Probabilistic Bug Localization for Analog/Mixed-Signal Circuits using Probabilistic Graphical Models

Sangho Youn[1] and Chenjie Gu[2]
[1]Seoul National University, South Korea
[2]Intel Strategic CAD Labs, Hillsboro, OR, USA

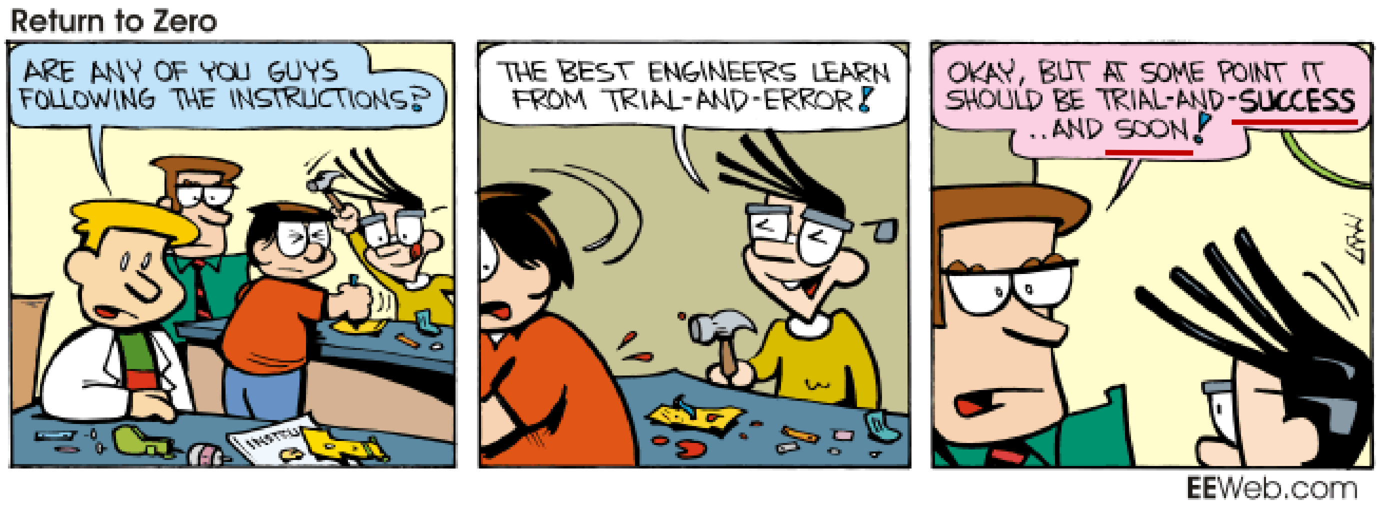
March 2014

Outline

- Overview
- Bug localization using graphical models
  - Graphical model creation
    - Gaussian Bayesian network
    - Table-based Bayesian network
  - Bug localization by statistical inference
- Experimental results
- Conclusion

Problem: Time-Consuming Debugging

- Debugging tasks are major bottlenecks in IC design
  - Mostly depends on trial-and-errors
  - Takes a significant amount of time!

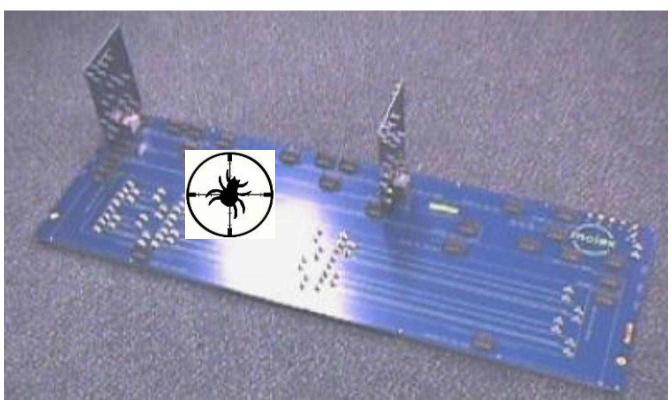


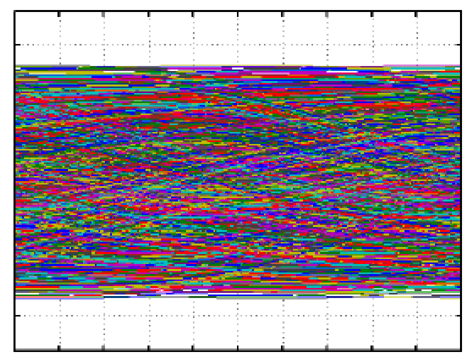
http://www.eeweb.com/rtz/trial-error

Goal: Automatic Bug Localization

- *Goal is to develop a tool that can automatically localize bugs from available waveforms and models, primarily for* ***post-silicon validation***

Which block caused this failure ?

