To roll the dice or not; Fusing statistical, probabilistic, and deterministic analysis tools

Donald Estep **Jeff Sandelin**

Department of Mathematics

Department of Statistics

Colorado State University

Research supported by

Department of Energy: DE-FG02-04ER25620, DE-FG02-05ER25699, DE-FC02-07ER54909;

Sandia Corporation:PO299784;

National Aeronautics and Space Administration: NNG04GH63G;

National Science Foundation: DMS-0107832, DMS-0715135, DGE-0221595003, ECCS-0700559, MSPA-CSE-0434354

US Department of Agriculture: 58-5402-3-306

Introduction

The Knee Problem

A deceptively simple problem:

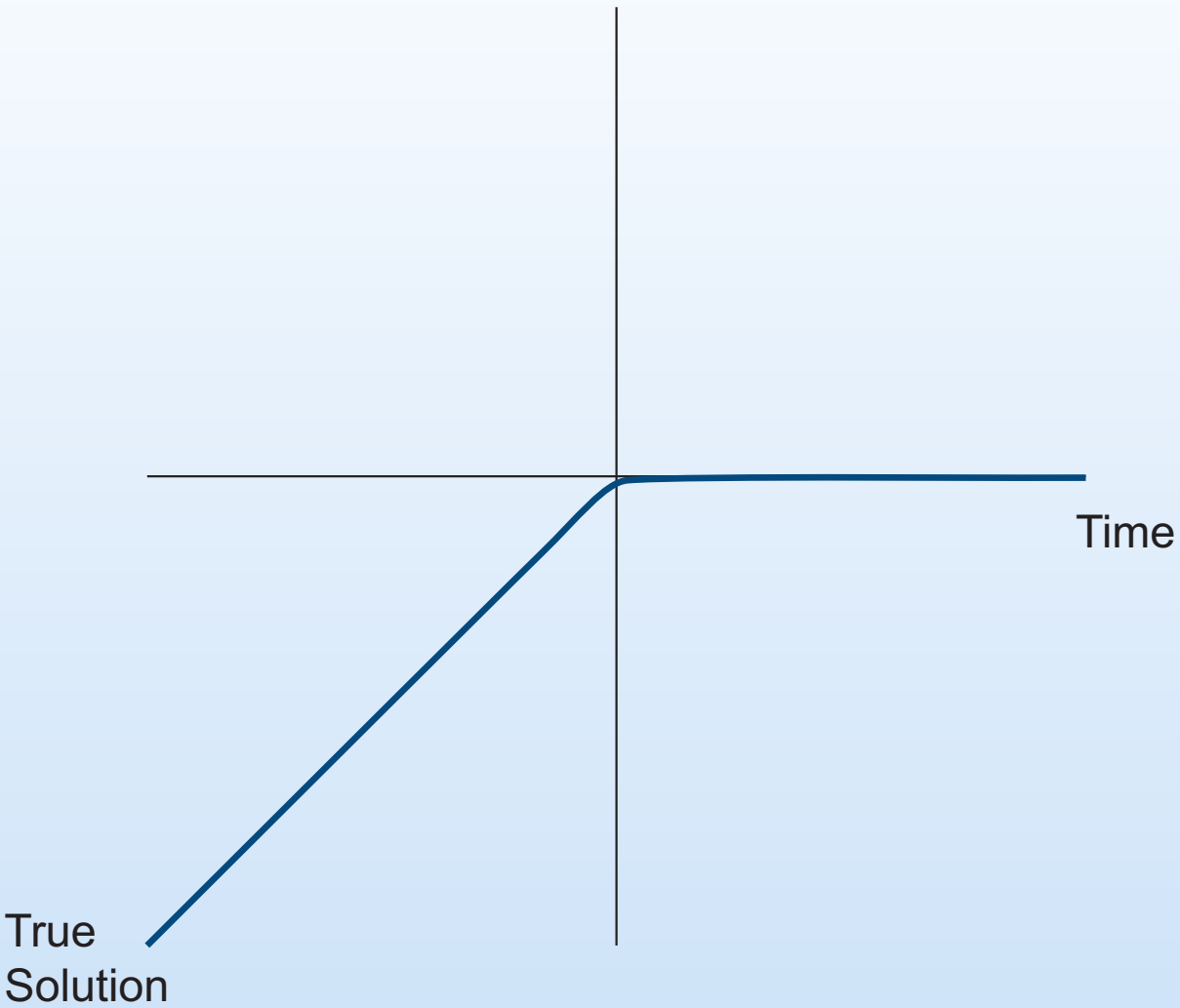
$$\begin{cases} \dot{y} = \frac{1}{\epsilon} y(y-t), & -1 < t \leq 1, \\ y(-1) = -1. \end{cases}$$

We compute with $\epsilon = 10^{-4}$

This causes all kinds of trouble for implicit numerical methods coupled with “local” time step control

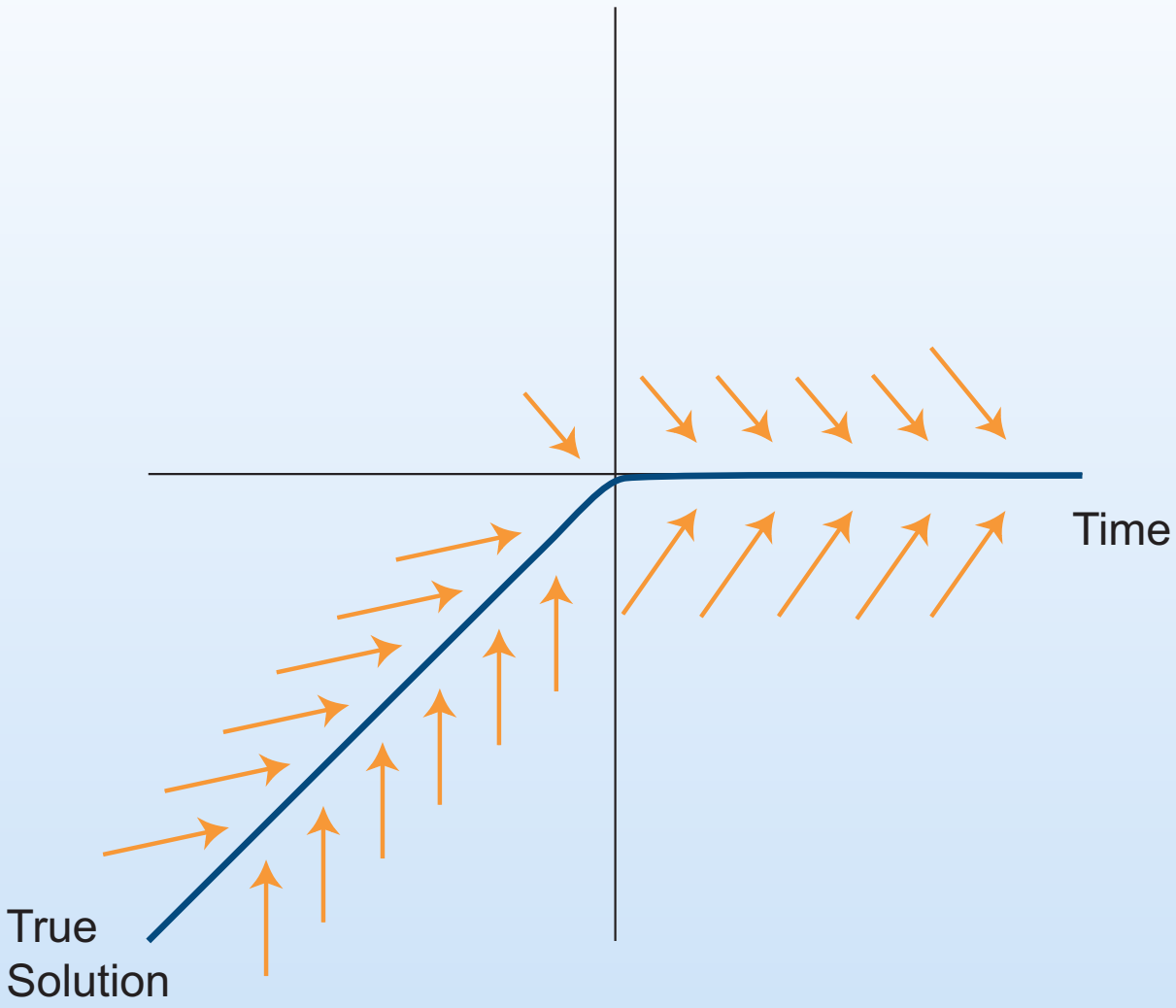
Reference: *Some stability aspects of schemes for the adaptive integration of stiff initial value problems*, L. Dieci and D. Estep, SISC 12 (1991), 1284-1303. That paper has earlier references.

The Knee Problem



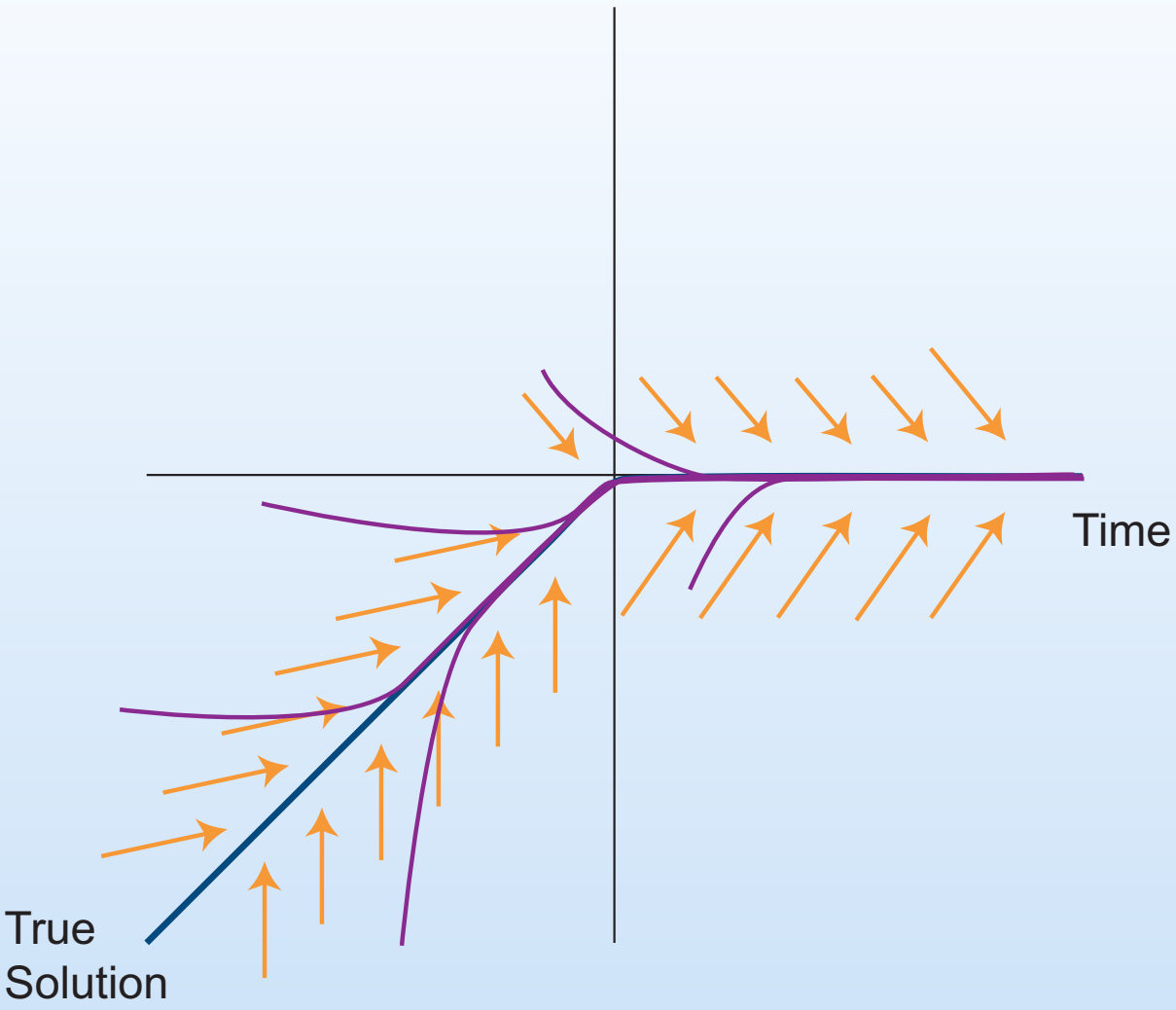
The solution we approximate is very smooth,
mainly linear in one region and constant in another

