Exceptional service in the national interest

Sandia National Laboratories

A Universal Methodology and Toolkit for Quantifying Simulation Error via both Bayesian Inference and Model Reduction Strategies

Jeremiah Wilke, Khachik Sargsyan, Martin Drohmann

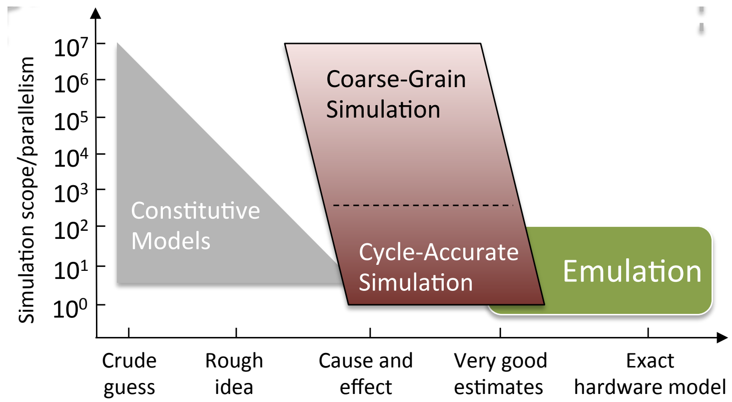
Sandia National Labs, Livermore, CA

U.S. DEPARTMENT OF ENERGY

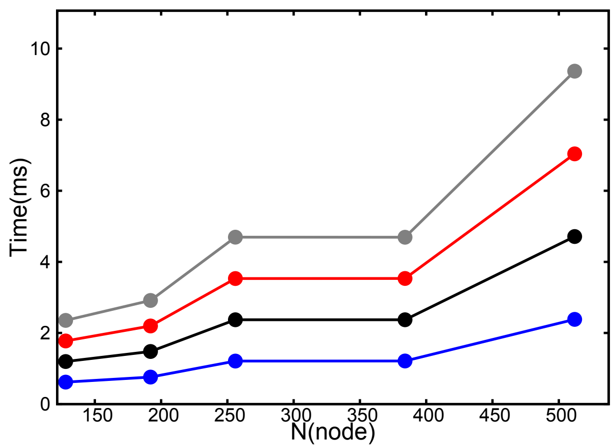
NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

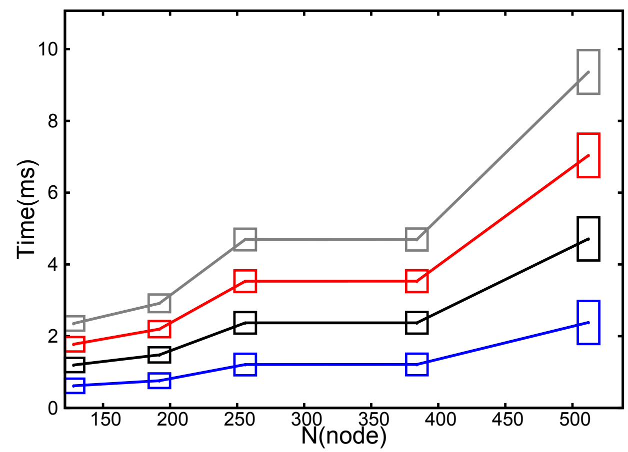
Uncertainty quantification is critical to useful simulation, regardless of detail or fidelity



Single data point isn't enough information



Need a mean value and error bound

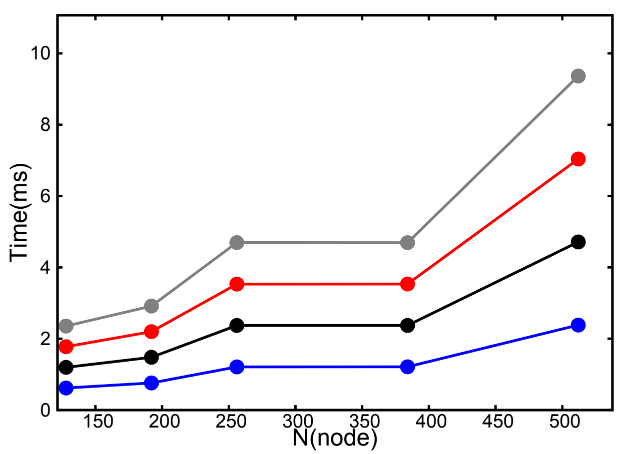


Critical question to integration of UQ tools: intrusive vs non-intrusive, code vs workflow

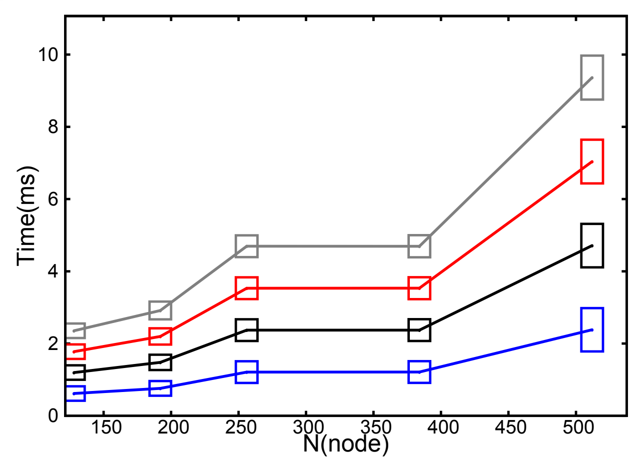
What bridges the gap from left to right?

C/C++ API
or
Toolkit

Model/simulator standalone code



Results + UQ



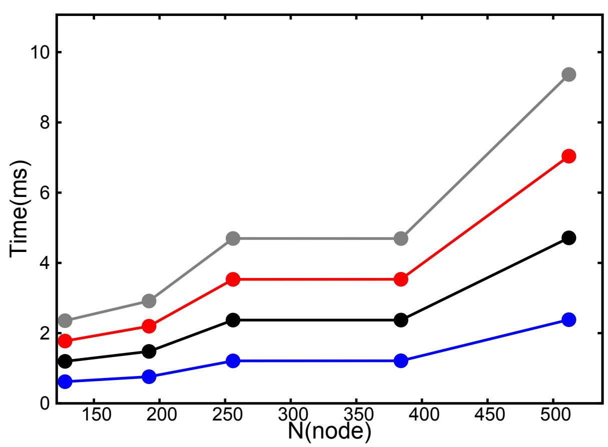
Critical question to integration of UQ tools: intrusive vs non-intrusive, code vs workflow

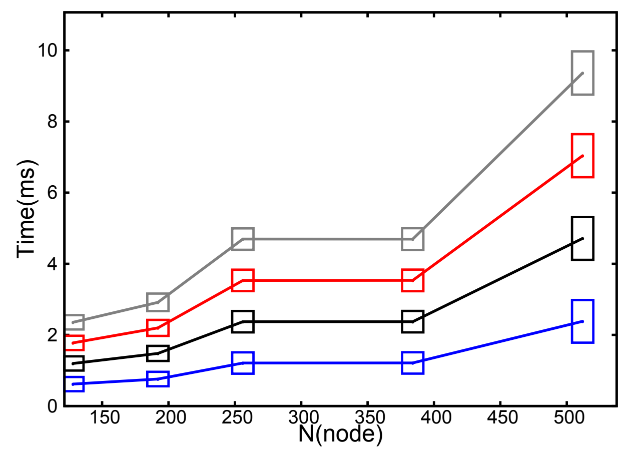
Goal of presentation here is not universal solution to UQ methods and code integration,
but *framing* the problem via two use cases