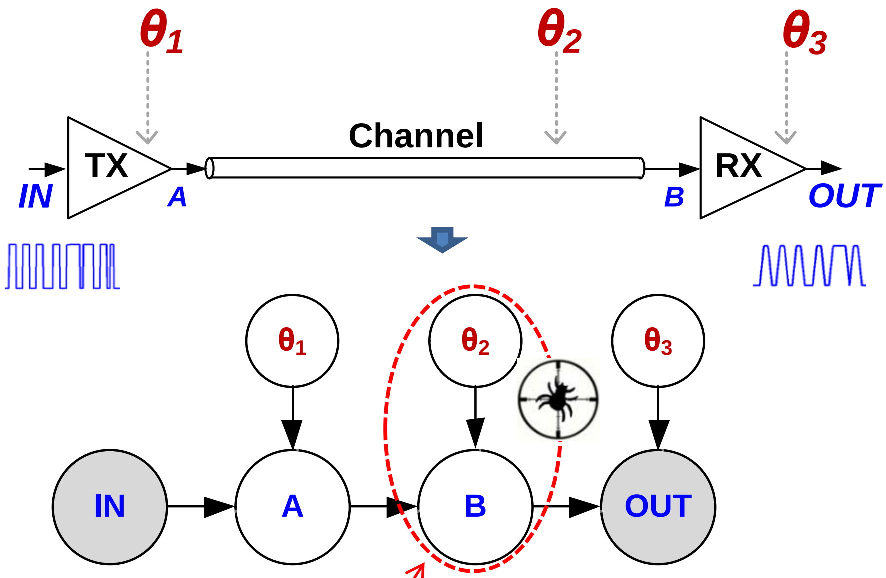


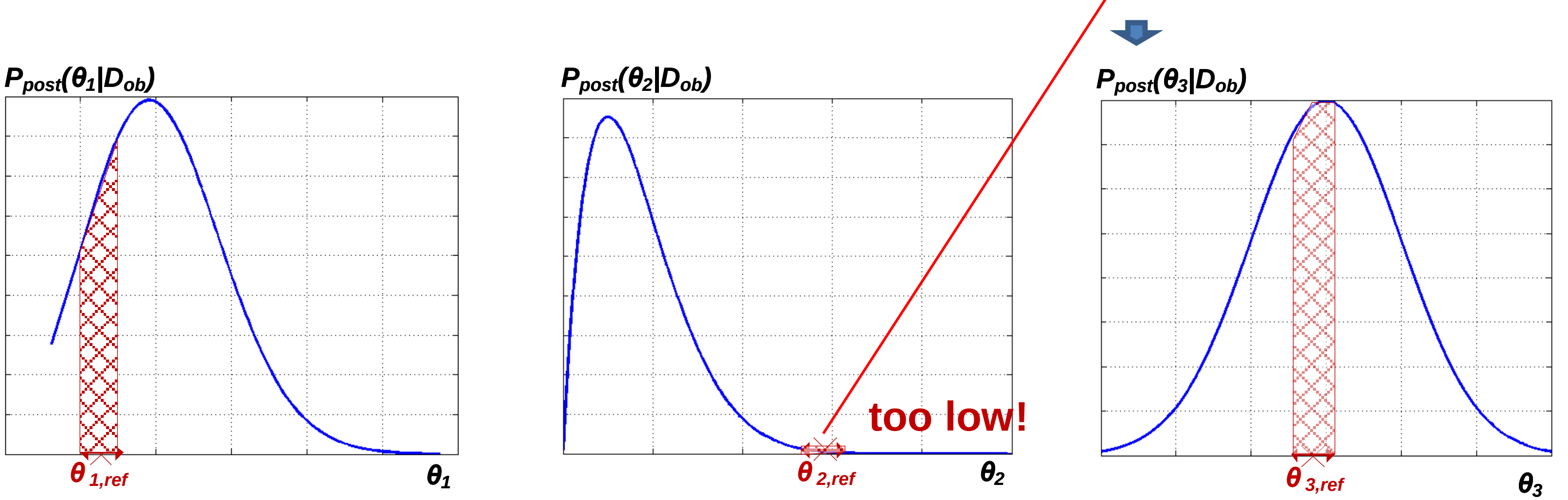


Closed eye, Failure!

Proposed Approach: Bug Diagnosis Using Probabilistic Graphical Models

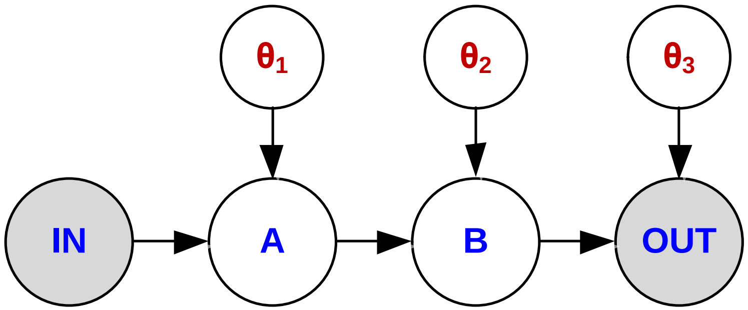
1. Construct probabilistic graphical model
2. Make an observation
3. Estimate the posterior probability of a system's parameter θ
4. If $P_{post}(\theta \text{ in } \theta_{spec\_range}) < threshold$, θ and its associated sub-block are reported as failure root-causes
   - If multiple bug root-causes are found, rank them according to $P(\theta=\theta_{ref}|D_{ob})$





Advantages of Our Approach

- Uncertainty/noise can be modeled
- Non-linearity can be modeled
- Efficient inference algorithms exist

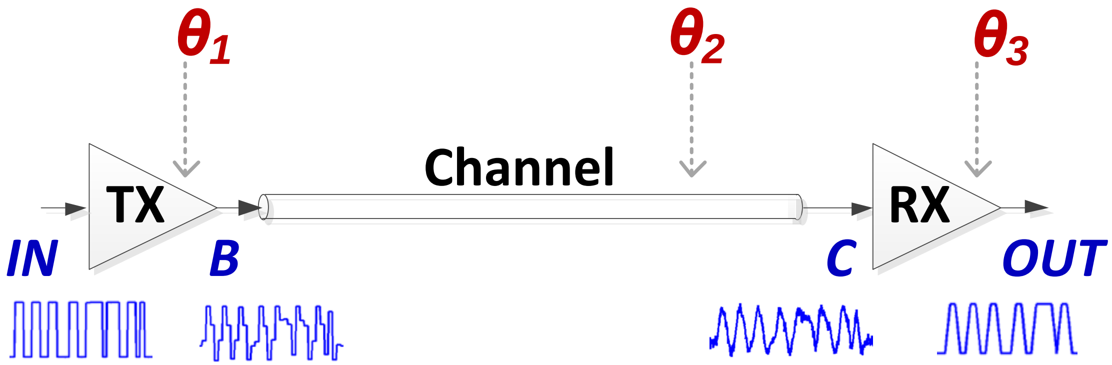


Outline

- Overview
- Bug localization using graphical models
  - Graphical model creation
    - Gaussian Bayesian network
    - Table-based Bayesian network
  - Bug localization by statistical inference
- Experimental results
- Conclusion

Probabilistic Models

- A system's behavior can be described by *probability* instead of a *functional* relationship