The Knee Problem



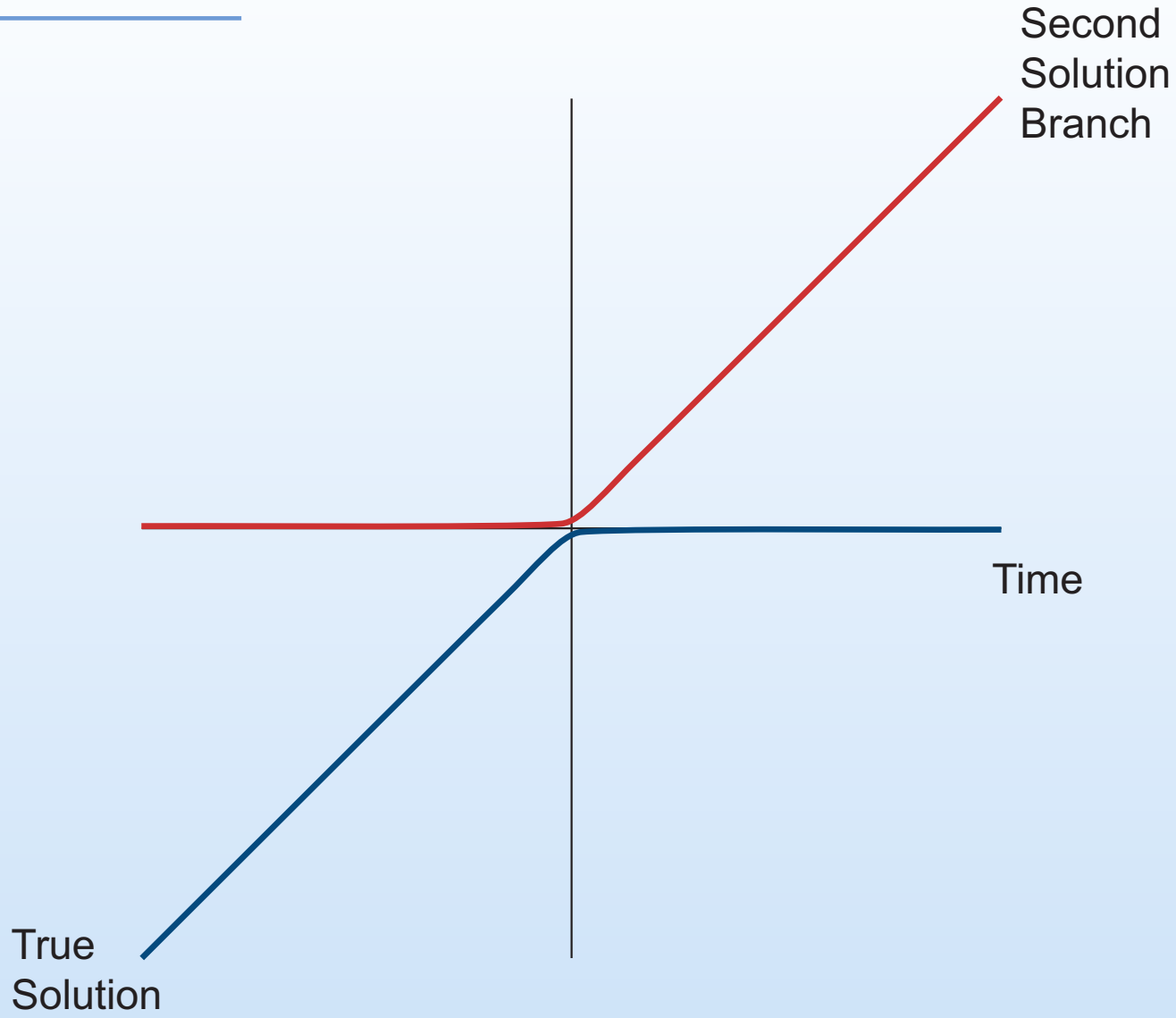
The true solution is extremely stable

The Knee Problem



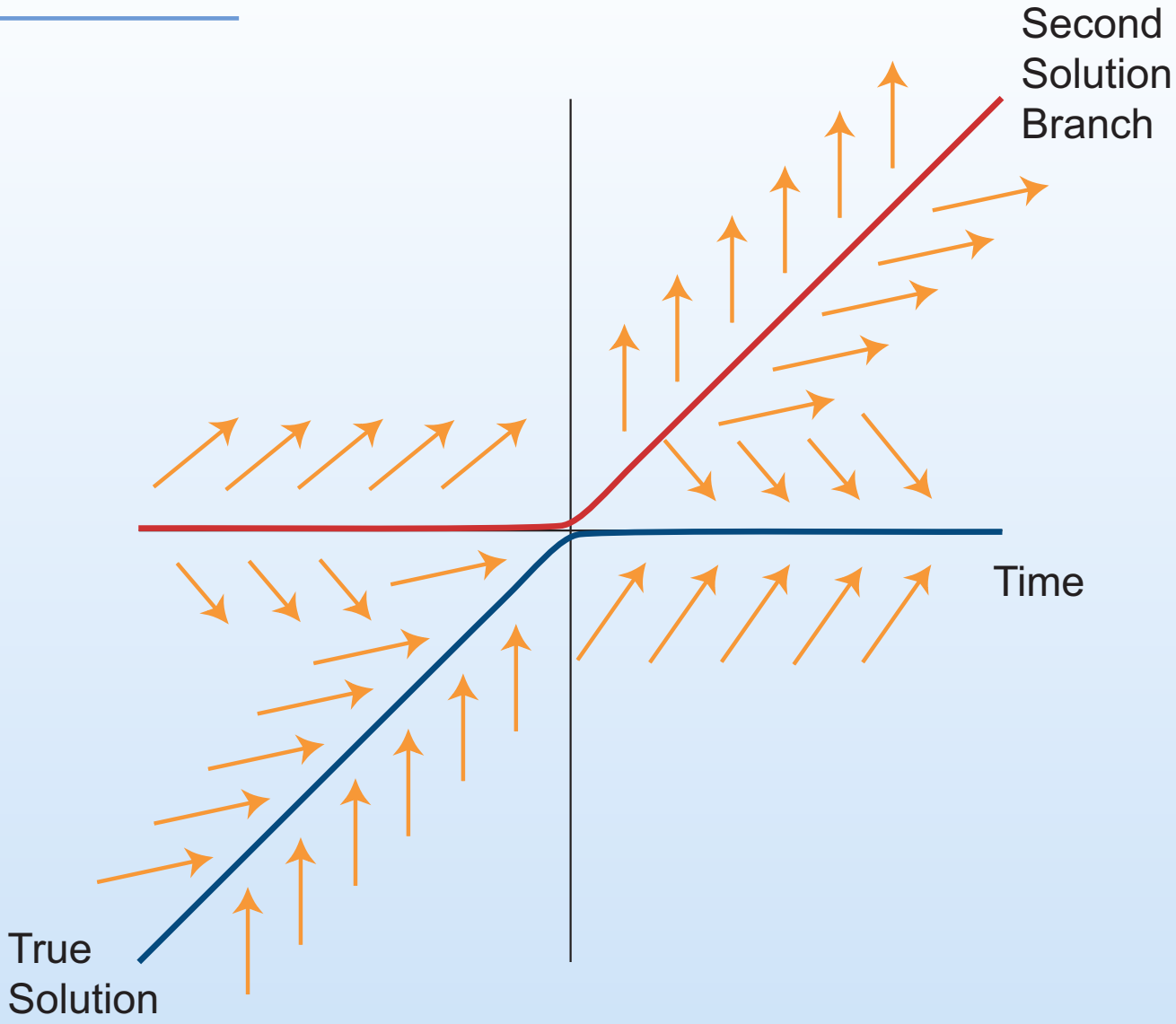
Nearby solutions converge rapidly to the stable solution