Model/simulator
standalone code

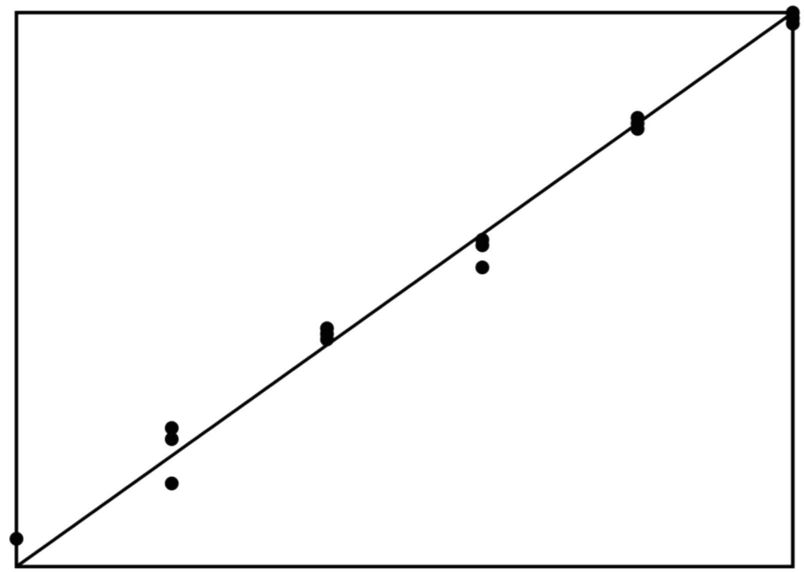
C/C++ API
or
Toolkit

Results + UQ





Where we understand UQ better: experimental noise and series expansion



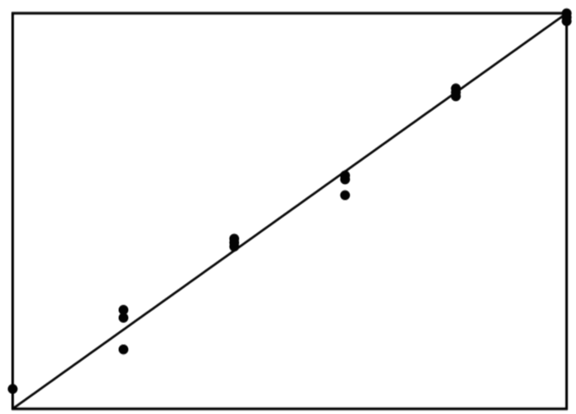
Errors due to randomness or experimental scatter

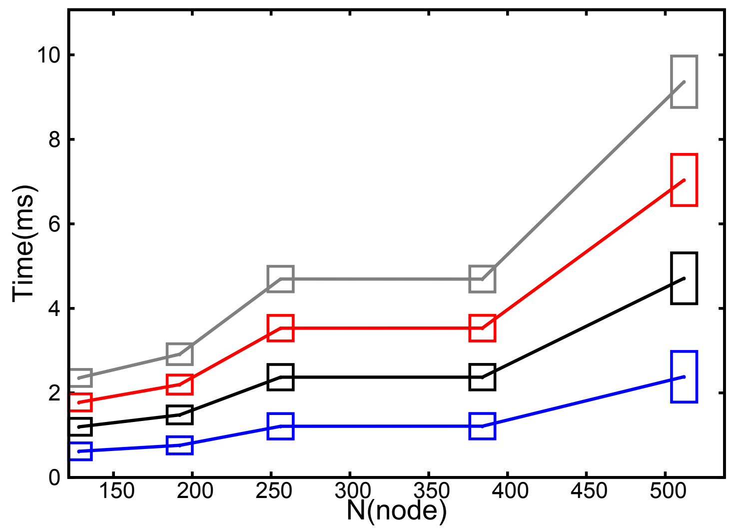
$$F(x) = \sum_{i=0}^{k} \frac{f^{(i)}(a)}{i!}(x-a)^i + R(x)$$

$$R(x) = \frac{f^{(k+1)}(Z)}{k!}(x-Z)^k(x-a)$$

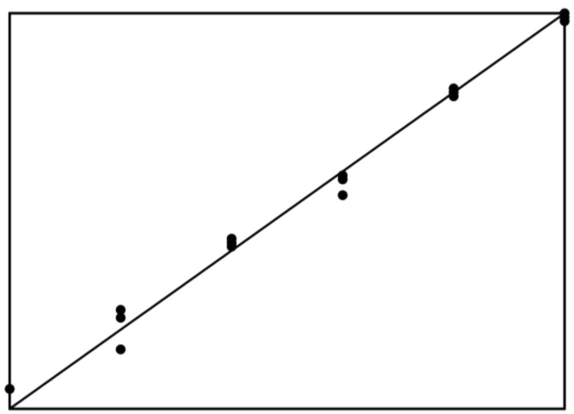
Analytic formula derived from Taylor series

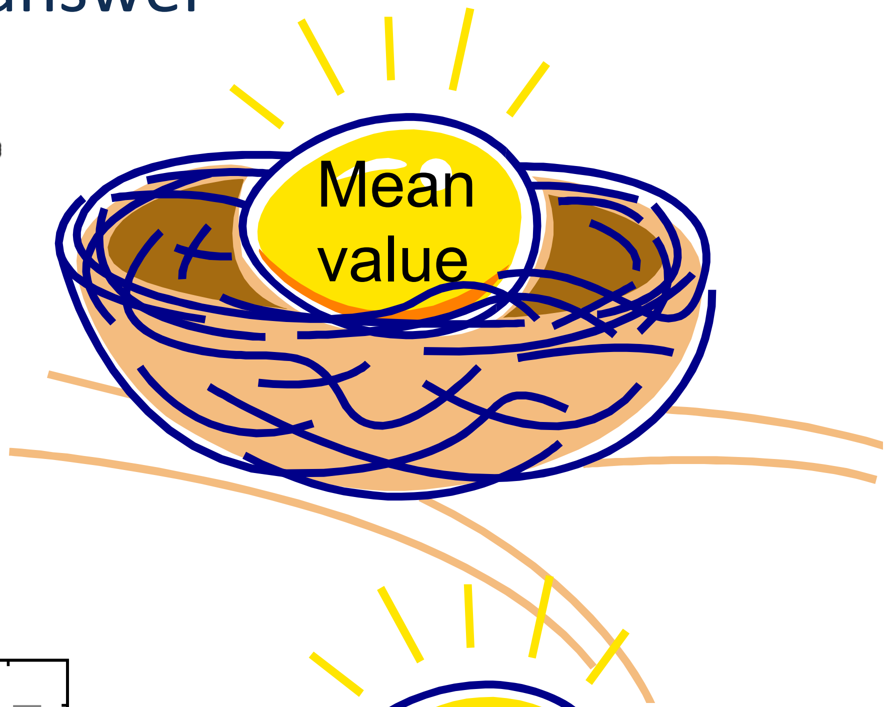
The chicken and the egg of UQ: Knowing the error without knowing the answer

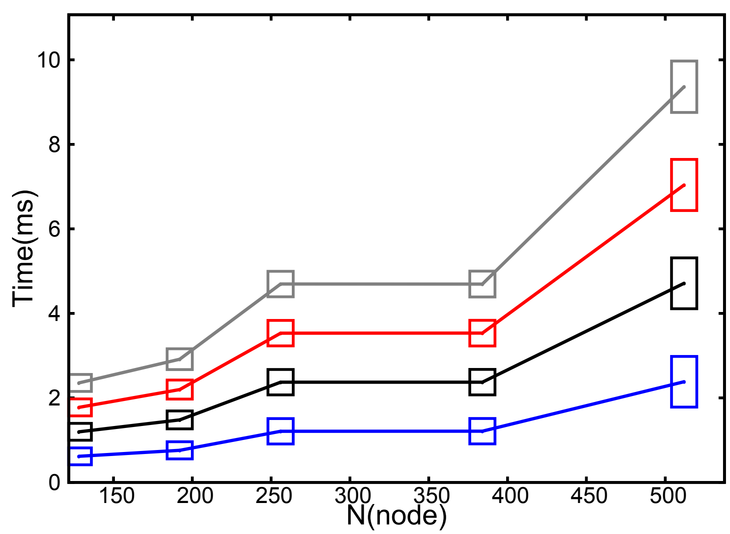


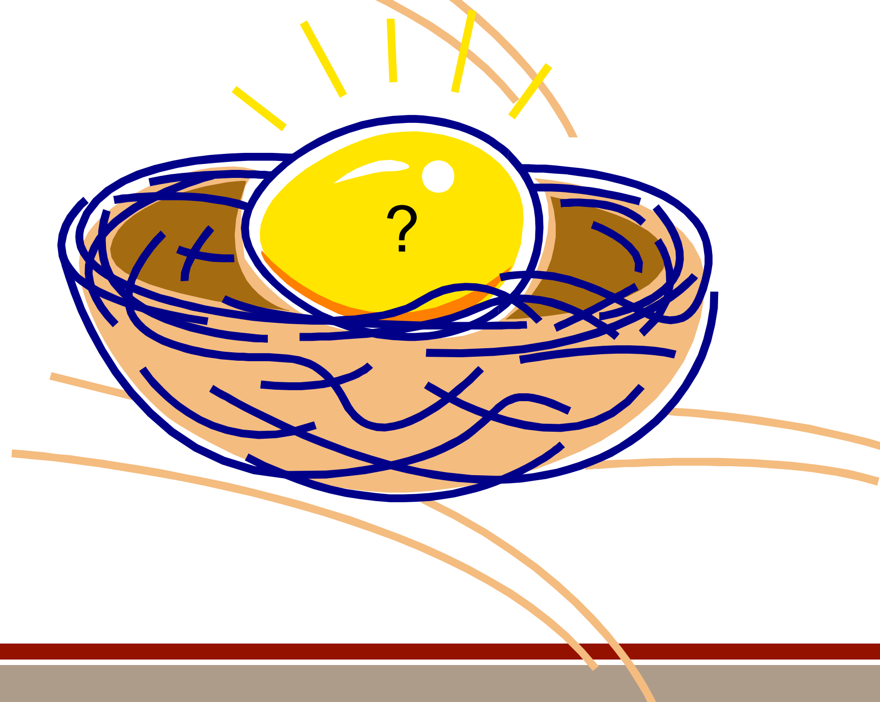


The chicken and the egg of UQ: Knowing the error without knowing the answer









Making Bayesian inference a universally understood concept

Sandia National Laboratories

Posterior:
Given prior knowledge and new data, encapsulates best knowledge of parameters