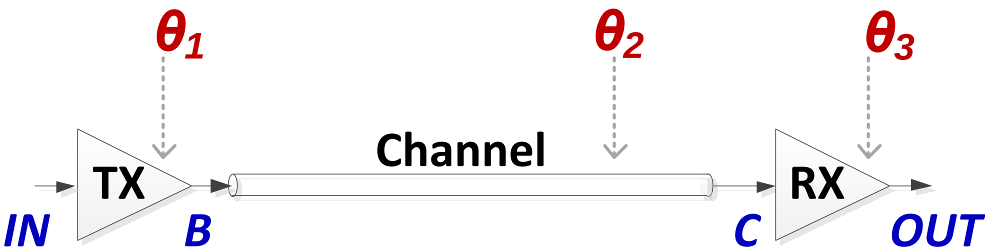


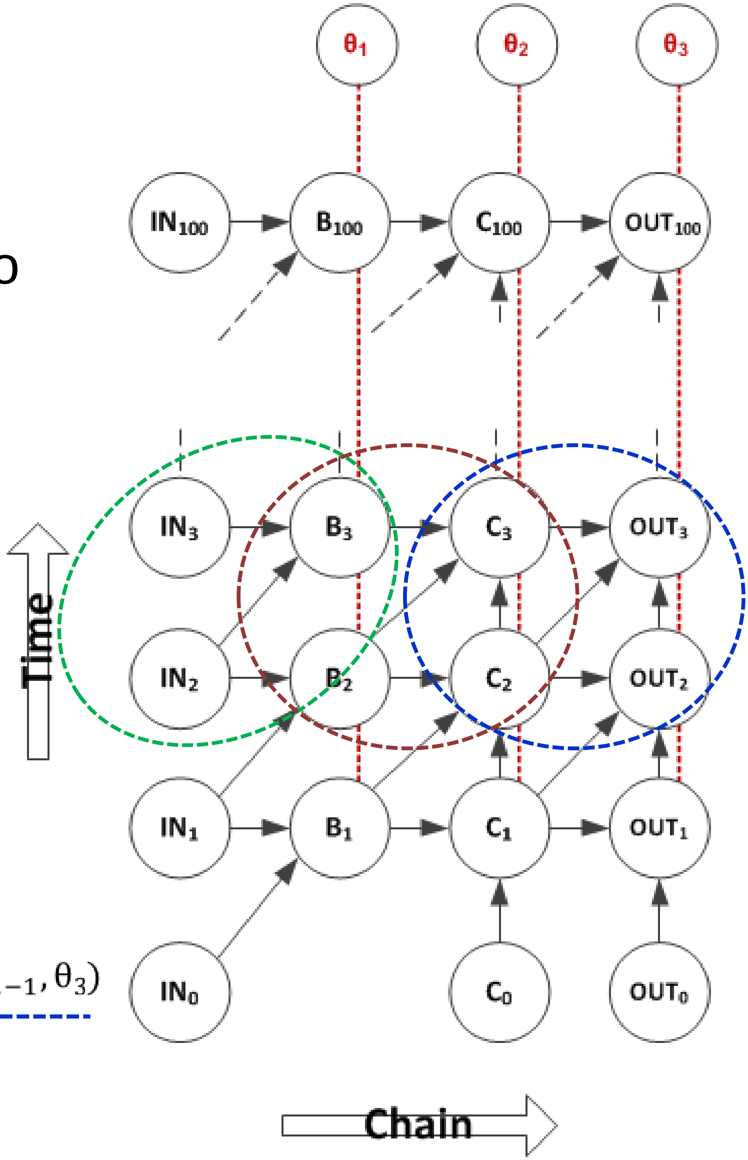
P(IN, B, C, OUT, θ_1, θ_2, θ_3)

This is difficult to characterize!

Probabilistic Graphical Model

- We can significantly reduce the complexity ***by graphical model***
  - Can decompose a full joint distribution into small factors





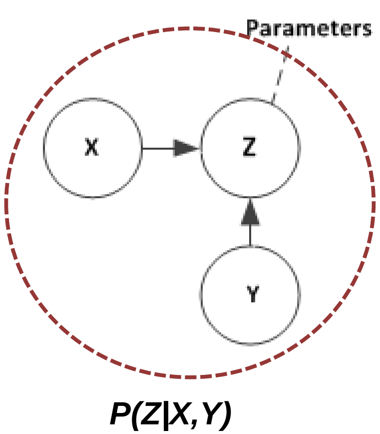
$$P(IN_{t:1}, B_{t:1}, C_{t:1}, OUT_{t:1}, \theta_1, \theta_2, \theta_3)$$

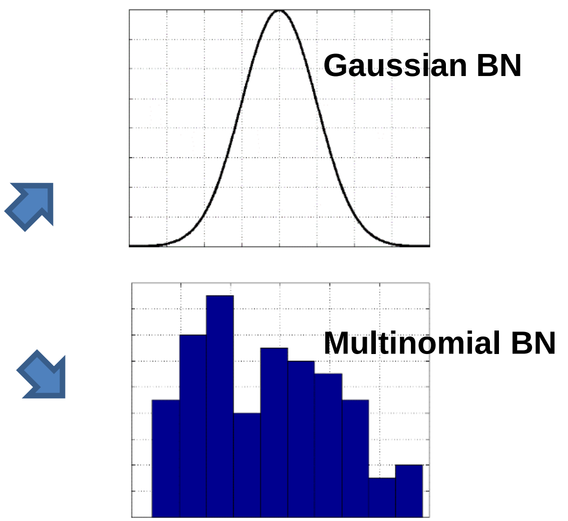
$$= \prod_{n=1}^{100} P(IN_n) P(B_n | IN_{n-1:n}, \theta_1)\ P(C_n | B_{n-1:n} C_{n-1}, \theta_2)\ P(OUT_n | C_{n-1:n} OUT_{n-1}, \theta_3)$$

Two Parametric Model of Factors in Graphical Model

- Conditional Probability Density (CPD)
  - A template to describe CPD, $\boldsymbol{P(Z_{out}|X_{in}, Y_{in})}$

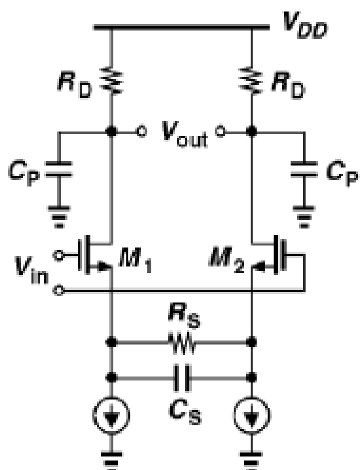
1. Gaussian Bayesian network (GBN)
   - $P(Z \mid X, Y) \sim Normal\ (aX+bY, \sigma^2)$
   - For ***linear*** block
2. Table-based Bayesian network (TBN)
   - $P(Z \mid X, Y) \sim Multinomial\ (p_1, p_2, \ldots, p_k)$
   - For ***nonlinear*** block





Gaussian Bayesian Network (GBN) Model Example – Continuous Time Linear Equalizer

- CTLE example



$$H(s) = \frac{g_m}{C_p} \frac{(s + \frac{1}{R_s C_s})}{(s + \frac{1 + \frac{g_m R_s}{2}}{R_s C_s})(s + \frac{1}{R_D C_p})}$$