The Knee Problem



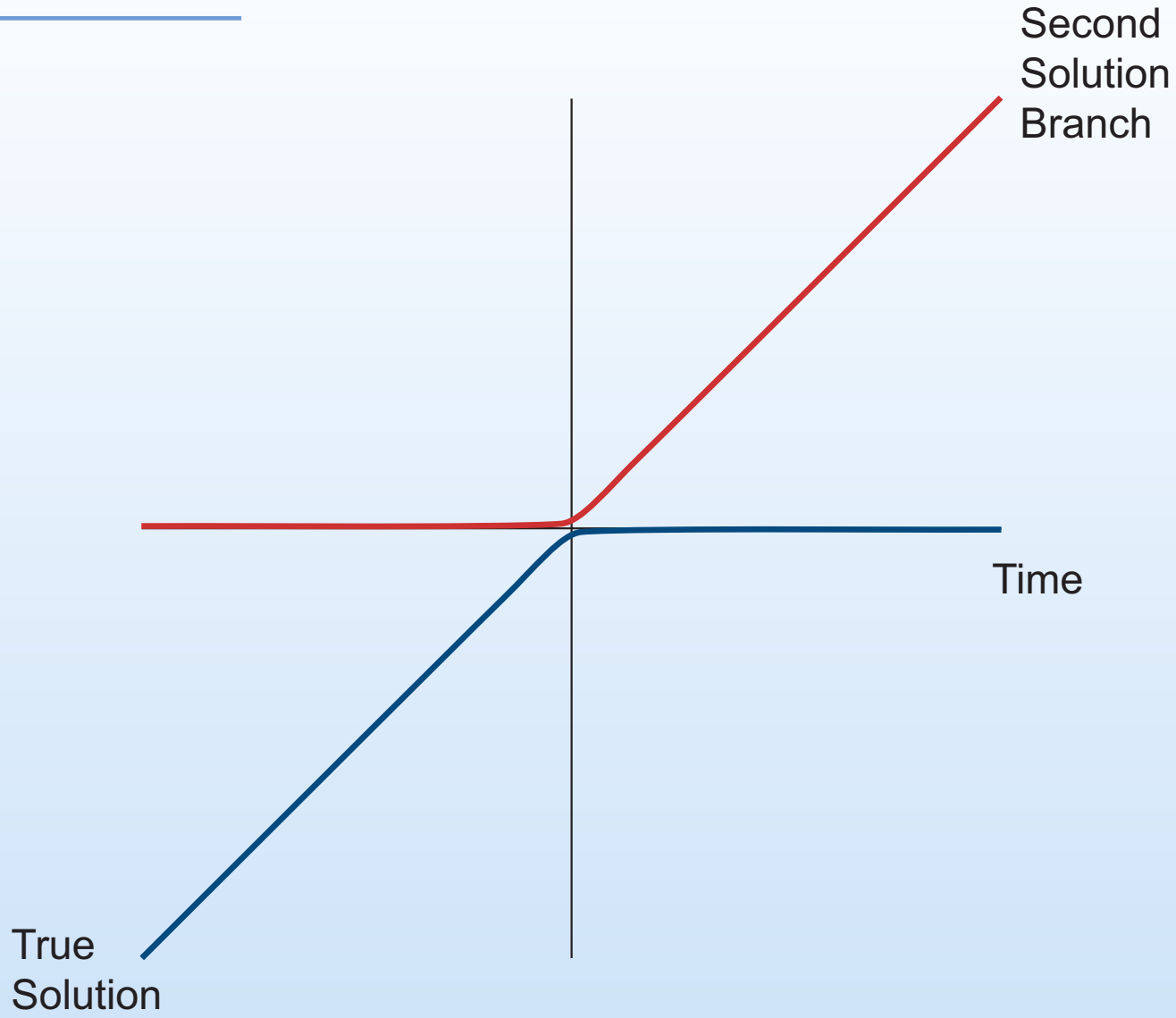
There is a second solution branch that is close near $t=0$

The Knee Problem



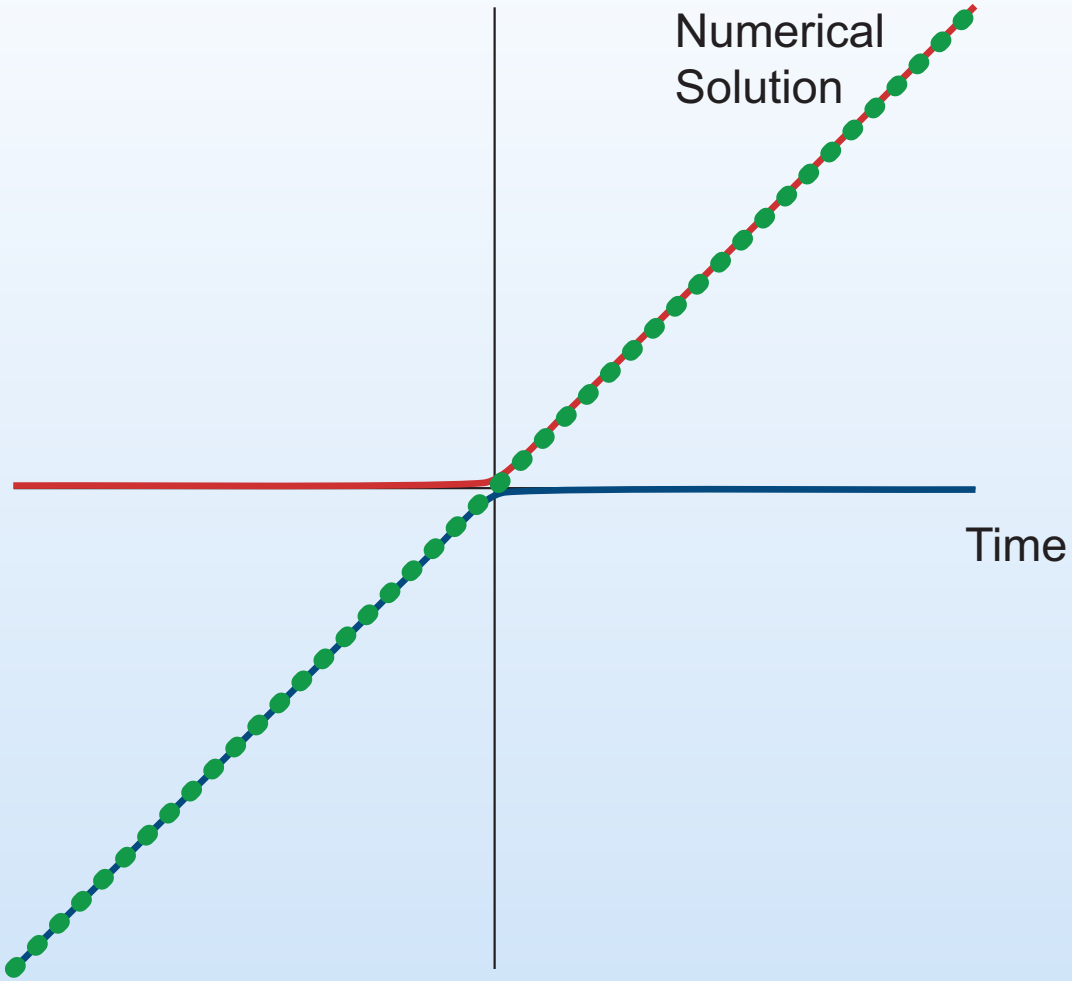
The second solution branch is very unstable

The Knee Problem



We use an implicit method using Newton's method
and so-called "local error" time step control

The Knee Problem



Such numerical solutions “hop” from the true solution branch to the **unstable** solution branch!

Lessons I Drew from the Knee Problem

- Controlling the “local error” or residual does not actually control the error in general
- Small numerical errors can lead to unpredictable behavior
- Using implicit methods does not guarantee the correct dynamical behavior
- The impracticality of enforcing assumptions for a priori convergence theorems often makes those results moot
- Computing stability information by solving the adjoint problem can lead to accurate error estimation

This led to my interest in adjoint-based a posteriori error analysis

Resistance to Error Estimation

In the nineties, the idea that accurate error estimates are useful met a lot of resistance

Arguments included:

- “There is no scientific need for accurate error estimates”
- “Computational error estimates are not real mathematics and adjoint analysis is not useful”
- “The cost associated with computing accurate estimates is prohibitive and such estimates will or can never be computed”

Of course, analogs of these arguments in the broader context of uncertainty quantification are still encountered today

Uncertainty, Verification, and Validation

Today, the climate for error estimation, uncertainty quantification, and verification and validation is much friendlier

The change is being driven by the needs of scientific disciplines spanning a diverse range from ecology to engineering dealing with multiphysics, multiscale systems

The goal has broadened to quantify all sources of uncertainty in the application of a mathematical model to describe and predict the behavior of a physical system

Sources include

- uncertainty in the model formulation
- uncertainty in numerical solution of the model
- uncertainty in data and parameters used in the model

Statistical, Probabilistic, and Deterministic Analysis

The main thesis of this talk is

> The problem of quantifying all sources of uncertainty in computational modeling demands a fusion of deterministic, probabilistic, and statistical approaches

In my experience, there are disciplinary and communication gulfs between mathematics, probability, and statistics that hampers progress in error estimation, UQ, and V&V

This gulf also hampers training of graduate students

Nonparametric Density Estimation

Probability Distributions

Let X be a random vector on a probability space $(\Omega, \mathcal{B}, P)$

The cumulative distribution function (c.d.f.) of X is the function

$$F_X(s) = P(X \leq s) \text{ (interpreted componentwise)}$$

If F_X is differentiable, then $\rho = \partial_{s_1} \ldots \partial_{s_d} F_X$ is the probability density function (p.d.f) and

$$\mu_X(A) = \int_A \rho \mathrm{d}\mu_L$$

for Borel sets A

We can determine the probability distribution of X by computing either the c.d.f. or the p.d.f.

Classic Statistical Problem: Density Estimation

General density estimation

Given a set of observations $\{X_i\}$ of a random variable X, determine the corresponding c.d.f or p.d.f. of X

Parametric density estimation

Assume X has one of the standard densities and estimate the parameters defining that density

Nonparametric density estimation

Let the data $\{X_i\}$ determined the density to be estimated, whatever it may be

The point of view of statistics is that the data $\{X_i\}$ is the only information known about X

Solving the Classic Density Estimation Problem

The classic nonparametric solution method is the histogram

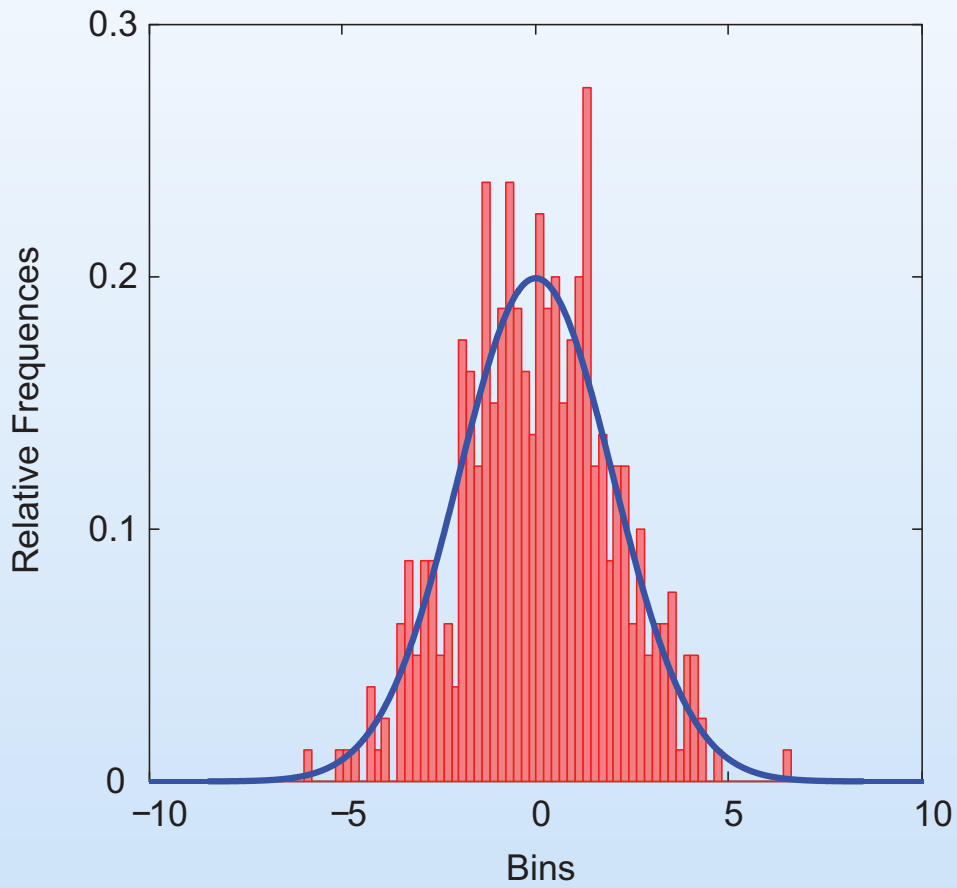
Choose an initial point x_0 and a bin size Δx and create bin nodes $x_i = x_0 + (i-1)\Delta x$, $i = 1, \cdots, N$