Likelihood Function:
Physically motivated likelihood estimate of data points assuming a set of parameters

Prior:
Encapsulates prior knowledge of problem

$$P\left(\{\lambda_i\} \middle| \{x_i\}\right) \propto P\left(\{x_i\} \middle| \{\lambda_i\}\right) \times P\left(\{\lambda_i\}\right)$$

Quantity of interest, but not possible to directly estimate

NOT quantity of interest, but can be estimated directly

Infer posterior from physically motivated likelihood!

Making Bayesian inference a universally understood concept

Posterior:
Given prior knowledge and new data, encapsulates best knowledge of parameters

Likelihood Function:
Physically motivated likelihood estimate of data points assuming a set of parameters

Prior:
Encapsulates prior knowledge of problem

$$P\left(\{\lambda_i\} \,\middle|\, \{x_i\}\right) \propto P\left(\{x_i\} \,\middle|\, \{\lambda_i\}\right) \times P\left(\{\lambda_i\}\right)$$

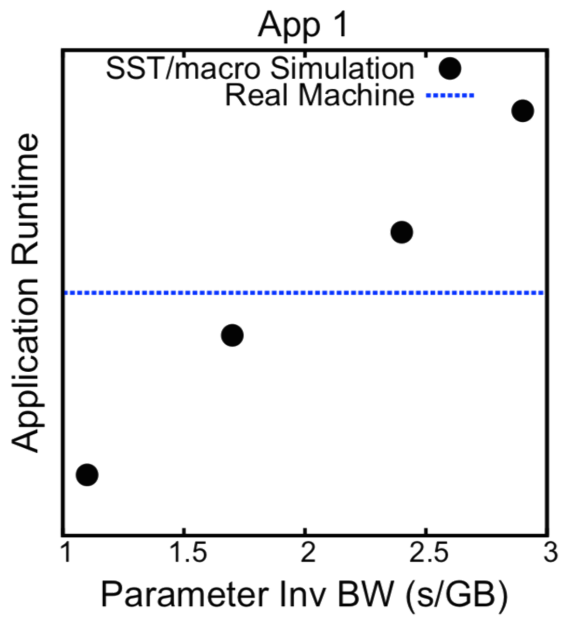
Data points:
Coin flip: heads or tails
Simulated runtime

Model parameters:
Coin flip: P(heads) = H
Latency/bandwidth

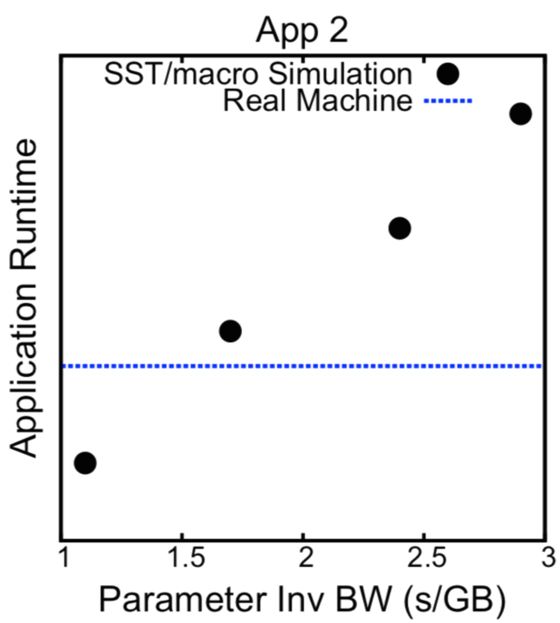
$P\left(\{x_i\} \,\middle|\, \{\lambda_i\}\right)$ Given physical model, $\{\lambda_i\}$, estimate likelihood of data

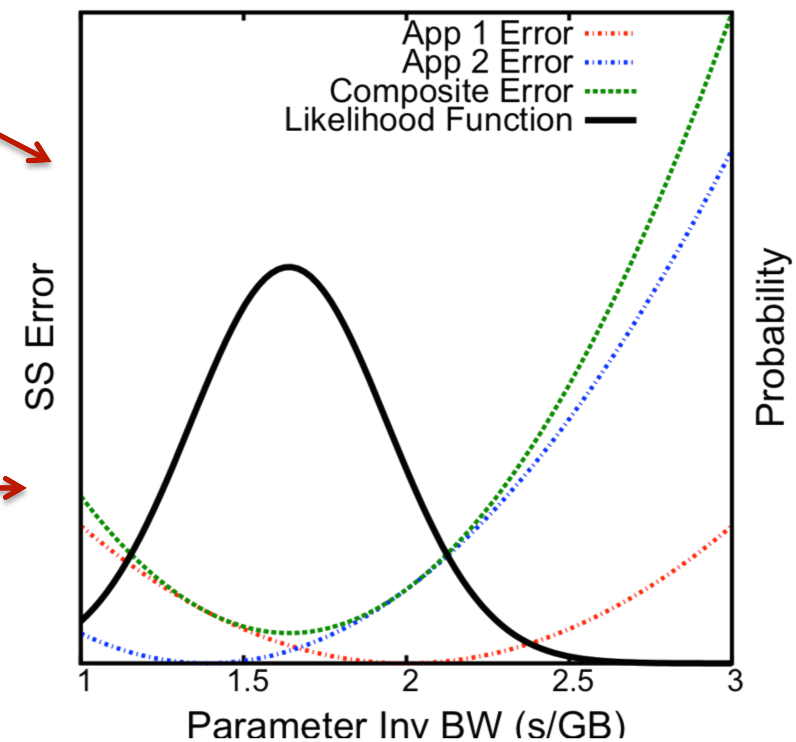
$P\left(\{\lambda_i\} \,\middle|\, \{x_i\}\right)$ Given data, $\{x_i\}$, estimate likelihood of physical model

How to translate abstract "model imperfection" into something more tangible

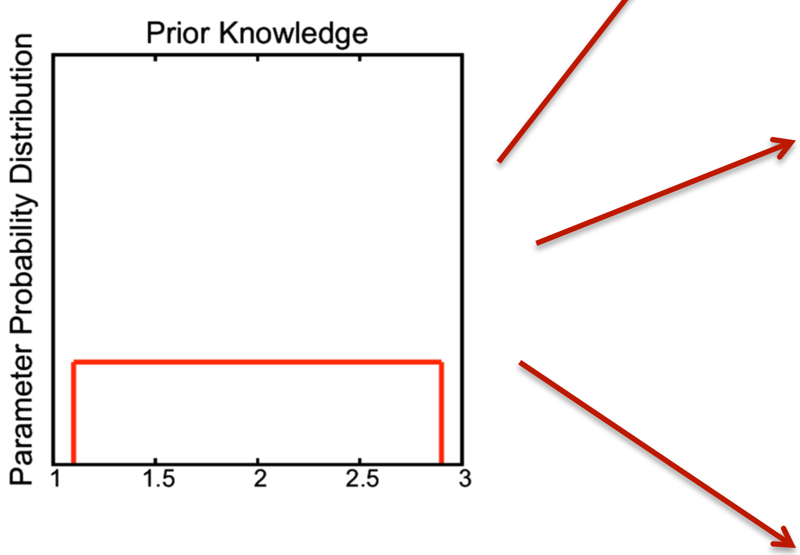


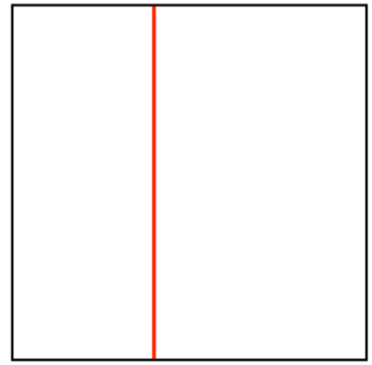
- App 1 simulation suggests 2.0 s/GB
- App 2 simulations suggests 1.5 s/GB
- No single bandwidth is exact
- "Most likely" parameter is ~1.6



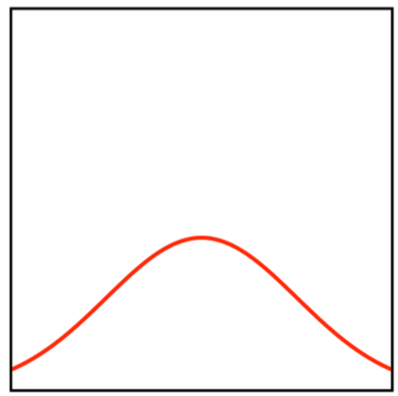


How does model uncertainty manifest itself in parameter uncertainties?