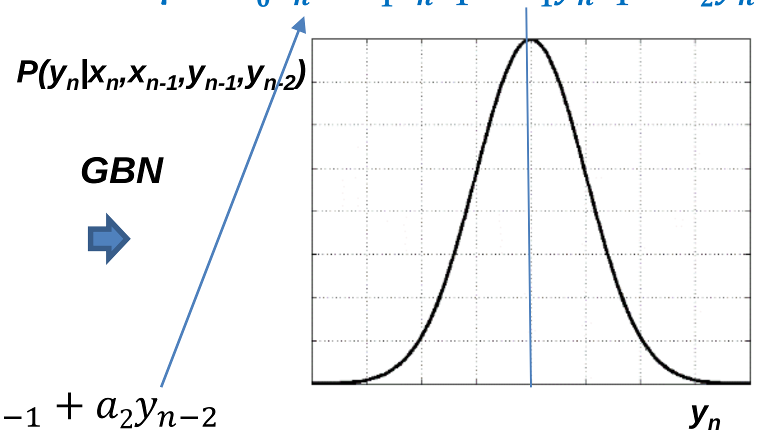
Discrete-time

$$H(z) = K \frac{b_0 + b_1 z^{-1}}{1 + a_1 + a_2 z^{-1}}$$

$$y_n = b_0 x_n + b_1 x_{n-1} + a_1 y_{n-1} + a_2 y_{n-2}$$

GBN

$$\boldsymbol{\mu = b_0 x_n + b_1 x_{n-1} + a_1 y_{n-1} + a_2 y_{n-2}}$$



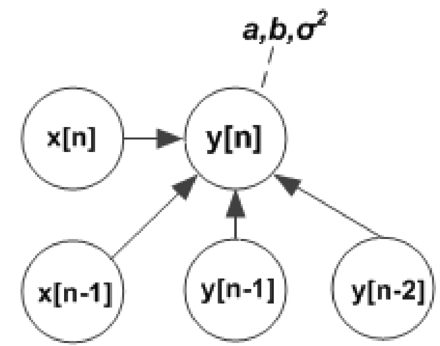
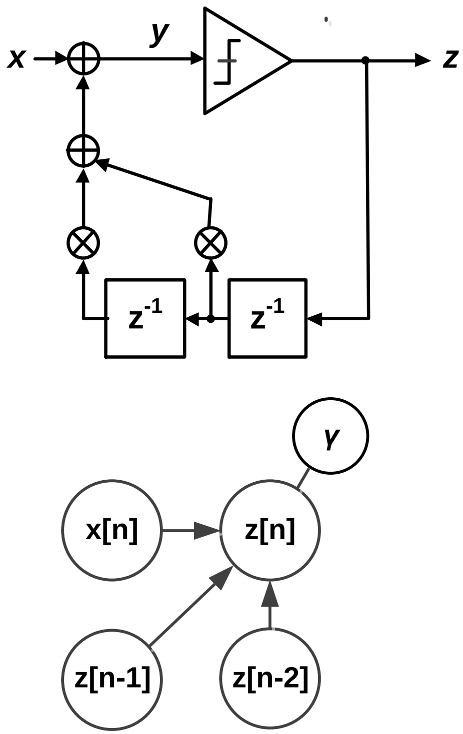
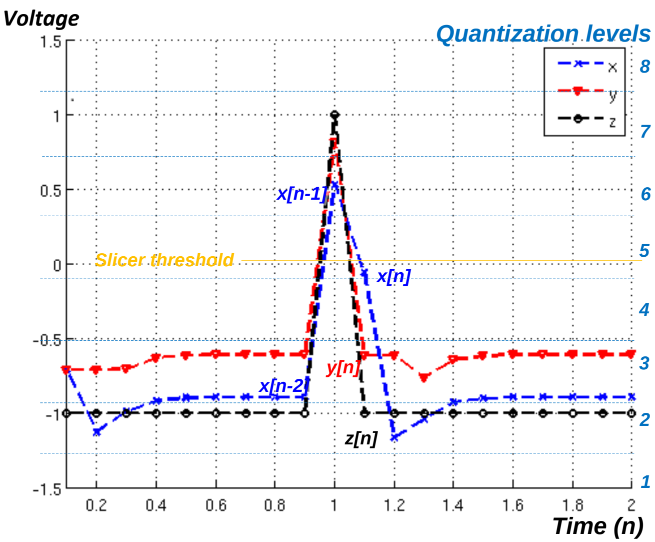


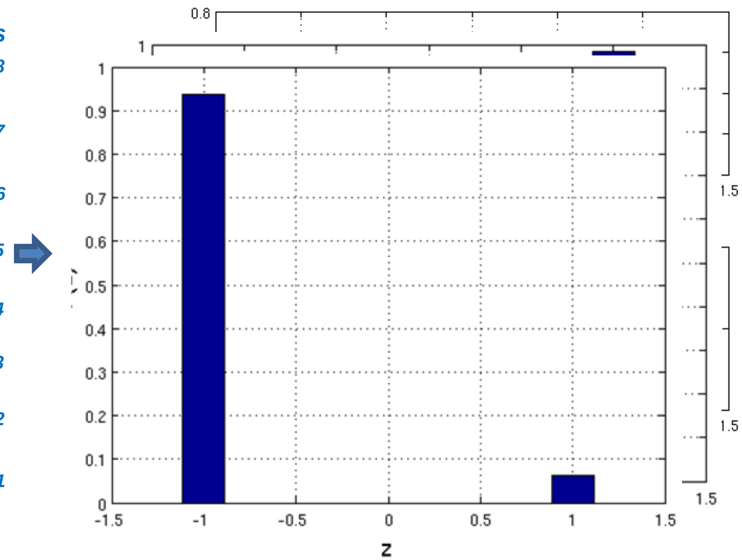
Table-Based Bayesian Network (TBN) Model Creation – Decision Feedback Equalizer

- DFE example





x[n]=input
y[n]=slicer's input
z[n]=output



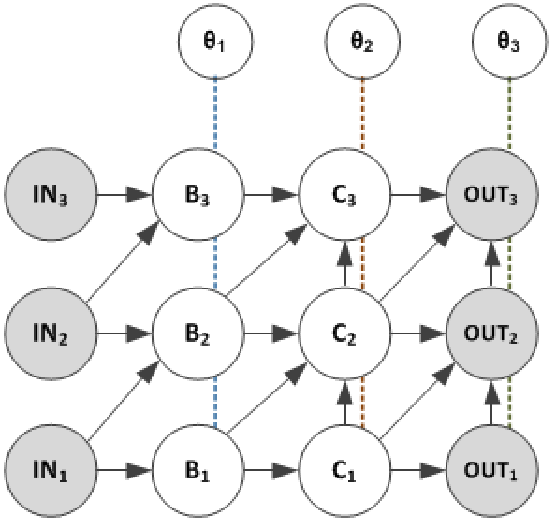
P(y[n]|x[n]=5,z[n-1]=1,z[n-2]=-1)
P(y[n]|x[n]=6,z[n-1]=-1,z[n-2]=-1)
P(y[n]|x[n]=3,z[n-1]=-1,z[n-2]=-1)