Given samples $\{X_i\}$, set

$$f_N(x) = \frac{\big(\text{number of } X_i \text{ in the same interval } (x_{j-1}, x_j) \text{ as } x\big)}{N\Delta x}$$

Solving the Classic Density Estimation Problem

The classic nonparametric solution method is the histogram



Histogram Computed From 400
Random Numbers with a Normal Distribution

Density Estimation for Differential Equations

The density estimation problem for a differential equation is significantly different

Suppose that y is the solution of a differential equation

$$\mathcal{L}(\dot{y}, y; \lambda_1) = 0, x \in \Omega,\ 0 < t,$$

together with specified initial and boundary data described by λ_0

$\lambda = (\lambda_0, \lambda_1)$ describes any data and parameters that might vary or are uncertain

Density estimation for a differential equation

Assuming λ is a random vector with a given distribution, compute the distribution of a functional of the solution,

$$q(\lambda) = q(Y(\lambda))$$

The Chaotic Lorenz Model

As an example, consider the Lorenz equations

$$\begin{cases} \dot{x}_1 = \sigma(x_2 - x_1), \\ \dot{x}_2 = rx_1 - x_2 - x_1x_3, \\ \dot{x}_3 = x_1x_2 - bx_3, \\ x_1(0) = a_1,\ x_2(0) = a_2,\ x_3(0) = a_3, \end{cases} \qquad t > 0,$$

$\lambda_0 = (a_1, a_2, a_3)$ and $\lambda_1 = (\sigma, r, b)$

Parameters might be distributed in a region around the critical values $\sigma = 10.1$, $r = 28.0$, and $b = 8/3$, and initial values distributed around some point

Interesting functionals include solution values, distances from solutions to a given fixed point, and the number of periods around a given fixed point

Estimating a Density for a Differential Equation

Abstract version

Y is a random vector determined by a smooth function $Y = G(\lambda)$ of a random vector λ with a specified probability distribution

We determine the probability distribution of a continuous functional $q(\lambda) = q(Y(\lambda))$ computed from Y

The computational question is: Can information about G be used to compute the distribution efficiently?

The solution operator G encapsulates the known information about processes in the model while λ encapsulates known information about the data for the model

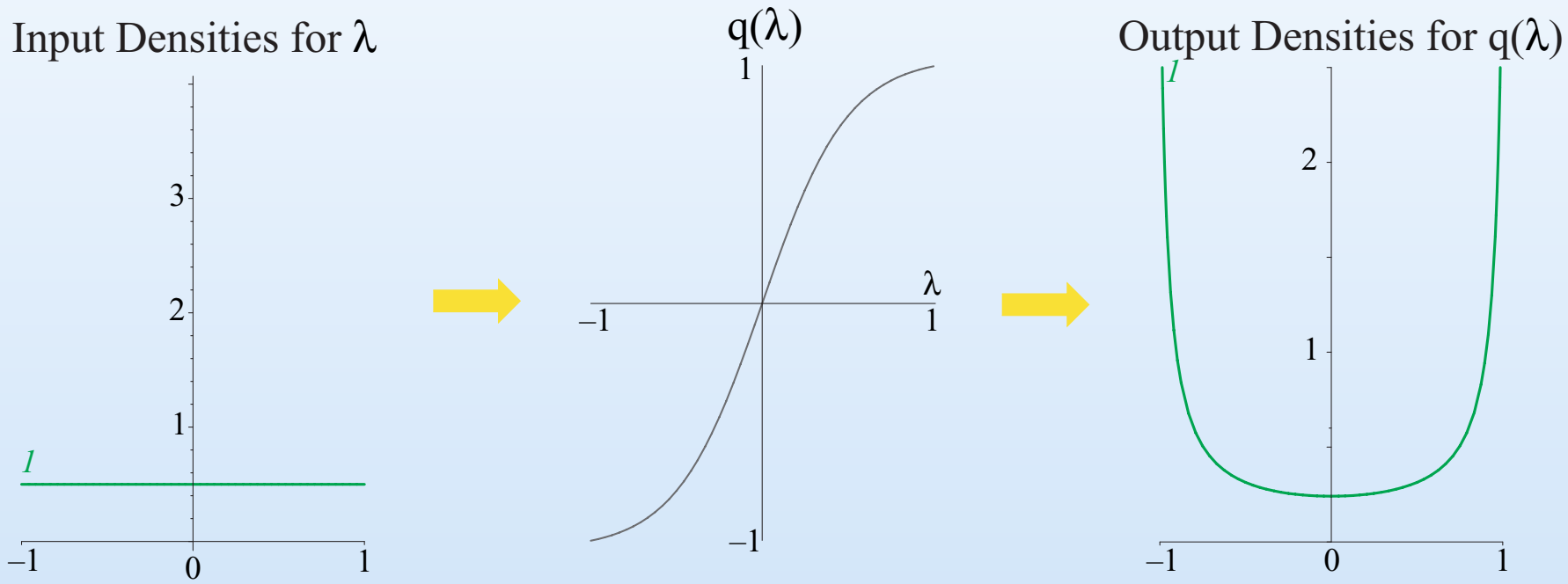
All Components are Important

The density of $q(Y(\lambda))$ is determined by the distribution of λ and the properties of G and q

Set $q(\lambda) = G(\lambda) = \tanh(2\lambda)/\tanh(2)$ with λ distributed on $[-1, 1]$

All Components are Important

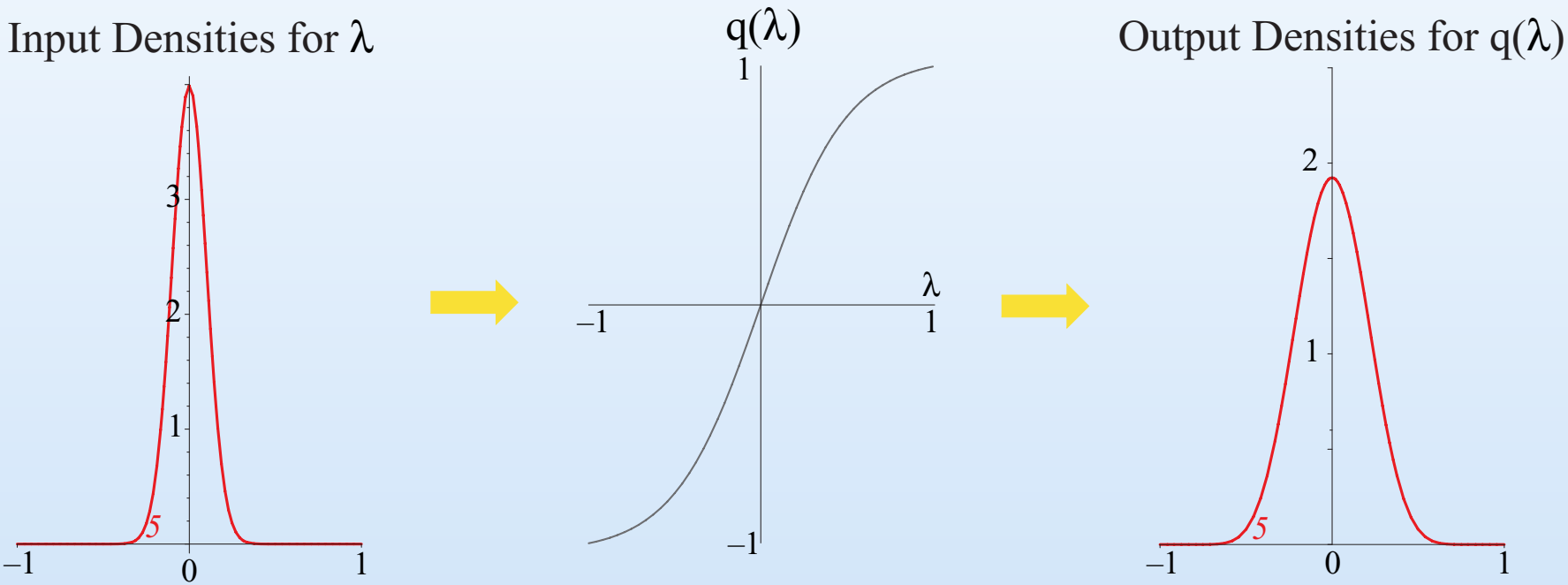
The density of $q(Y(\lambda))$ is determined by the distribution of λ and the properties of G and q



The nonlinearity of q significantly transforms a uniform distribution

All Components are Important

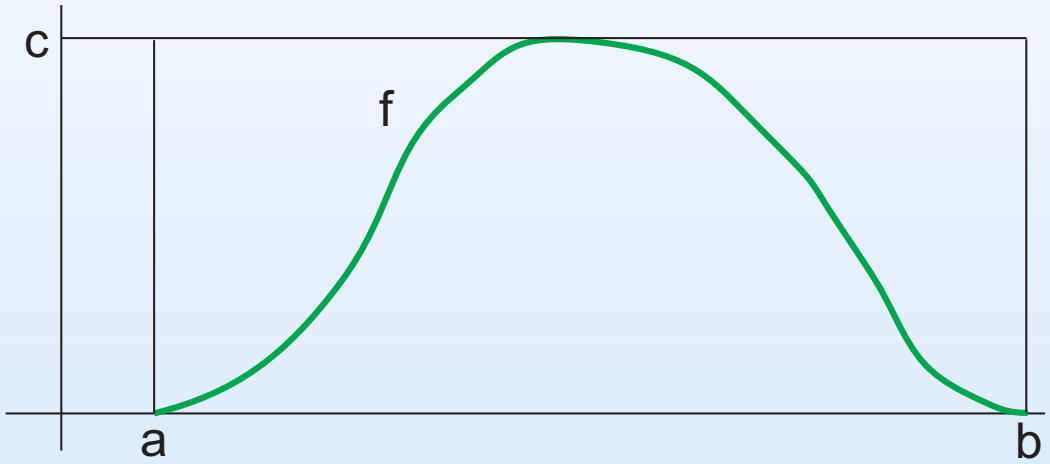
The density of $q(Y(\lambda))$ is determined by the distribution of λ and the properties of G and q



