and universities

40 years of fostering and growing long-term partnerships among industry and academe based on shared value

(Adapted from Hoffman, *NSF Assessment Coordinator Introduction*, April 25, 2017, CAEML IAB Meeting)

CAEML Industry Members for 2018

CAEML Research Projects (2018)

- **Modular Machine Learning for Behavioral Modeling of Microelectronic Circuits and Systems**
  - The modularity offered by the behavioral approach will be leveraged to develop mathematical tools for assessing the performance and minimal data requirements for learning a low-complexity representation of the system behavior, one component or subsystem at a time
- **Intellectual Property Reuse Through Machine Learning**
  - Recast an analog or full custom digital design from one technology node to another, assuming the same circuit topology
- **Behavioral Model Development for High-Speed Links**
  - Systematically develop a hierarchy of behavioral models of circuits that have the same accuracy as the transistor-level models, but require 25–50X less CPU time and memory.

Research Projects

- **Models to Enable System-level Electrostatic Discharge Analysis**
  - ML is used to create ESD models of the system's nonlinear components, as needed for SOA analysis and soft failure prediction. The models are targeted for circuit or hybrid (EM-circuit) simulators.
- **Optimization of Power Delivery Networks for Maximizing Signal Integrity**
  - Develop ML based software to optimize the system output response based on a large set of design factors (control parameters). Co-optimization of the signal path and power delivery network is necessary.
- **Design Rule Checking with Deep Networks**
  - Investigate the feasibility of training a deep convolutional network to perform Design Rule Checking (DRC)

Research Projects

- **Machine Learning for Trusted Platform Design**
  - Use ML to assess if an IoT system is under attack and develop countermeasures
- **Machine Learning to Predict Successful FPGA Compilation Strategy**
  - Produce FPGA compilation recipes that show high success rate and fast compilation time
- **Causal Inference for Early Detection of Hardware Failure**
  - Use time-series sensor data to detect wear-out of a hardware component, e.g. HDD or SSD in a storage array. Longitudinal causal inference techniques will omit redundant covariates or features that might be correlated with the failure but do not help in the prediction task
- **Applying Machine Learning to Back End IC Design**
  - Utilize surrogate modeling to guide tool setup for a specific design and specific design goals