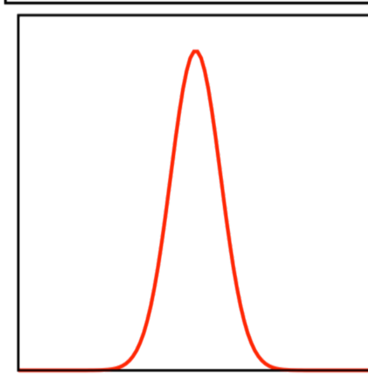




Single BW parameter exact for all tests. “Delta” function probability distribution.



Simulator is generally inaccurate
OR
Many parameters are equally accurate



No single parameter is exact for all tests, but small parameter range gives high accuracy

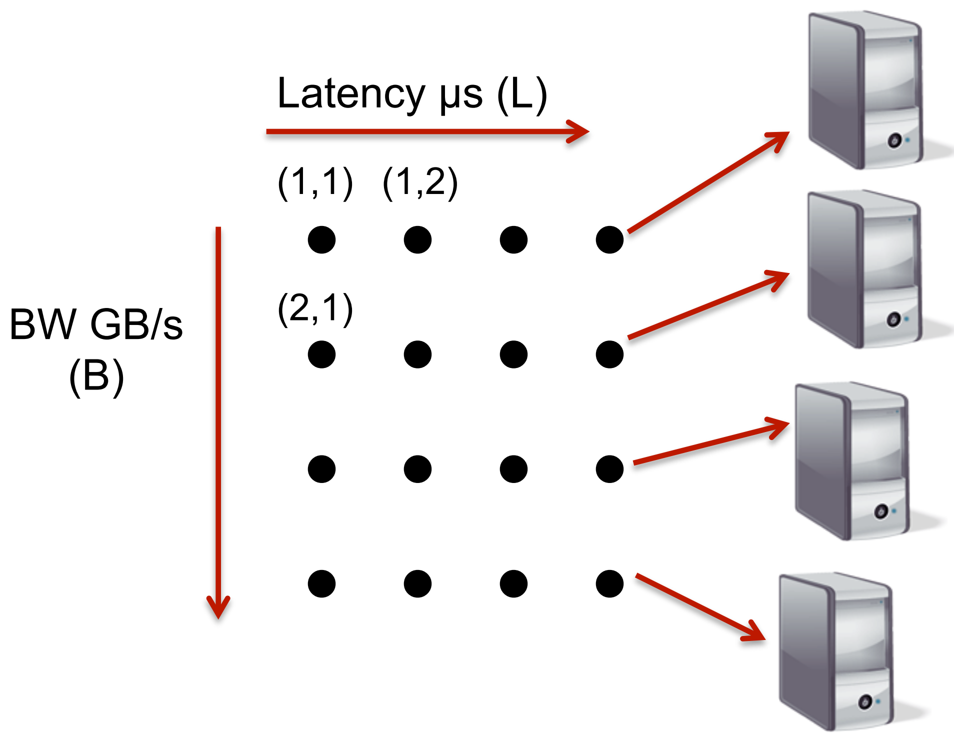
Calibration phase requires many, many samples in parameter space to build distributions

$$M(\{\lambda_i\}) \approx S(\{\lambda_i\}) = \sum_k c_k \Psi_k(\xi)$$

Simulation Model

Surrogate polynomial

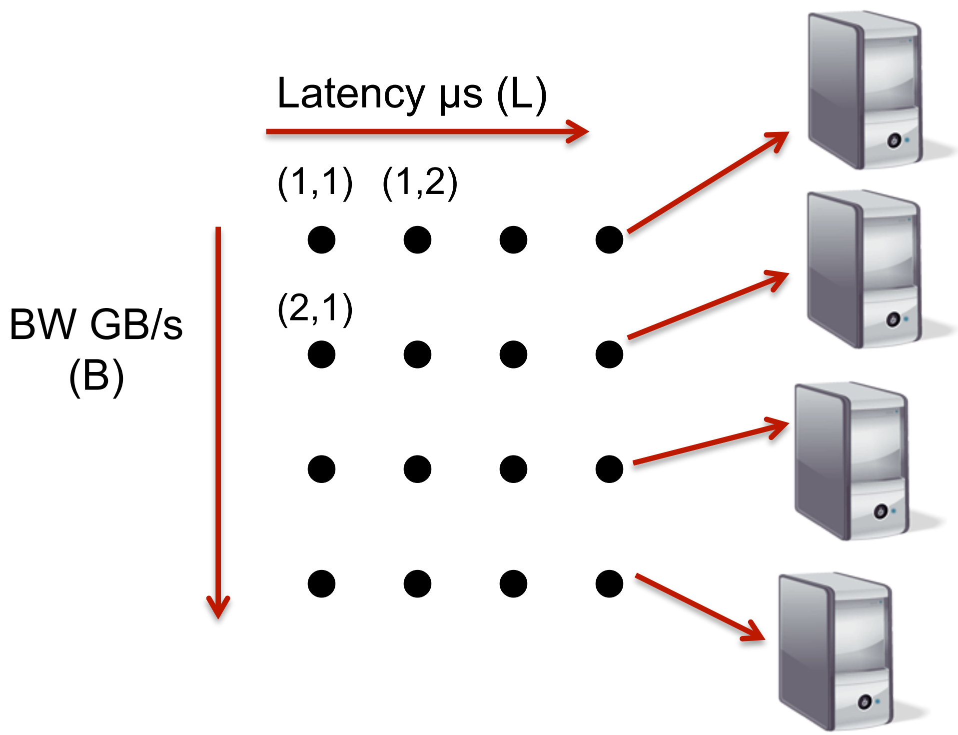
Expansion coefficients



- Coefficient c_k fit over grid in parameter space
- Computation of surrogate is embarrassingly parallel
- Polynomial allows rapid AMCMC sampling of model output in parameter space
- Expansion coefficients used in sensitivity analysis

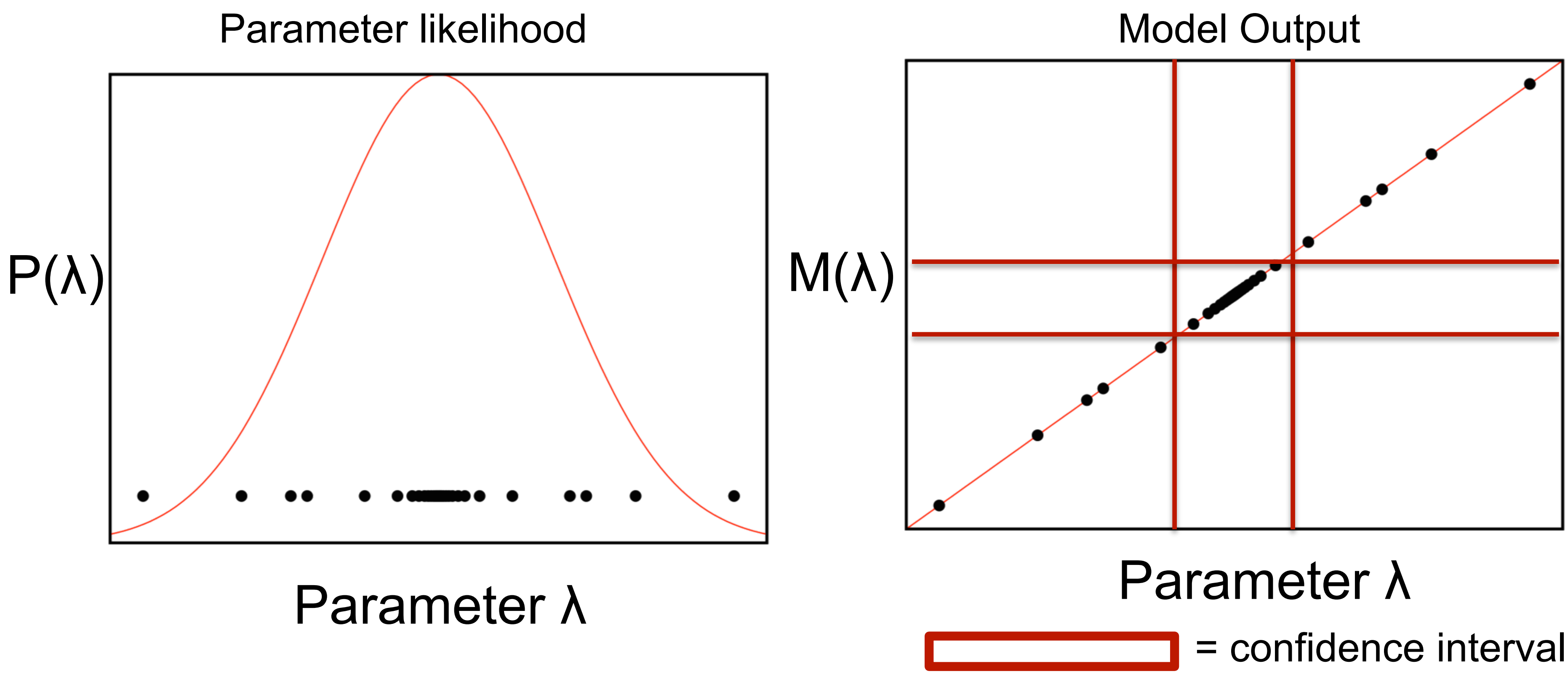
Calibration phase requires many, many samples in parameter space to build distributions