Inputting a normal distribution at a point where q behaves linearly results in a normal distribution on the output

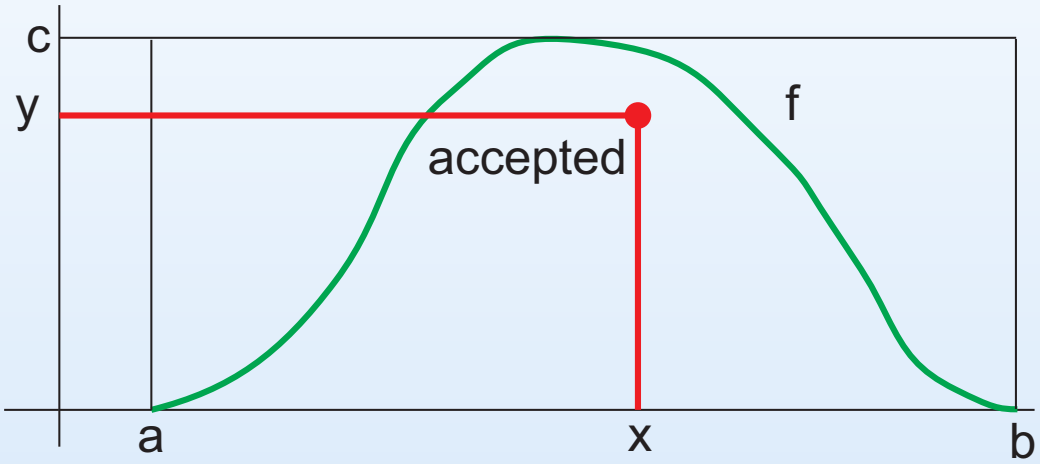
A Related Problem: Generating Random Numbers

A related statistical problem is generating random numbers from a nonstandard distribution f on $[a, b]$



A Related Problem: Generating Random Numbers

A related statistical problem is generating random numbers from a nonstandard distribution f on $[a, b]$

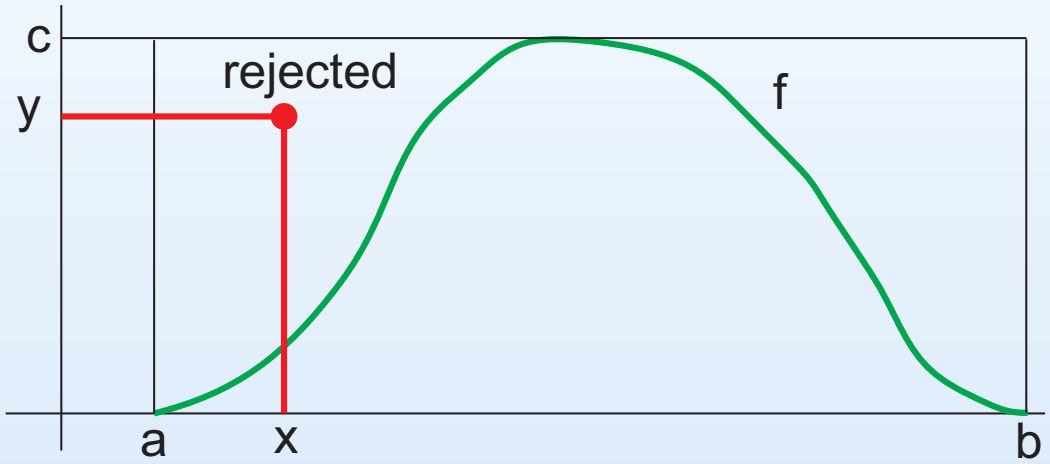


Accept/Reject Method:

Choose $X \sim U[a, b]$ and $Y \sim U[0, c]$
Accept (X, Y) if $Y \leq f(X)$, otherwise reject
A histogram computed from accepted samples approximates f

A Related Problem: Generating Random Numbers

A related statistical problem is generating random numbers from a nonstandard distribution f on $[a, b]$



Accept/Reject Method:

Choose $X \sim U[a, b]$ and $Y \sim U[0, c]$
Accept (X, Y) if $Y \leq f(X)$, otherwise reject
A histogram computed from accepted samples approximates f

A Related Problem: Generating Random Numbers

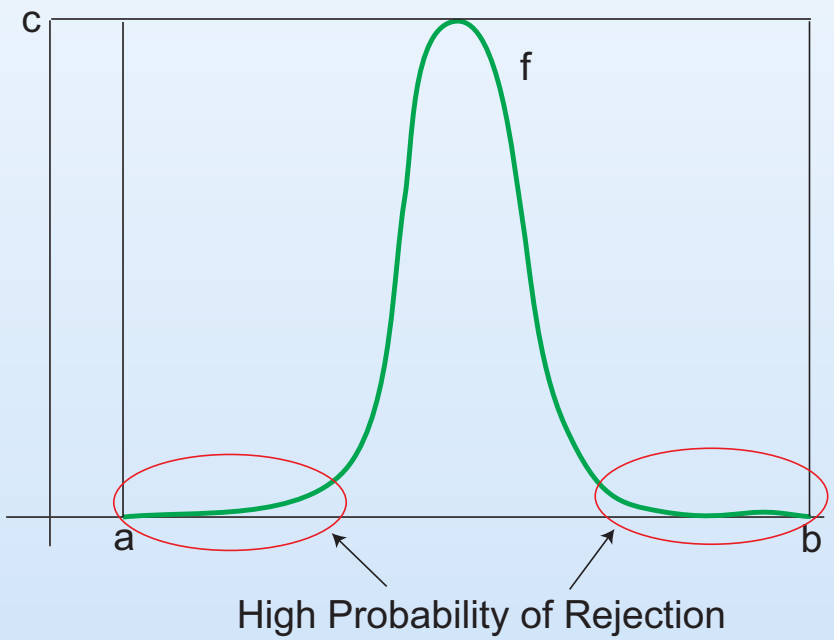
The Accept/Reject Method is powerful tool for stochastic simulation

Efficiency is an important issue however

A Related Problem: Generating Random Numbers

The Accept/Reject Method is powerful tool for stochastic simulation

Efficiency is an important issue however

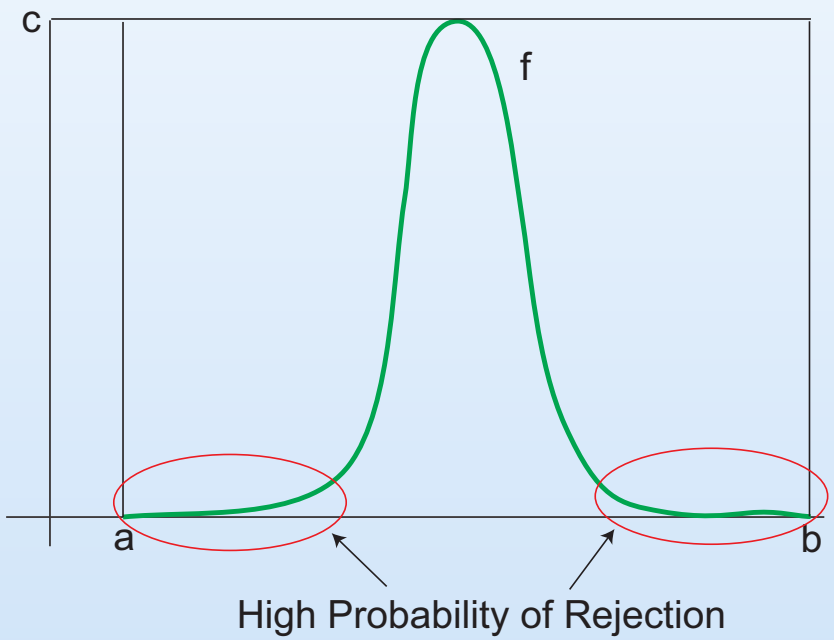


Accept/Reject becomes very inefficient when f leads to a high rate of rejection

A Related Problem: Generating Random Numbers

The Accept/Reject Method is powerful tool for stochastic simulation

Efficiency is an important issue however



Accept/Reject can be made much more efficient using knowledge of f

Density Estimation for an ODE

Consider the initial value problem

$$\begin{cases} \dot{y}(t;\lambda) = f(y(t;\lambda);\lambda_1), \quad t > 0, \\ y(0;\lambda) = \lambda_0 \end{cases}$$

with $y \in \mathbb{R}^n$ and $f : \mathbb{R}^{n+d} \to \mathbb{R}^n$, $\lambda = (\lambda_1, \lambda_0)^\top \in \mathbb{R}^{d+n}$

The quantity of interest is a linear functional

$$q(\omega) = q(\lambda(\omega)) = \int_0^T \langle x(t;\lambda(\omega)), \psi(s) \rangle \, \mathrm{d}s,$$

where $\langle \cdot, \cdot \rangle$ is the $\mathbb{R}^n$ inner product

The Adjoint and the Generalized Green's Function

We assume that λ is a random variable with a specified distribution

We describe the effect of varying the parameter λ around a reference parameter value $\mu = (\mu_1, \mu_0)^\top$

The generalized Green's function solves the adjoint problem,

$$\begin{cases} -\dot{\phi}(t) - A^\top(t)\phi(t) = \psi(t), & T \geq t \geq 0, \\ \phi(T) = 0, \end{cases}$$

$$A(t) \equiv D_x f(\tilde{y}(t); \mu_1)$$

$\tilde{y}(t)$ denotes the deterministic solution at the reference value

Representing the Variation Using the Adjoint Solution

A variational argument gives

Theorem

$$q(\lambda) = \int_0^T \langle y, \psi \rangle \, \mathrm{d}s \approx \int_0^T \langle \tilde{y}, \psi \rangle \, \mathrm{d}s + \langle \lambda_0 - \mu_0, \phi(0) \rangle$$
$$+ \int_0^T \langle D_{\lambda_1} f(\tilde{y}; \mu_1)(\lambda_1 - \mu_1), \phi \rangle \, \mathrm{d}s$$

The last term on the right describes the effect of variations in the model parameters

The second to last term describes the effect of variations in the initial conditions

The result extends to PDEs

The Point Is ...?