Outline

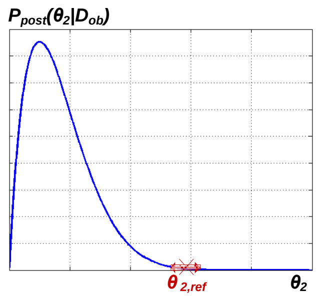
- Overview
- Bug localization using graphical models
  - Graphical model creation
    - Gaussian Bayesian network
    - Table-based Bayesian network
  - Bug localization by statistical inference
- Experimental results
- Conclusion

Bug Localization by Statistical Inference: Computing $P_{posterior}(\theta \mid D_{ob})$

- We want to estimate the probability of a parameter (θ) after observation (D_{ob}) by statistical inference
- Possible Approaches
  - Exact inference
    - Junction tree algorithm
  - Approximate inference
    - ***Gibbs sampling***

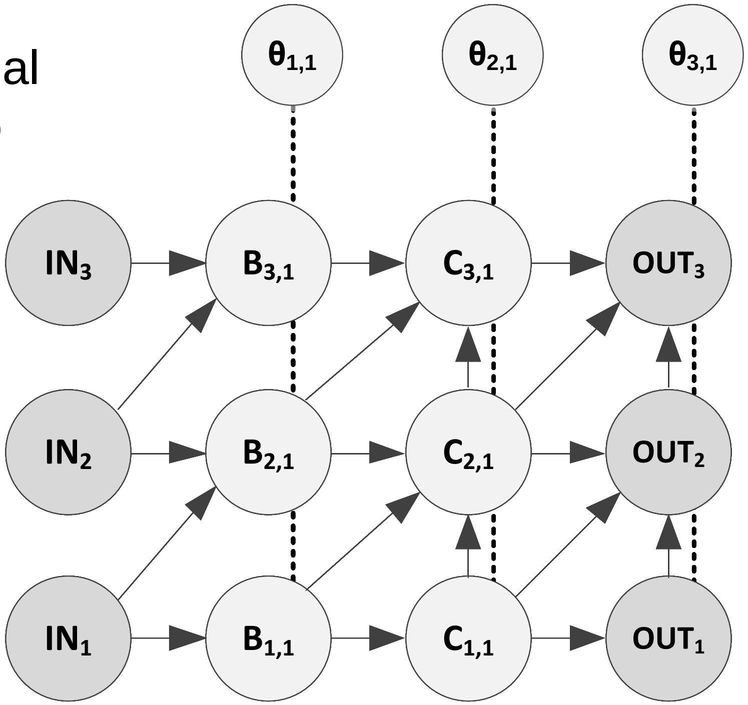


How do we get $P_{posterior}(\theta \mid D_{ob})$?



Statistical Inference by Gibbs Sampling: Computing $P_{posterior}(\theta \mid D_{ob})$

- ***Gibbs Sampling*** can be used when the conditional distribution of each variable is known and is easy to sample from
  1. Start with an initial guess $X_0=(B_{1,0}, B_{2,0}, \ldots, \theta_{3,0})$
  2. Take a sample $B_{1,1}$ from $P(B_1 \mid B_{2,0}, B_{3,0}, \ldots, \theta_{3,0})$ and update B_1
  3. Take samples for B_2 to B_3 and update them
  4. Take a sample $\theta_{1,1}$ from $P(\theta_1 \mid B_{1,0}, \ldots, \theta_{2,0}, \theta_{3,0})$ and update θ_1
  5. Take samples for C_1 to C_3 and update them
  6. Take samples θ_2 to θ_3 and update them
  7. Iterate 2~6 step N times
  8. Estimate $P_{post}(\theta \mid D_{ob}) \sim Histogram(Samples)$
     - $P(\theta_1 \mid D_{ob}) \sim Histogram(\theta_{1,k+1}, \theta_{1,k+2}, \ldots \theta_{1,k+N})$



$(B_{1,1}, B_{2,1}, \ldots, C_{1,1}, C_{2,1}, \ldots, \theta_{3,1})$

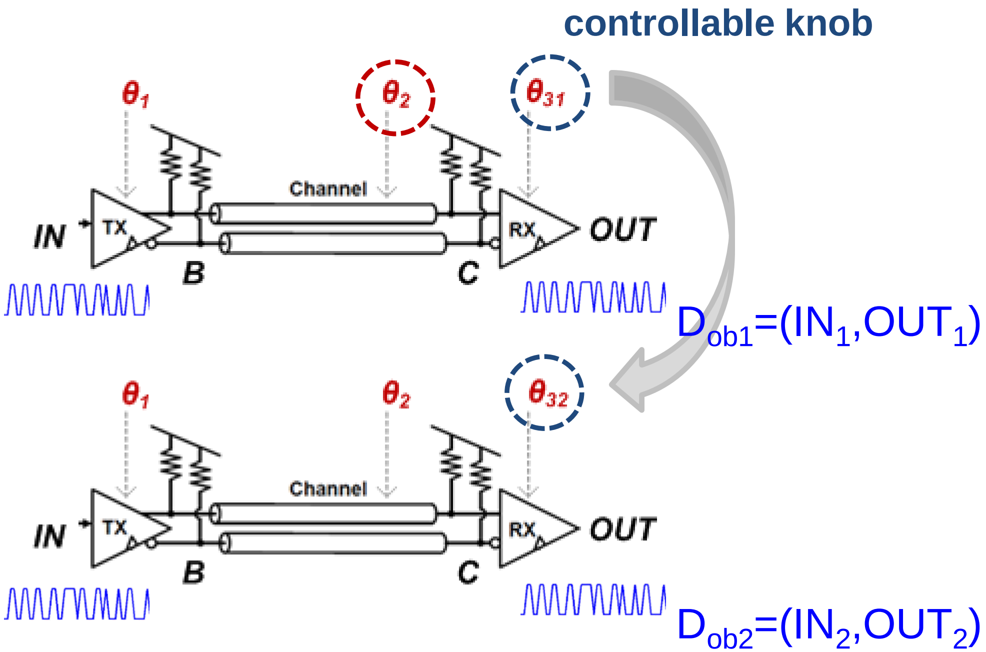
$(B_{1,2}, B_{2,2}, \ldots, C_{1,2}, C_{2,2}, \ldots, \theta_{3,2})$