Extensive sweep of parameter space for modest problem sizes where we have “correct” answer or we have really good idea of “correct” answer



Extrapolating uncertainties into the unknown still requires sampling

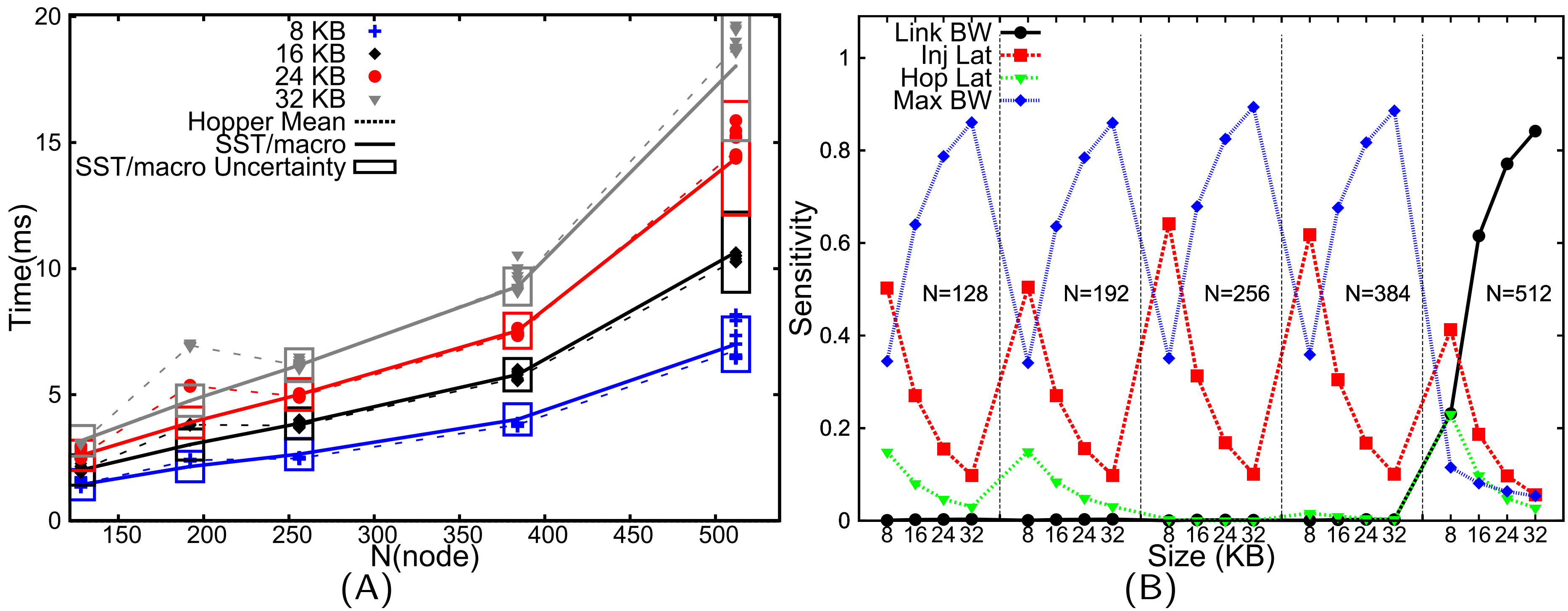
- Parameter likelihood distributions inform where to sample
- With few samples, can build semi-quantitative confidence interval
- Workflow integration to generate/run/collect/analyze samples



Four step UQ workflow for Bayesian Inference: not intrusive to existing codes

- ***Generate:*** list of samples in parameter space generated by UQ toolkit
- ***Run:*** parameter inputs through simulation/model
- ***Collect:*** simulation/model outputs into standard format
- ***Analyze:*** run outputs through machinery in UQ toolkit to generate uncertainty distributions

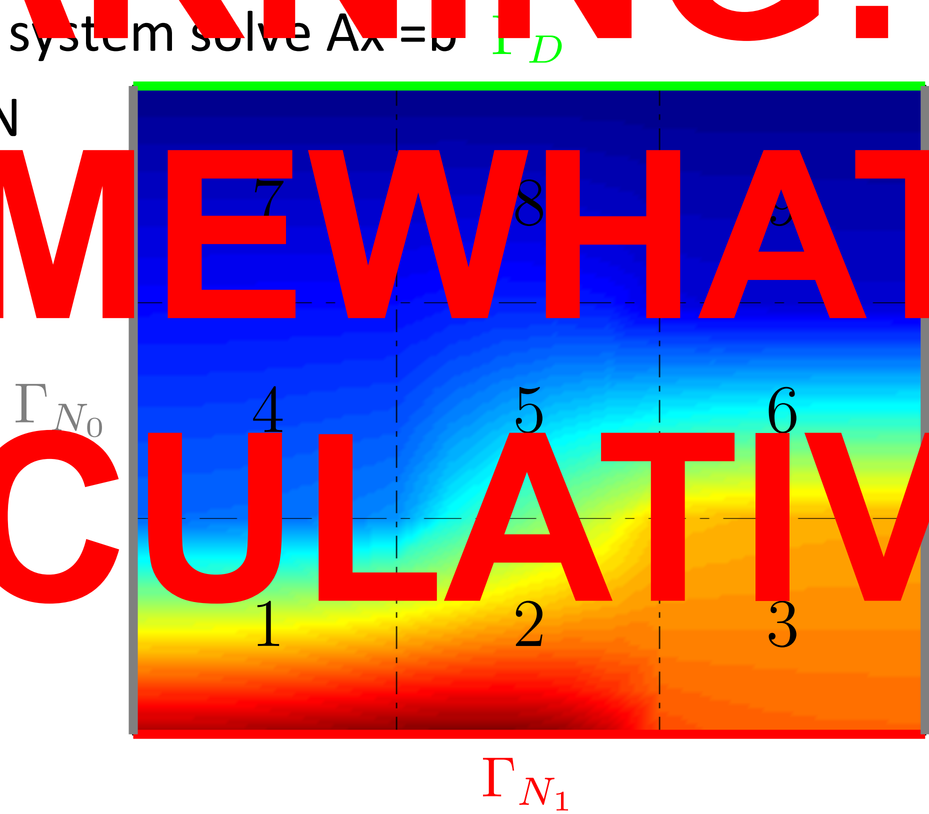
Output from analyze phase: error distributions and parameter sensitivities



Reduced-order models or how PDE solvers are way ahead of discrete event simulations