The theorem says that

$$q(\lambda) \approx q(\mu) + \nabla q(\mu)(\lambda - \mu)$$

where the adjoint problem provides a cheap way to compute derivatives of the quantity of interest

This uses the special nature of the density estimation problem for differential equations

There are now multiple ways to use this gradient information to speed up the computation of the density

We discuss one approach

Piecewise Representations

To obtain globally accurate approximations:

We choose a sample $\{\mu_i\}_{i=1}^N$ and a partition of subregions $\{R_i\}_{i=1}^N$ with $\mu_i \in R_i$ for all i

We construct a piecewise constant approximation

$$\tilde{q}(\lambda) = \sum_{i=1}^{N} q(\mu_i)\chi_{R_i}(\lambda),$$

where χ_{R_i} is 1 for $\lambda \in R_i$ and 0 otherwise

On R_i, we estimate the error as

$$q(\lambda) - q(\mu_i) \approx \nabla q(\mu_i)(\lambda - \mu_i)$$

Fast Adaptive Parameter Sampling

We use the error estimates to guide adaptive sampling

The goal is to achieve the desired accuracy using a minimal number of sample points

The refinement process works iteratively:

- For a given sample, we estimate the local contributions to the error using the estimate
- We add new sample points in regions that contribute the most to the error

We only add sampling points in the direction of the largest component of ∇q

The Schaefer Model

The Schaefer Model for harvesting of a fish population x is

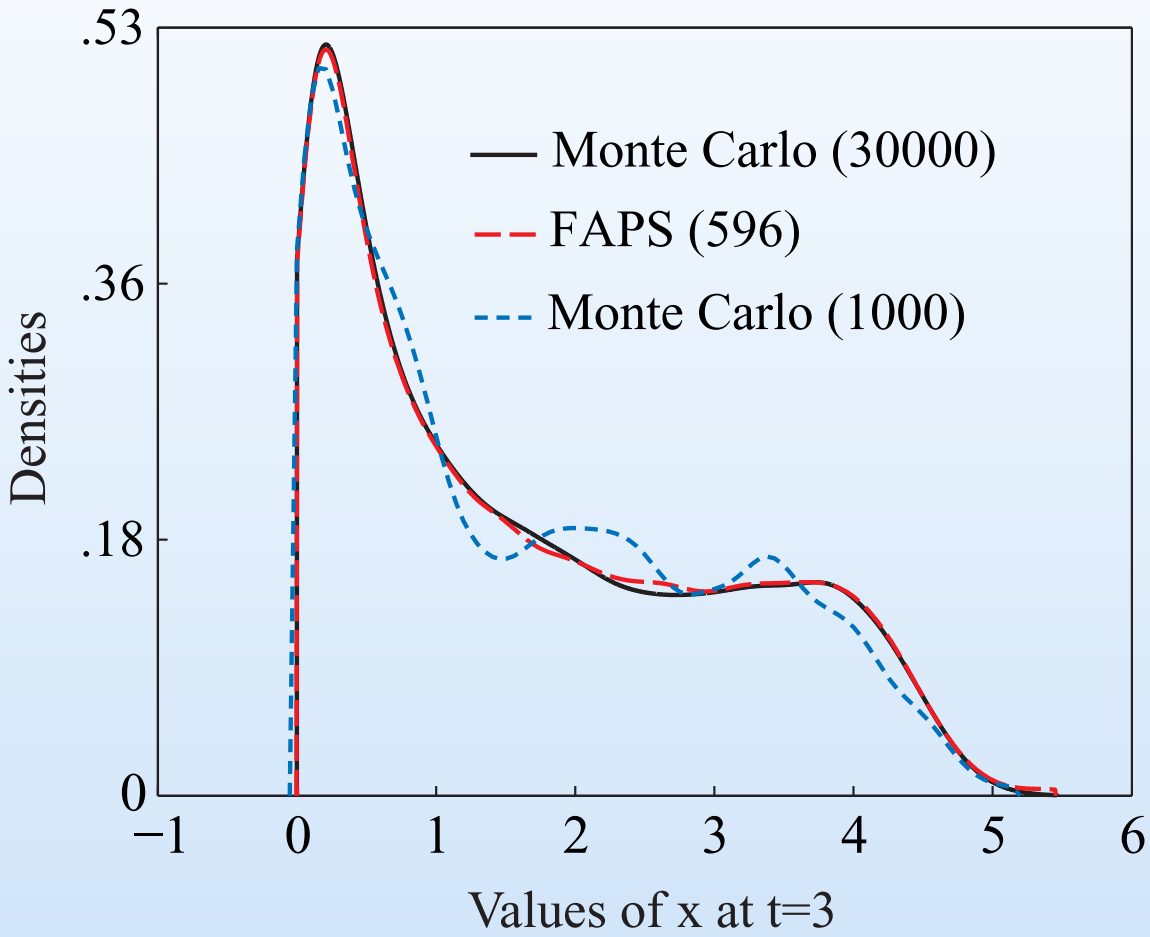
$$\begin{cases} \dot{x}(t;\lambda_1) = \frac{R}{K}\left(K - x(t;\lambda_1)\right)x(t;\lambda_1) - Hx(t;\lambda_1), \quad t > 0, \\ x(0;\lambda_1) = x_0, \end{cases}$$

where $\lambda_1 = (R, K, H)^\top$

We take data $\psi(s) = \delta(s-3)$ and $x_0 = 1$

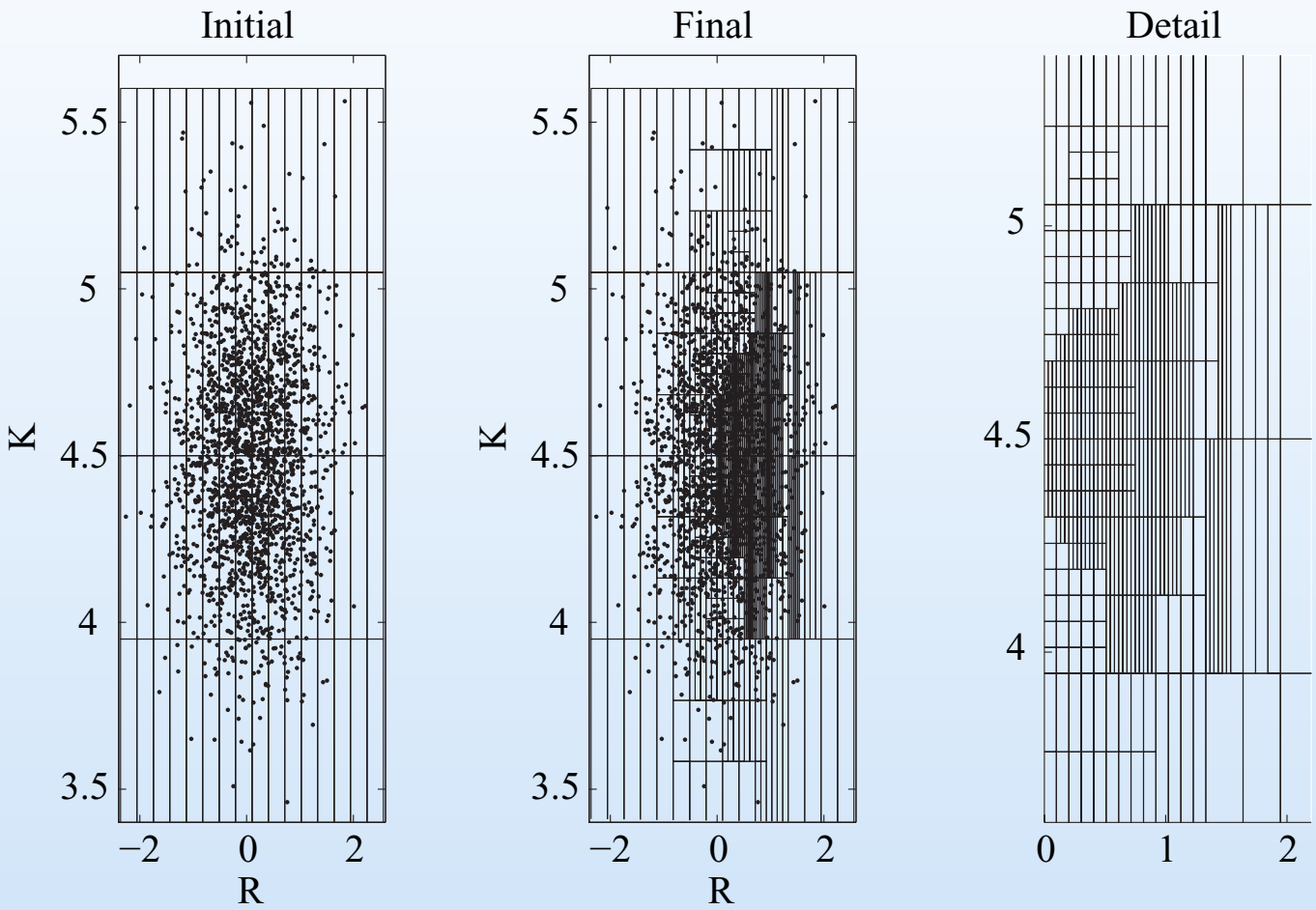
We assume that $H = 0$ and (R, K) are independent and $N(0.1, 0.5)$, $N(4.5, 0.1)$ (truncated) respectively