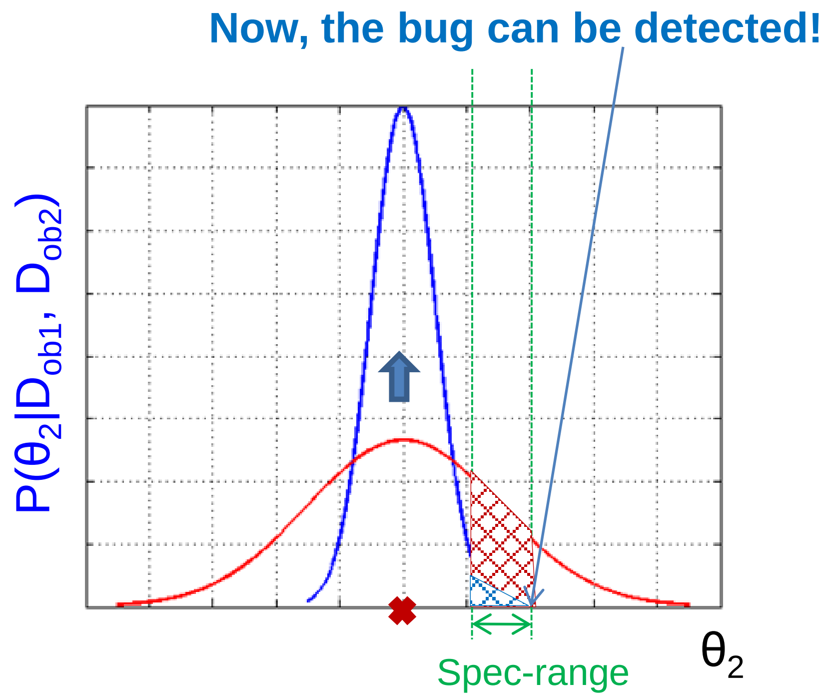
…

$(B_{1,N}, B_{2,N}, \ldots, C_{1,N}, C_{2,N}, \ldots, \theta_{3,N})$

Increasing Accuracy by Using Controllability

- The method may miss a bug root-cause due to highly **limited observability**
- However, we can increase accuracy and differentiate bug root-causes by using **controllability**



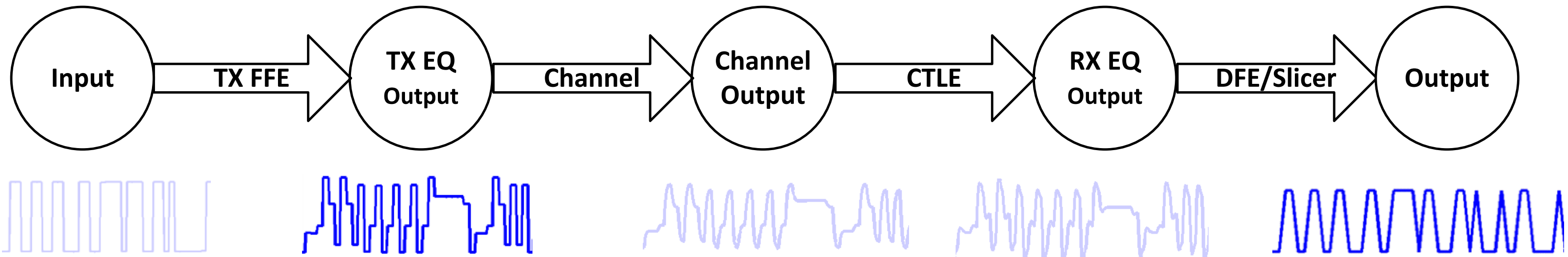


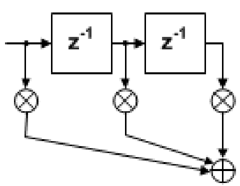
Outline

- Overview
- Bug localization using graphical models
  - Graphical model creation
    - Gaussian Bayesian network
    - Table-based Bayesian network
  - Bug localization by statistical inference
- Experimental results
- Conclusion

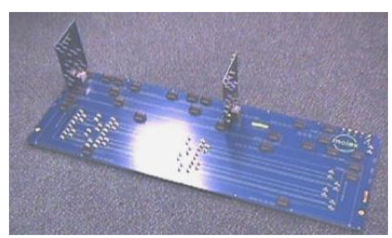
Test Case – A 5 Gbps I/O Link

- System Parameters (θ)
  - TX FFE, Channel, RX CTLE : pole / zero
  - DFE: tap coefficients / slicer threshold

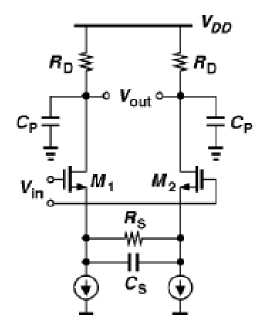




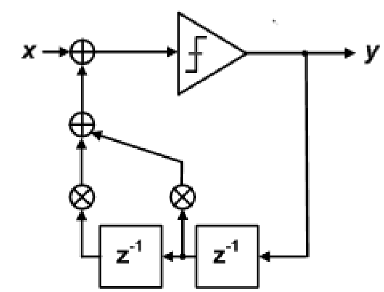
TX FFE : 3-tap



FR408 GBX Reference Backplane



Continuous Time Linear Equalizer

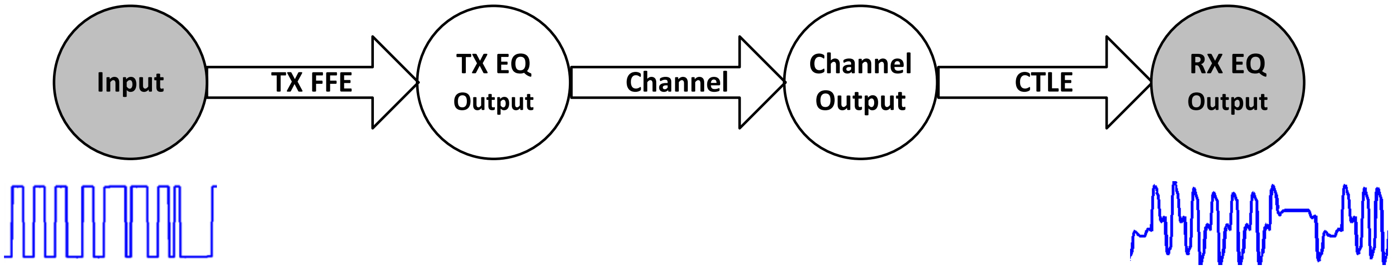


Decision Feedback Loop

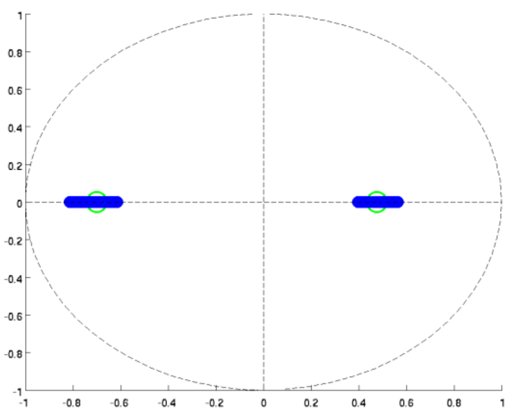
S-parameter channel model is from http://www.t11.org/ftp/t11/models/index.html