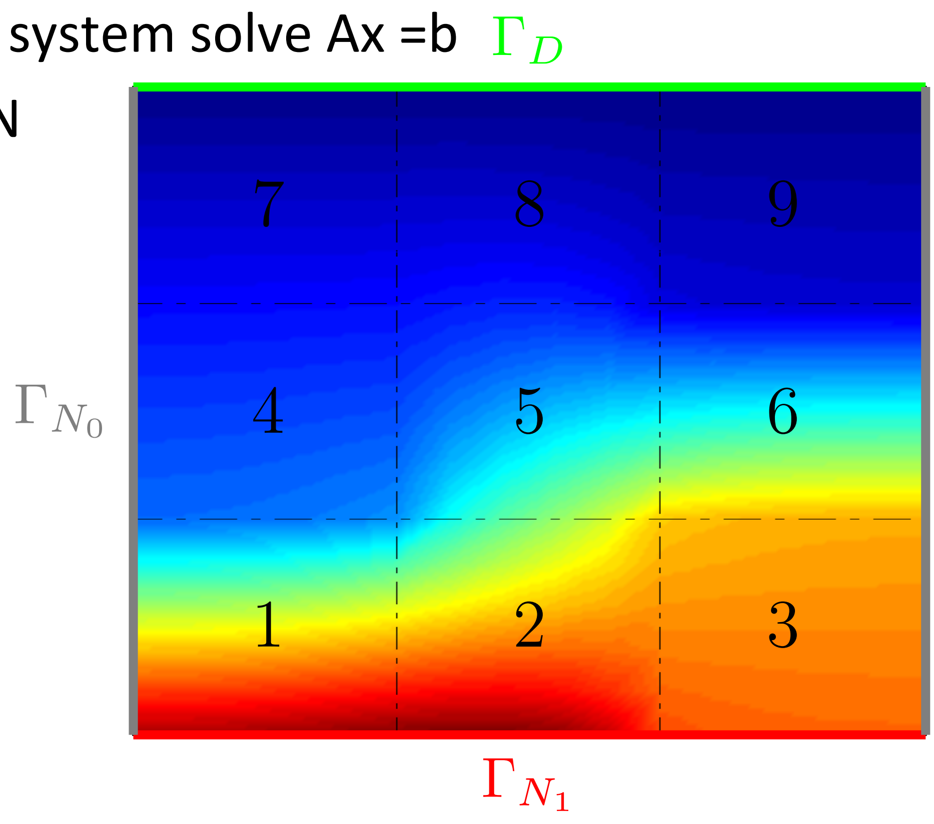
- Rather than sampling in parameter space, can a simulation self-diagnose its own errors?
- How many basis functions do you need to accurately describe heat flow problem with 9 different regions?
- Amounts to linear system solve Ax = b
- x is vector of size N

WARNING: SOMEWHAT SPECULATIVE



Reduced-order models or how PDE solvers are way ahead of discrete event simulations

- Rather than sampling in parameter space, can a simulation self-diagnose its own errors?
- How many basis functions do you need to accurately describe heat flow problem with 9 different regions?
- Amounts to linear system solve Ax =b
- x is vector of size N

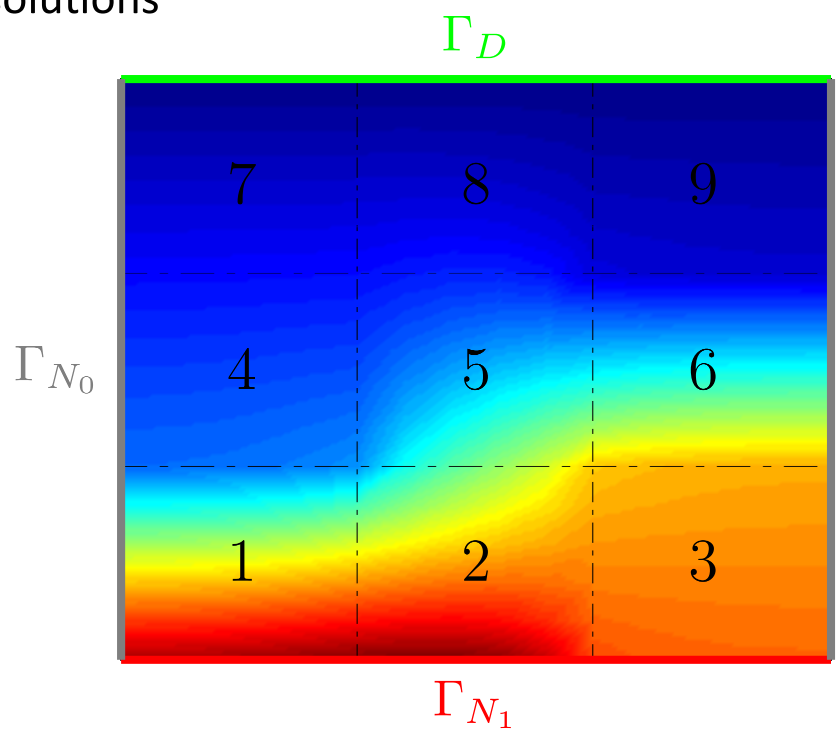


Reduced-order models or how PDE solvers are way ahead of discrete event simulations

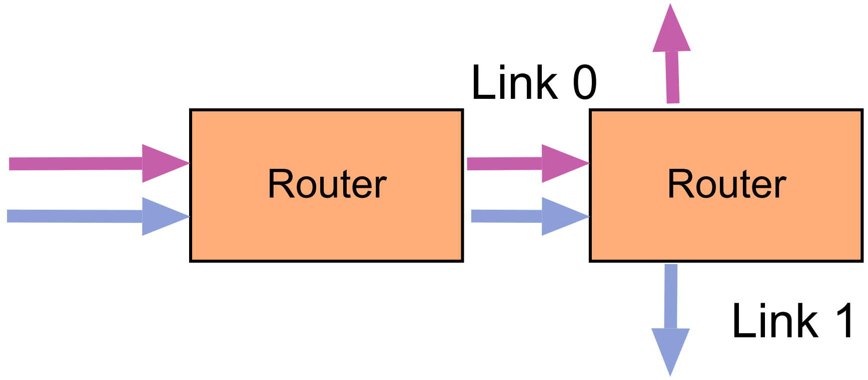
- Full order model
  - N = 10,000
  - Huge sparse system
  - Brute force solution
- Reduced order model
  - What if I already have several solutions x_1, x_2, x_3…? How accurate is:

$$\tilde{x} = \sum_i c_i x_i$$

  - Small, dense system



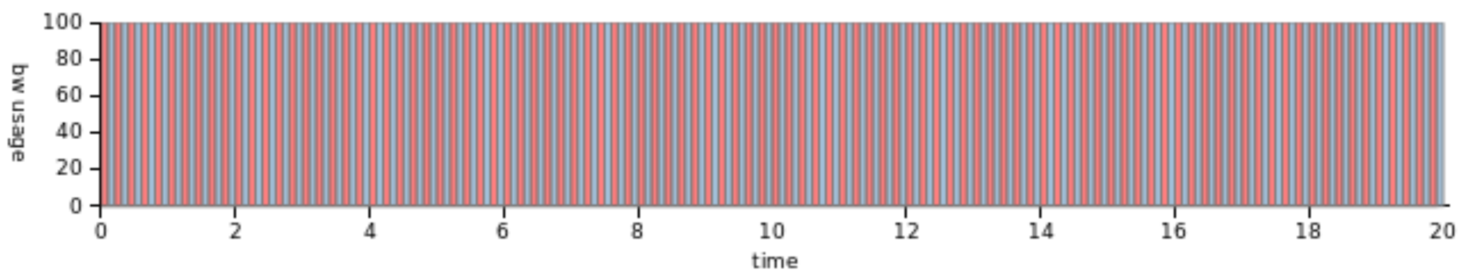
How is a simulation a reduced order model?