Schaefer Model: FAPS Results for $H = 0$



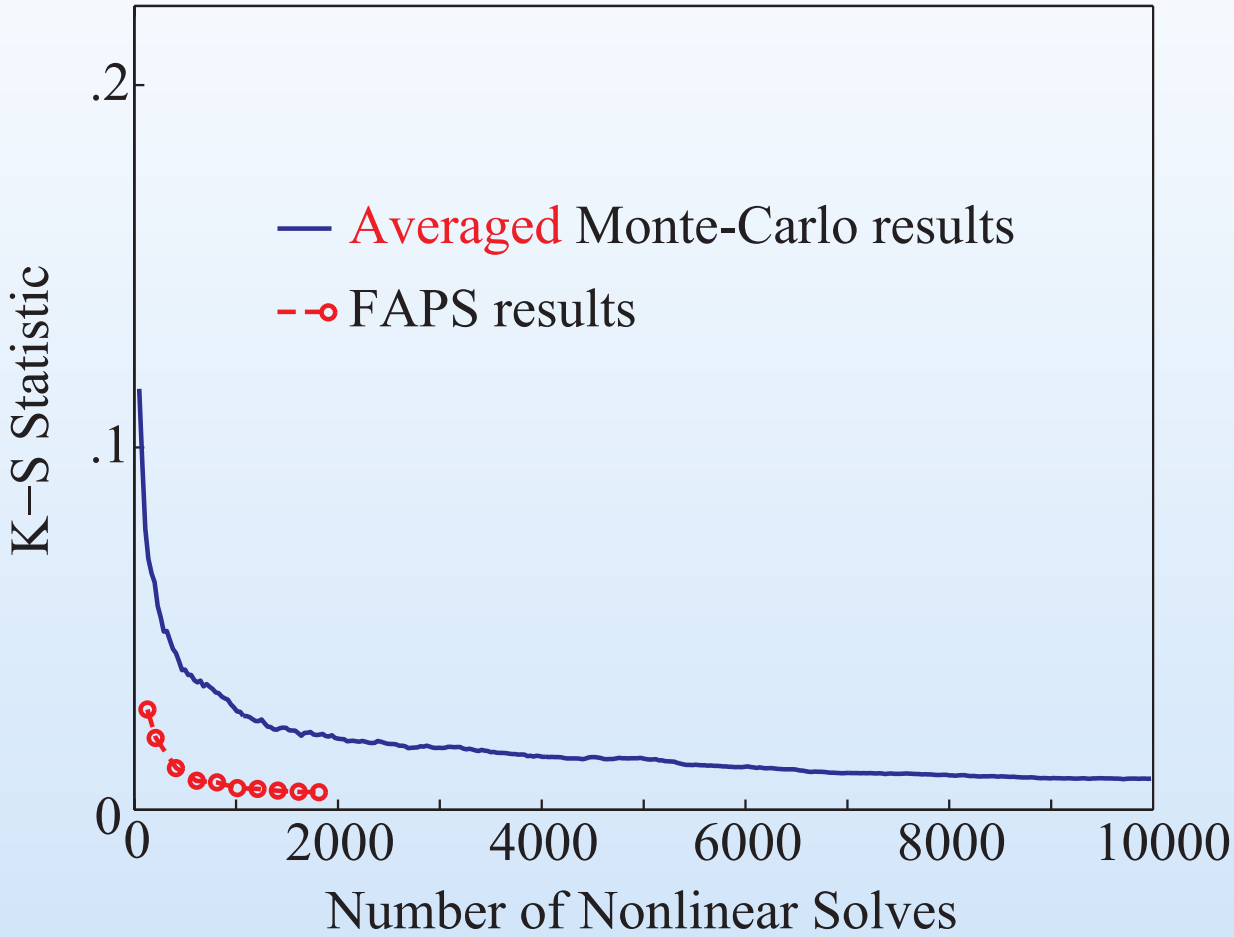
We compare a FAPS-computed density to two Monte-Carlo samples (all smoothed with a Gaussian kernel).

Schaefer Model: FAPS Meshes for $H = 0$



This shows the sampling grid for FAPS along with the points used for the 1000 point Monte-Carlo density. The center of each rectangle is a sample point in the FAPS density. Note the pattern of refinement.

Schaefer Model: Gain in Efficiency using FAPS



We use the standard K-S statistic to compare rates of convergence. To quantify a rate for the Monte-Carlo method, we use an expected rate computed using a number of Monte-Carlo sequences.

A Predator Prey Example

We model a prey u with a logistic birth/death process consumed by predator v

$$\begin{cases} \partial_t v - \delta \Delta v = \lambda_1 v\, h(u;\lambda_2) - \lambda_3 v, & \Omega \times (0,T], \\ \partial_t u - \delta \Delta u = \lambda_4 u(1 - \frac{u}{\lambda_5}) - \lambda_6 v\, h(u;\lambda_2), & \\ \partial_n v = \partial_n u = 0, & \partial\Omega \times (0,T], \\ v = v_0, u = u_0, & \Omega \times \{0\} \end{cases}$$

The (Holling II) functional response $h(u) = h(u;\lambda_2)$ satisfies

- $h(0) = 0$
- $\lim_{x\to\infty} h(x) = 1$
- h is strictly increasing

Predator Prey Reference Parameter Values

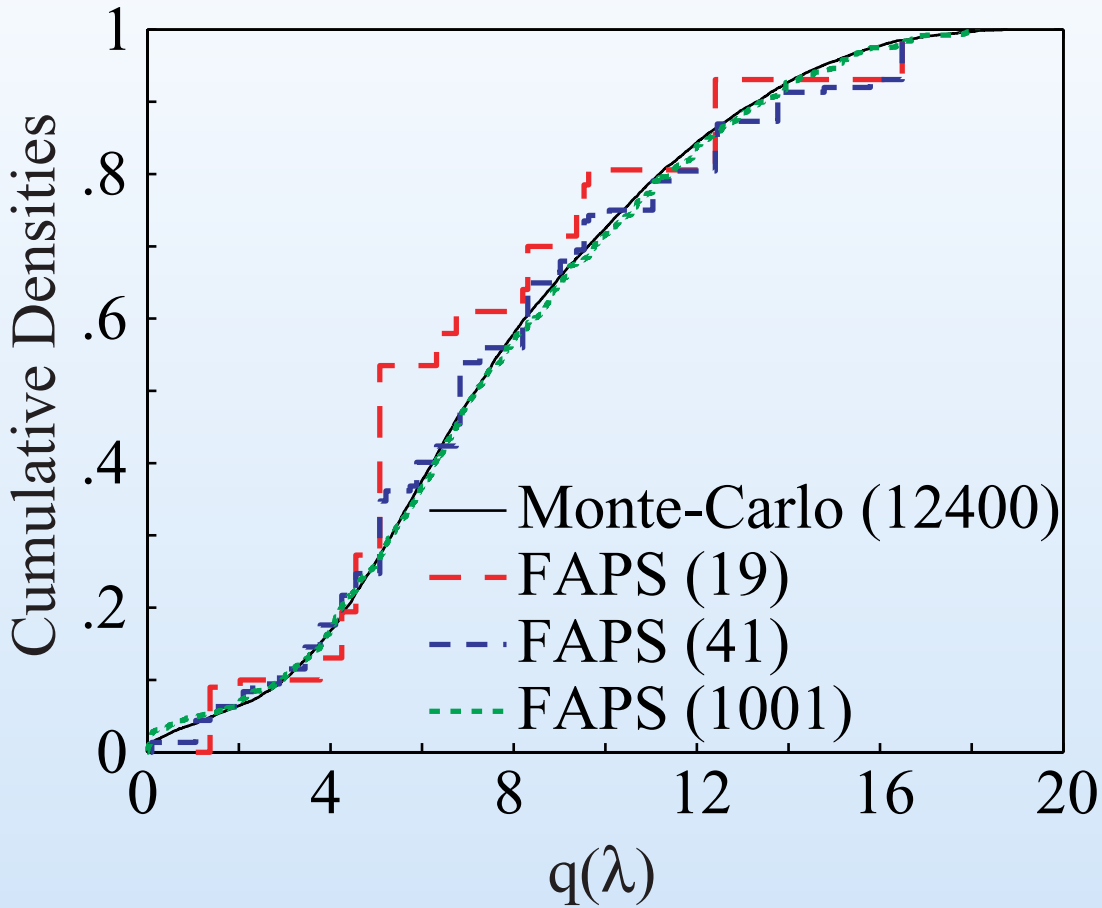
We assume a (truncated) normal distribution in the region

Description	Name	Mean	Perturbation
encounter gain	μ_1	1	$\pm 50\%$
response gain	μ_2	10.1	$\pm 50\%$
predator death rate	μ_3	1	$\pm 50\%$
prey growth rate	μ_4	5	$\pm 50\%$
prey carrying capacity	μ_5	1	$\pm 50\%$
encounter loss	μ_6	1	$\pm 50\%$

We use the L^1 norm of the prey population at $t = 10$ as the quantity of interest $(\psi = \delta(t - 10)(0, 1)^\top)$

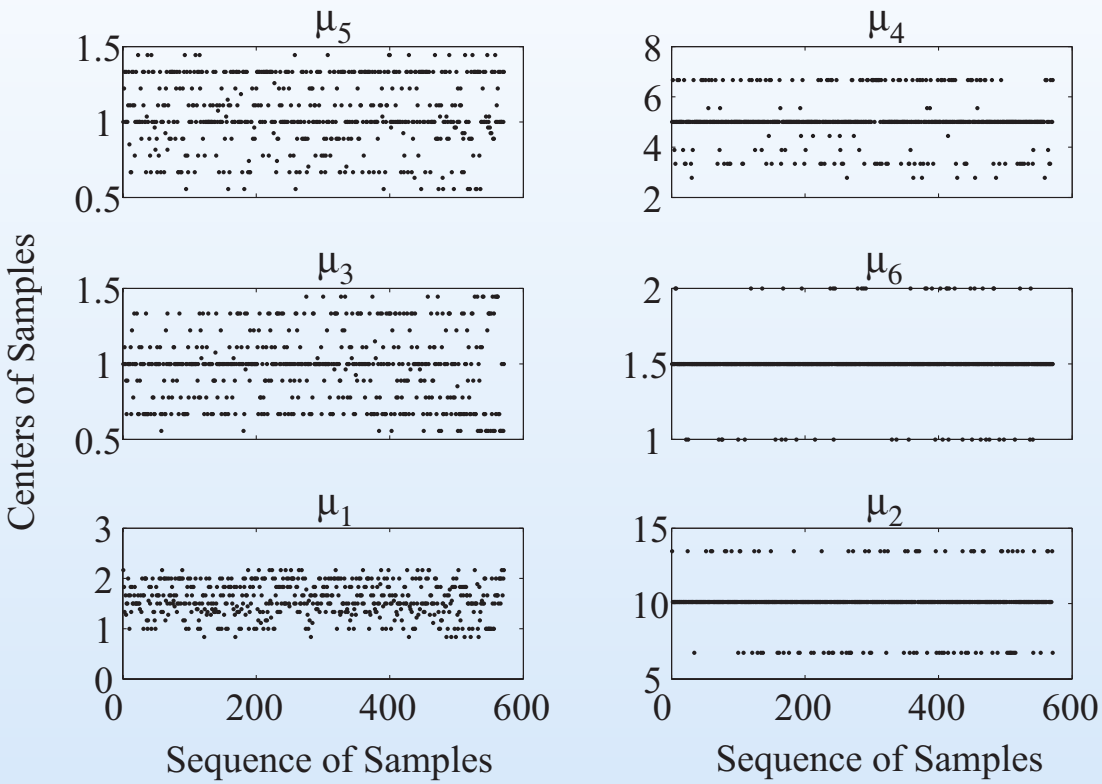
We use a 12400 point Monte-Carlo computation as a reference

Predator Prey Results (Cumulative Density)



We compare the c.d.f. of three FAPS computations to a Monte-Carlo computation with 12,400 points. Each “point” requires solving the PDE.