Experiment (1) – The Posteriors Cover True Parameters As Expected

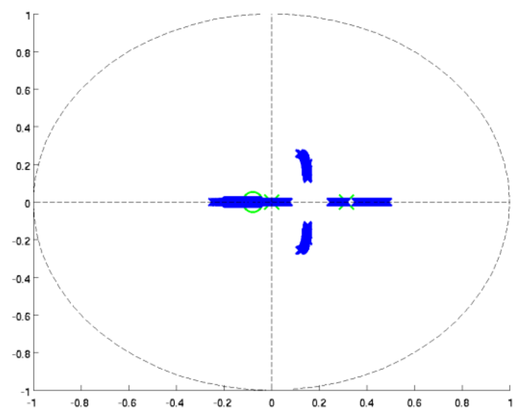
- Posterior distributions of FFE, channel and CTLE parameters and true parameter locations



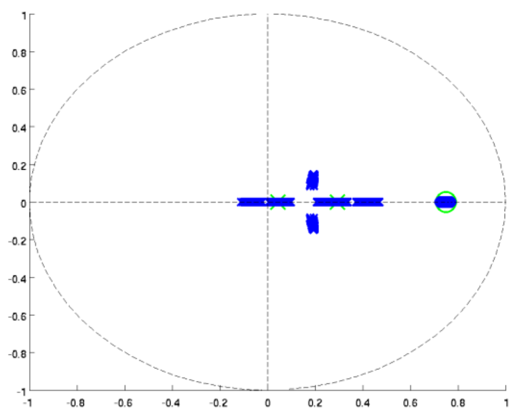
True pole/zero location (x / o)
Estimated posterior distribution of pole/zero (x / o)



Zero map of TX FFE



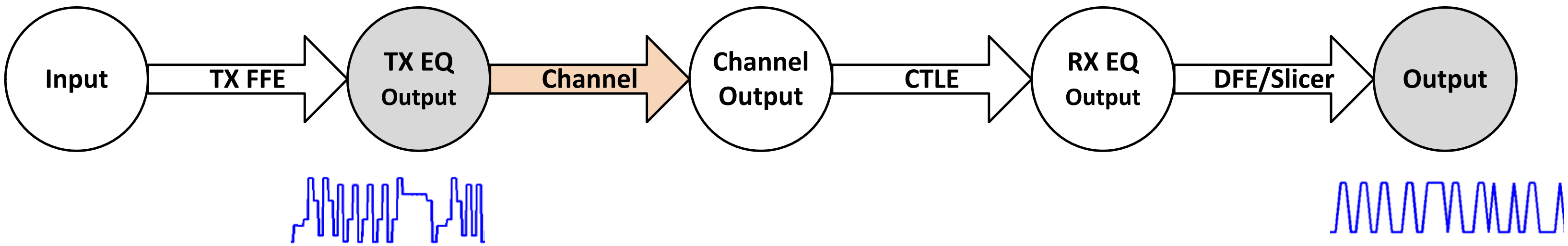
Pole/Zero map of channel

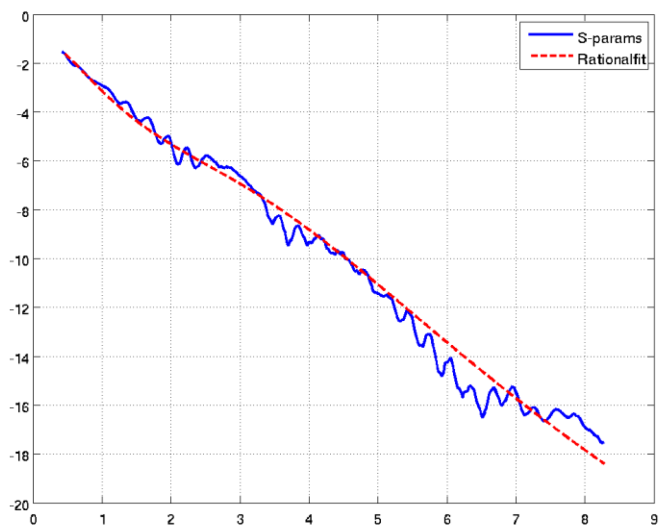


Pole/Zero map of CTLE

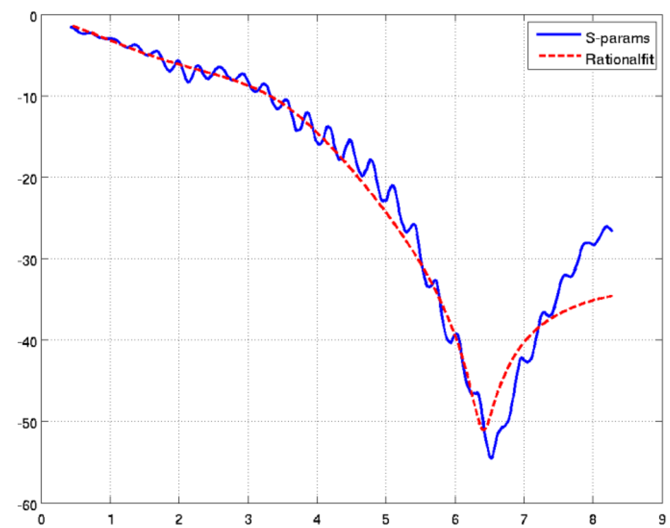
Experiment (2) – The Problematic Buggy Channel Can be Identified

- In this experiment, a channel is replaced by a problematic lossy channel





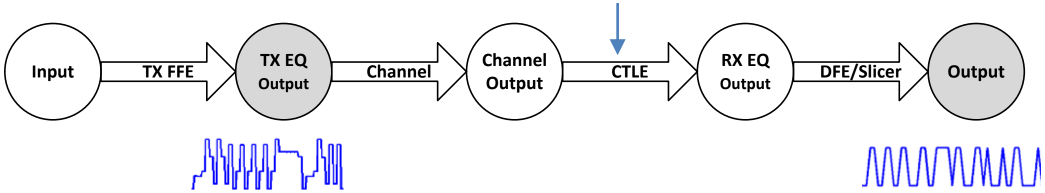
Frequency response of **desired** channel