Two flows competing for bandwidth

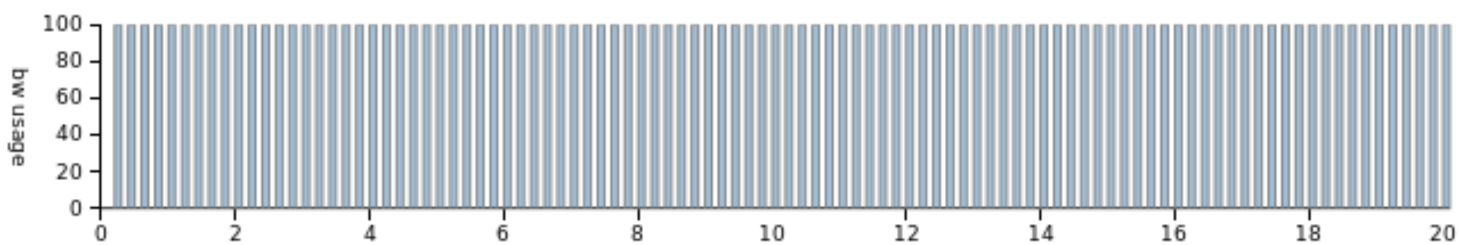
Discretization into flits shows even sharing of bandwidth on link 0

Link 0 (maximum bw: 100)

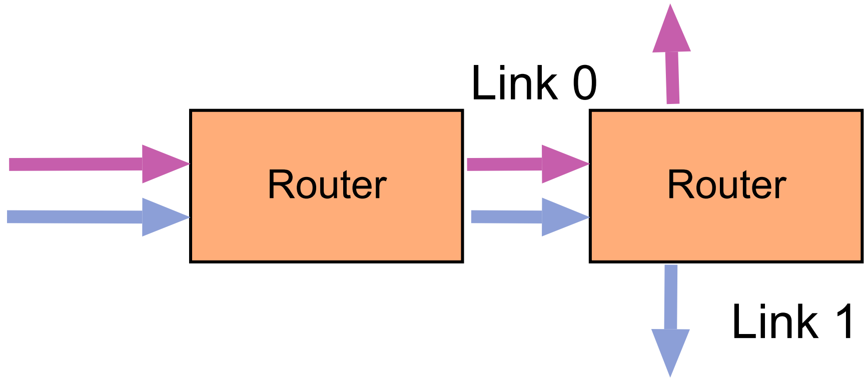


Link 1 (maximum bw: 100)

Link 1 consistently utilized at half rate

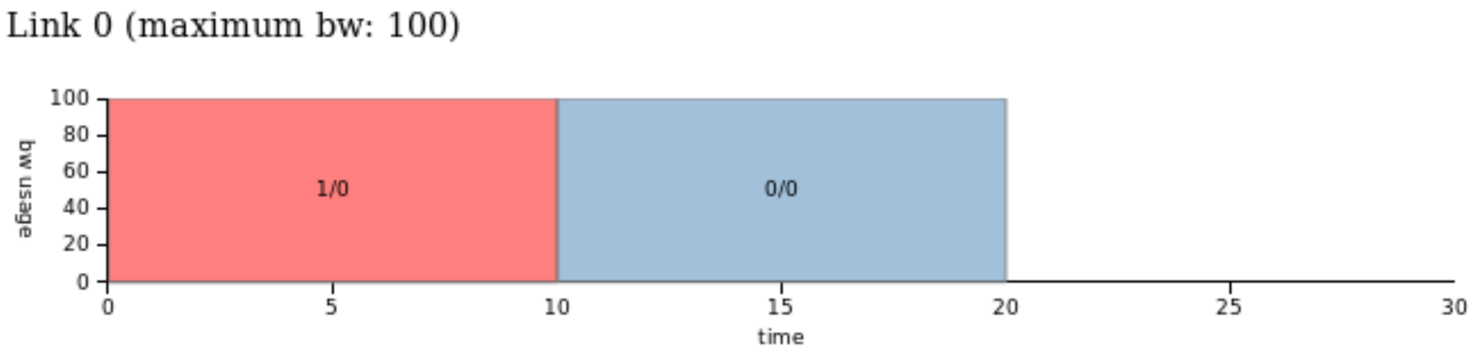


How is a simulation a reduced order model?

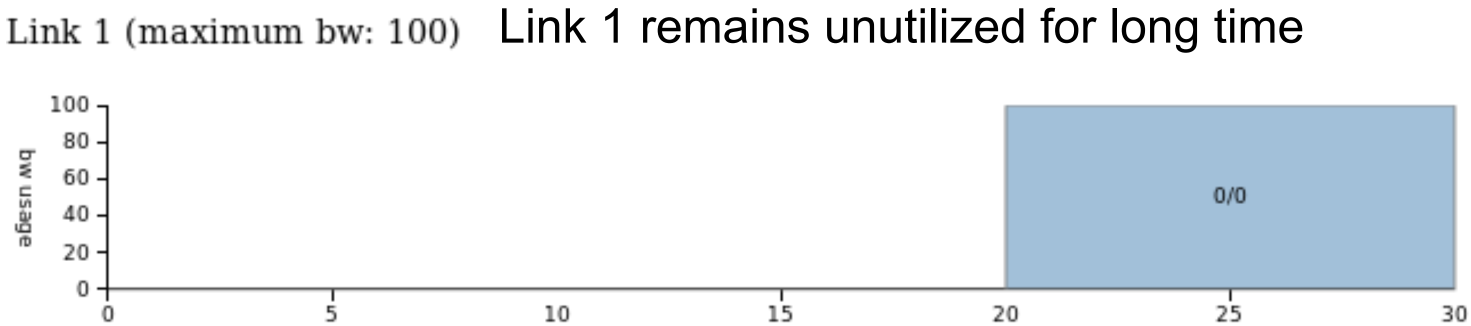


Two flows competing for bandwidth

Discretization shows uneven sharing of bandwidth on link 0



Link 1 remains unutilized for long time



How does reduced order model perspective help us get at error?

- We want to know to error

$$E = \left| \tilde{x} - x \right|$$

- All we can compute is residual

$$R = \left| A\tilde{x} - b \right|$$

How does reduced order model perspective help us get at error?

- We want to know to error

$$E = \left|\tilde{x} - x\right|$$

- All we can compute is residual

$$R = \left|A\tilde{x} - b\right|$$

- Can we train computer to convert R to E?
- Residual is an *indicator* for the error we care about

How does reduced order model perspective help us get at error?

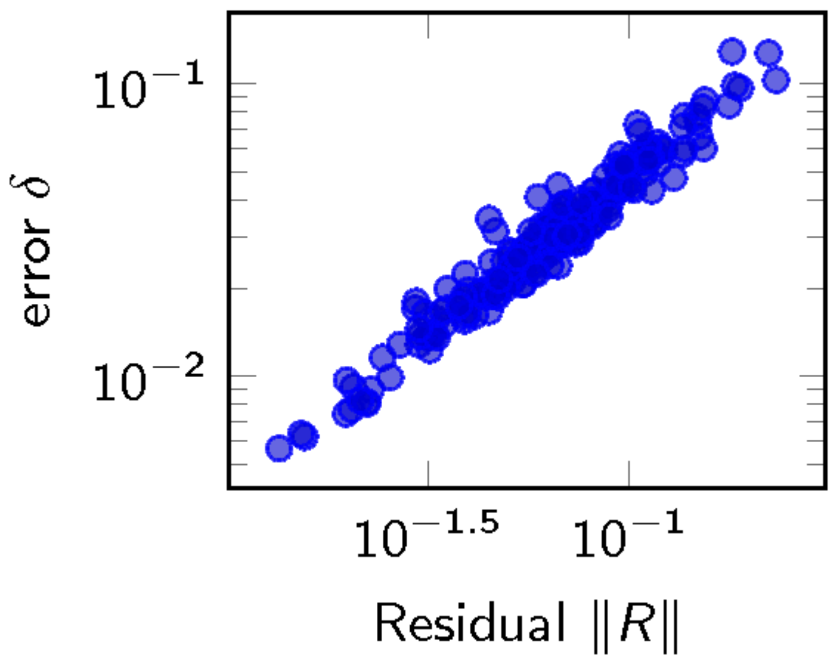
- We want to know to error

$$E = \left|\tilde{x} - x\right|$$

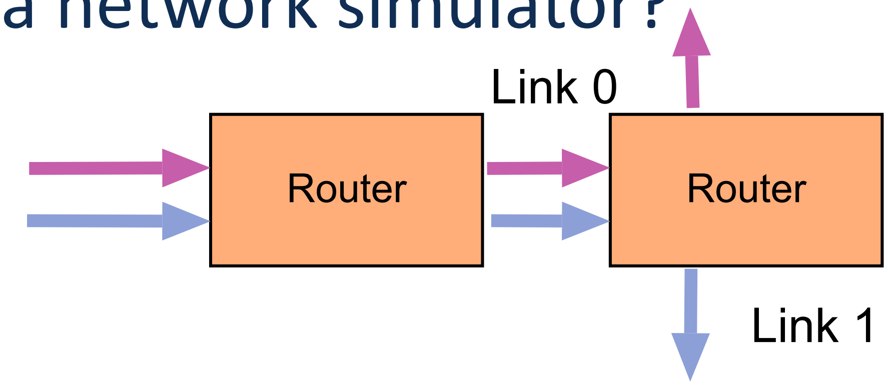
- All we can compute is residual

$$R = \left|A\tilde{x} - b\right|$$

- Can we train computer to convert R to E?
- Residual is an *indicator* for the error we care about



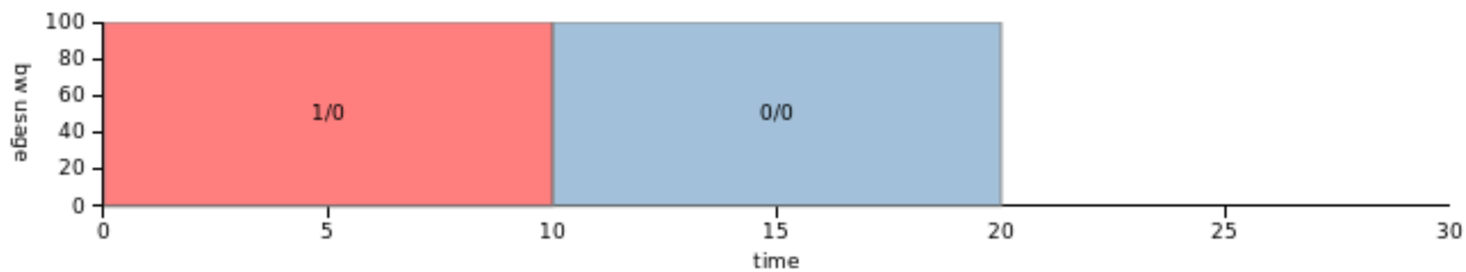
What might be an indicator for error in a network simulator?



Two flows competing for bandwidth

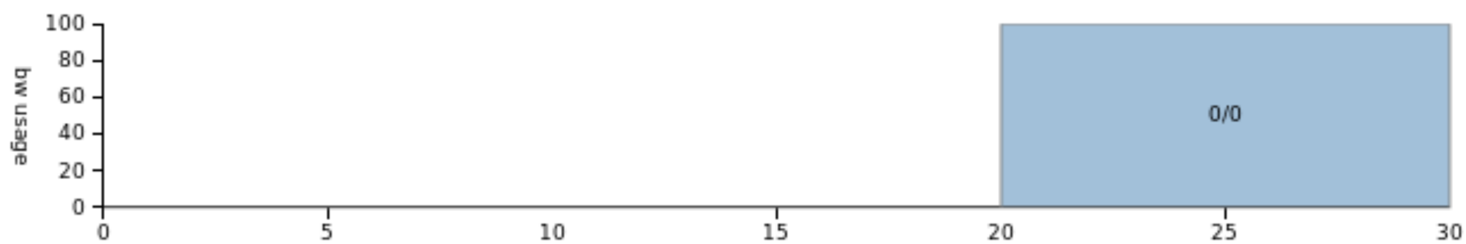
Even if we don't exactly model flit-level flow control, we still know that we did something wrong!

Link 0 (maximum bw: 100)



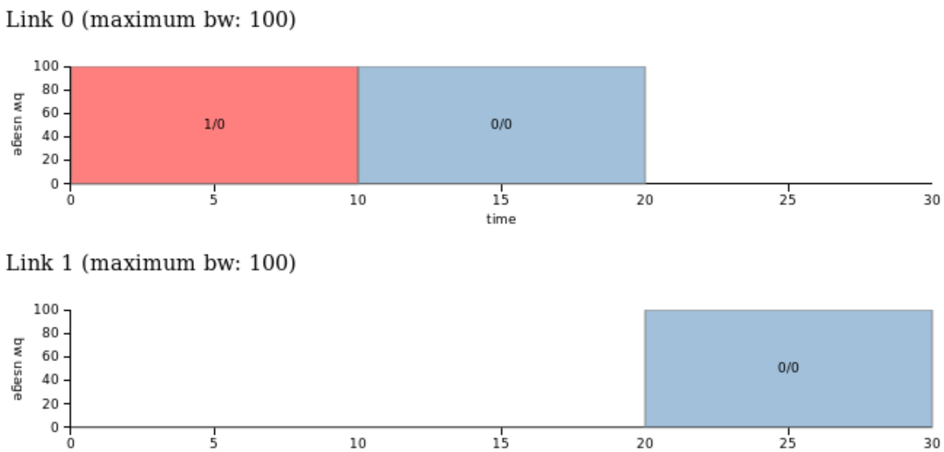
Here we have no idea that an error was made!

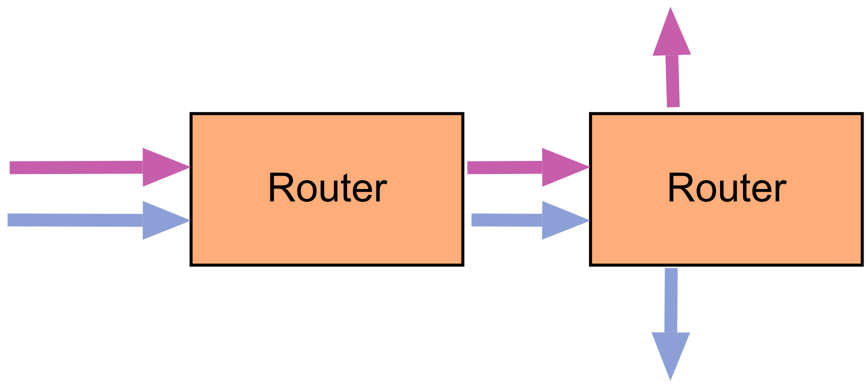
Link 1 (maximum bw: 100)



Is this an intrusive or non-intrusive UQ?

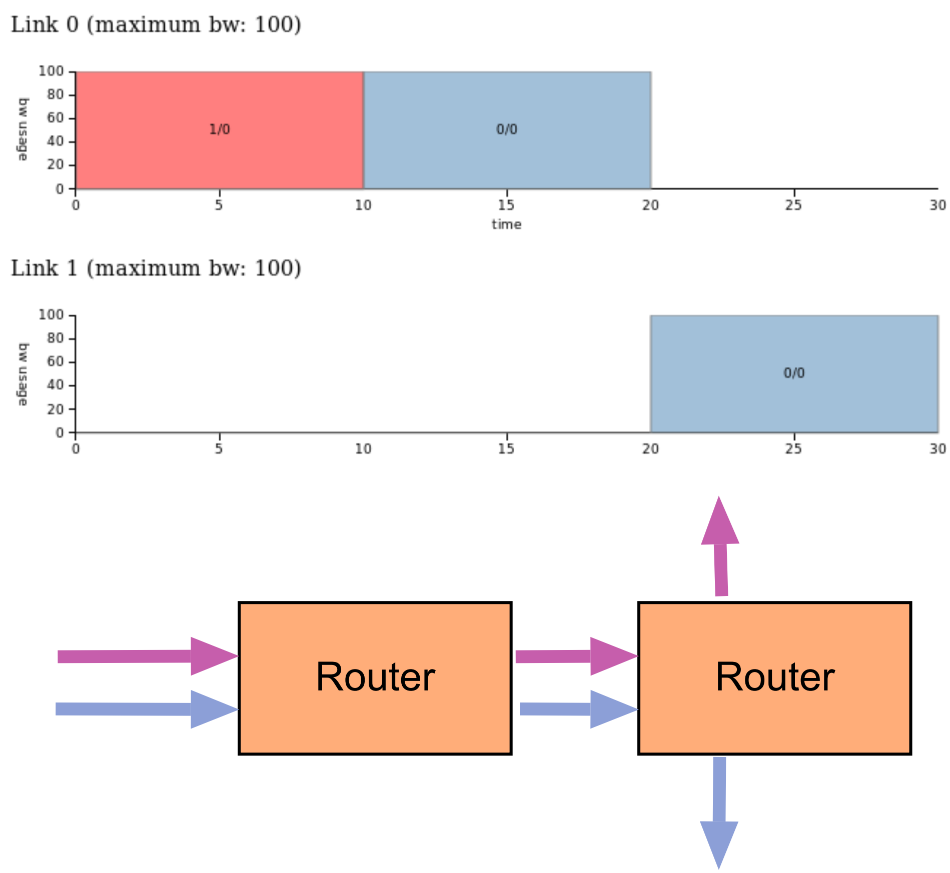
- Something must be logged for every packet event – intrusive
- Maintaining log of every event is too much data – need to reduce data into other metrics





Is this an intrusive or non-intrusive UQ?

- Results pending….



Addressing MODSIM questions

- Major contribution:
  - Math, workflow, AND toolkit
  - Demonstrating need for both intrusive and non-intrusive solutions
- Gaps:
  - Need bridge between mathematicians and programmers
  - Need to define both interchange formats for non-intrusive toolkits and APIs for intrusive libraries
- Bigger picture? Collaboration?
  - Simulators/modelers looking to bracket errors
    AND
  - UQ researchers with methods developed in other domains like PDEs
- How to leverage results?
  - Tutorials, not research papers
  - Know thine audience

UQ Toolkit: sandia.gov/uqtoolkit

C++ API for code integration

Python workflow integration