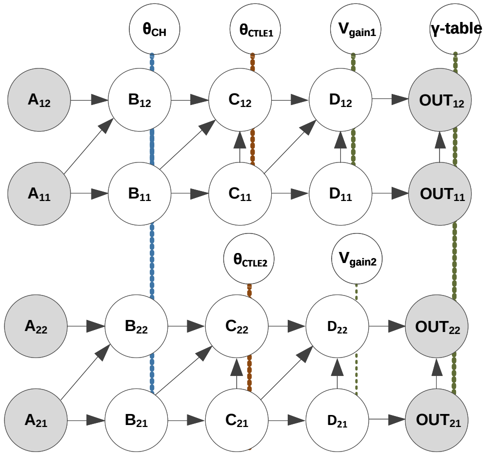


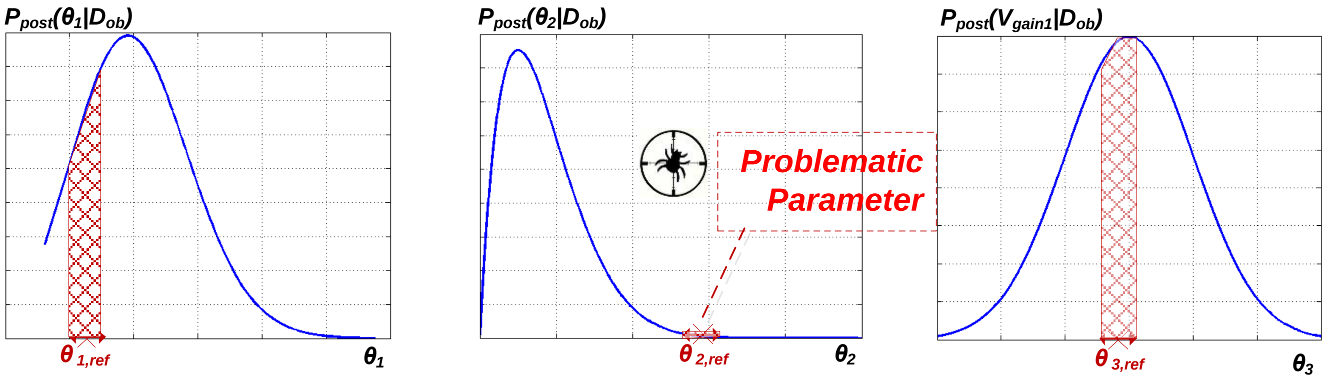
Frequency response of **lossy** channel

The Bug Localization and Ranking Procedure

controllability to CTLE's zero location







Ranking

- Rank them according to $P(\theta \text{ in } \theta_{spec}|D_{ob})$

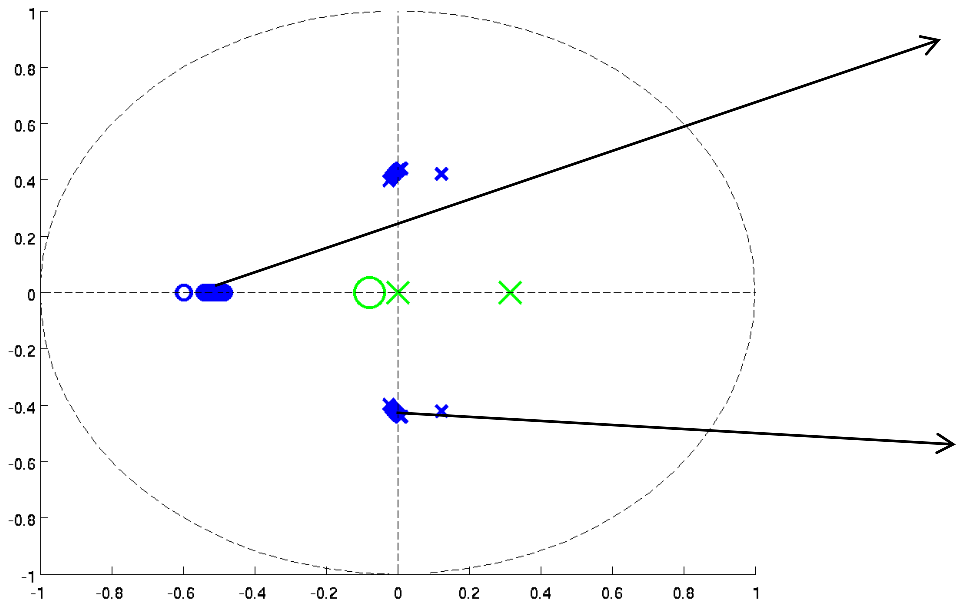
- $P(\theta_2 \text{ in } \theta_{2,spec}|D_{ob}) < P(\theta_1 \text{ in } \theta_{1,spec}|D_{ob}) < \ldots$
- *Rank in order of* $\theta_2, \theta_1, \ldots$

Experiment (2) – A Buggy Lossy Channel is Identified As the Bug Root-Cause

$P_{post}(\theta \text{ in } \theta_{spec})$	Real(z1)	Imag(z1)	Real(p1)	Imag(p1)	Real(p2)	Imag(p2)
Channel	0.17%	100%	1.7%	5.4%	15%	5.4%
CTLE	65%	100%	58%	90%	63%	90%

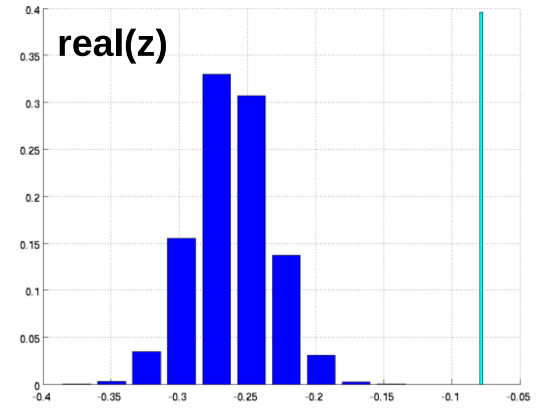
Desired pole(x)/zero(o) locations

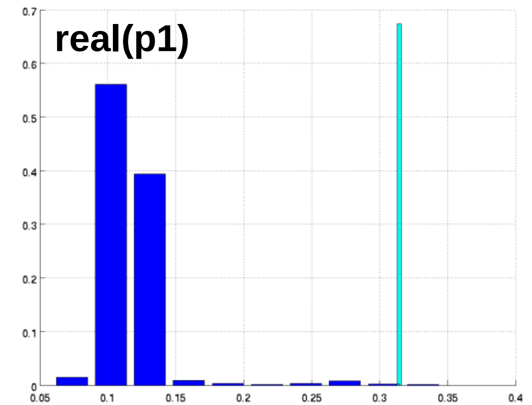
Estimated posterior distribution of pole/zero (x / o)



Estimated parameter posterior of buggy channel

Desired Parameter Value (Narrow bar)





Conclusion

- Under limited observability, the proposed bug method can automatically localize bugs
  - Nonlinearity and uncertainty could be well reflected
  - Can leverage controllability
  - Can rank multiple bug root-causes

Bug Diagnosis: Does Posterior cover a desired spec-range?