Predator Prey Results (Dimension Reduction)



These are the FAPS sample points for each parameter. Sample points are added only in directions in which the gradient components are largest, which provides localized dimension reduction. Here, FAPS only samples μ_6 at the midpoint of its range. If we compare to a computation with forced sampling in μ_6, we obtain the same densities.

Controlling the Numerical Error in Density Estimation

Solving the kernel density estimation problem requires solving the problem for a variety of data and parameter values

The corresponding solutions can exhibit a variety of behaviors

It is important to control the numerical errors so that they do not bias the analysis results

Controlling the Numerical Error in Density Estimation

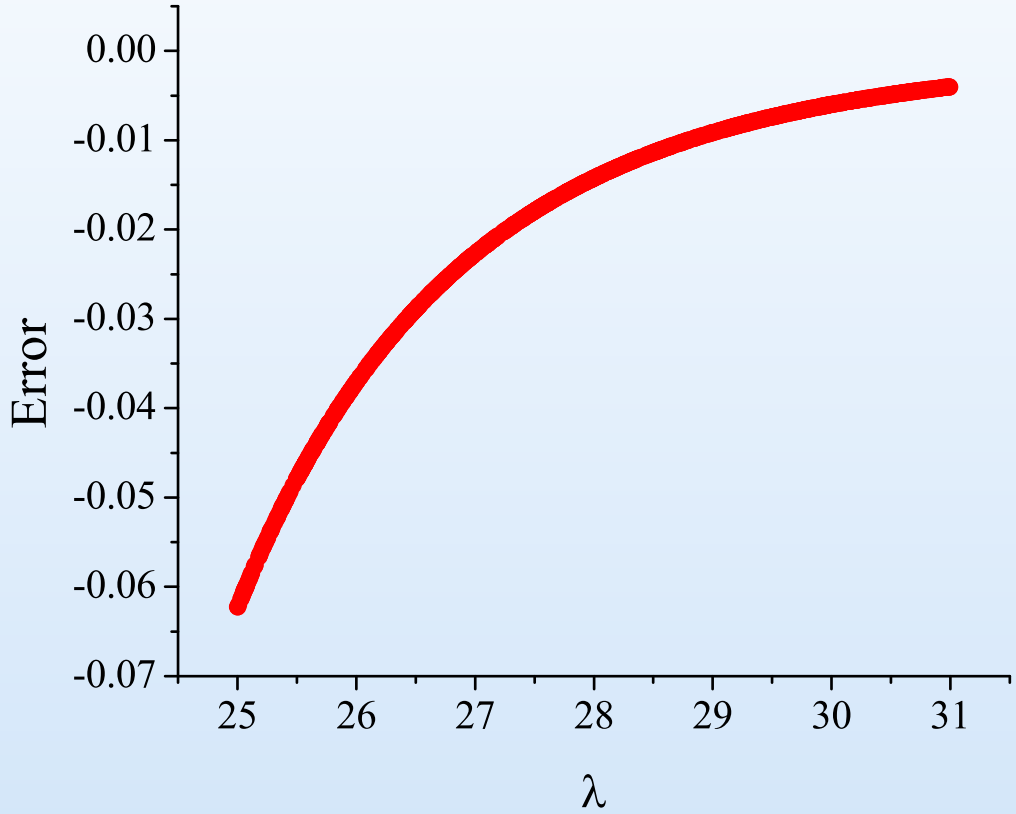
Example We consider the chaotic Lorenz problem

$$\begin{cases} \dot{u}_1 = -10u_1 + 10u_2, \\ \dot{u}_2 = \lambda u_1 - u_2 - u_1 u_3, \\ \dot{u}_3 = -\frac{8}{3}u_3 + u_1 u_2, \\ u_1(0) = -6.9742, u_2(0) = -7.008, u_3(0) = 25.1377 \end{cases} \qquad 0 < t,$$

We vary $\lambda \sim$ Unif $[25, 31]$

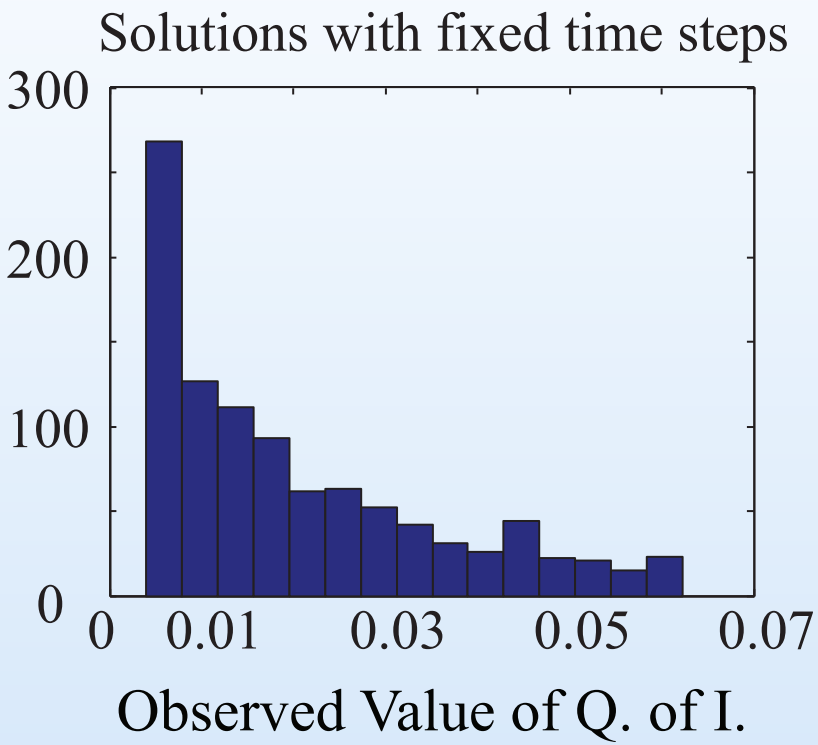
We use 1000 point Monte-Carlo sampling with both a fixed time step computation and an adaptive computation with error smaller than 10^{-5}

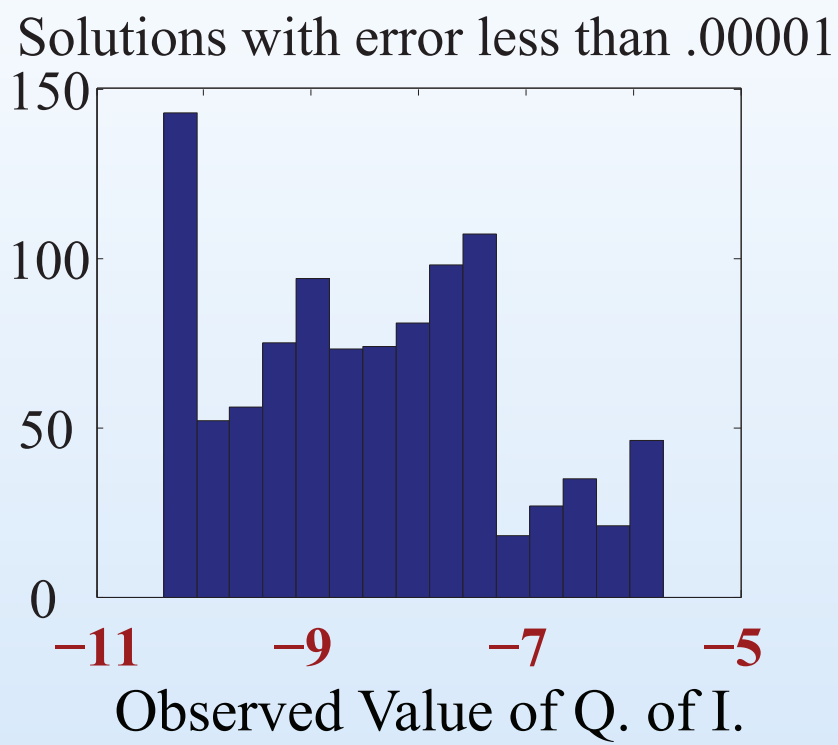
Controlling the Numerical Error in Density Estimation



Numerical error versus parameter for a fixed time step

Controlling the Numerical Error in Density Estimation





Distributions for computations with fixed and adaptive time steps. Note the difference in the horizontal scales!

Applying Probability to Adaptive Error Control

A Posteriori Analysis and Adaptive Error Control

Computing accurate error estimates provides the tantalizing idea of optimizing discretizations

Unfortunately, cancellation of errors complicates the possibility of defining, let alone computing, an optimal discretization

The existing theory is based on optimal control of error bounds that ignores cancellation of error

Probabilistic sampling may provide an alternative approach

The Optimization Approach to Adaptive Error Control

Given a tolerance TOL, an adjoint-based *a posteriori* error estimate calls for refinement when

$$\text{Error in a Q. of I.} \approx \left|\left(Residual, Adjoint\ Weight\right)\right| \geq \text{TOL}$$

Refinement decisions are based on a bound consisting of a sum of element contributions

$$\text{Error in a Q. of I.} \leq \sum_{elements\ K} \left|\left(Residual, Adjoint\ Weight\right)_K\right|$$

where $(\ ,\ )_K$ is the inner product on K

The element contributions in the bound do not cancel

Optimization Approach to Adaptive Error Control

There is no cancellation of errors across elements in the bound, so optimization theory yields

Principle of Equidistribution: The “optimal” discretization is one in which the element contributions are equal

The adaptive strategy is to refine some of the elements with the largest element contributions

In the adjoint weighted residual approach, the element residuals are scaled by an adjoint weight, which reflects how much error in that element affects the error in the quantity of interest

A more classic bound has a sum of norms of element residuals and stability information is taken out as a large constant in front of the sum

A Probability Approach to Adaptive Error Control

We first decompose the error in the quantity of interest into positive and negative contributions

$$\text{Error in a Q. of I.} \approx \sum_{\substack{positive \\ contributions}} \text{elt. contrib.} + \sum_{\substack{negative \\ contributions}} \text{elt. contrib.}$$

$$= E^+ - E^-$$

We apportion the number of elements N to be refined as

$$N^+ = N\frac{E^+}{E^+ + E^-}, \quad N^- = N\frac{E^-}{E^+ + E^-}$$

The idea is to balance the positive and negative contributions so they cancel to reach the tolerance

A Probability Approach to Adaptive Error Control

For $E^{\pm}$, we define a probability mass function

$$p_i^{\pm} = \frac{\left|\text{i}^{\text{th}}\text{element contribution}\right|}{\text{sum of element contributions}}$$

We select elements for refinement randomly according to $p^{\pm}$

Elements with the largest contributions are favored for refinement, but refinement can occur anywhere

For many quantities of interest, this approach results in significantly fewer steps than traditional approaches

We may also sample to reduce the variation of element contributions, increasing reliability on coarse discretizations

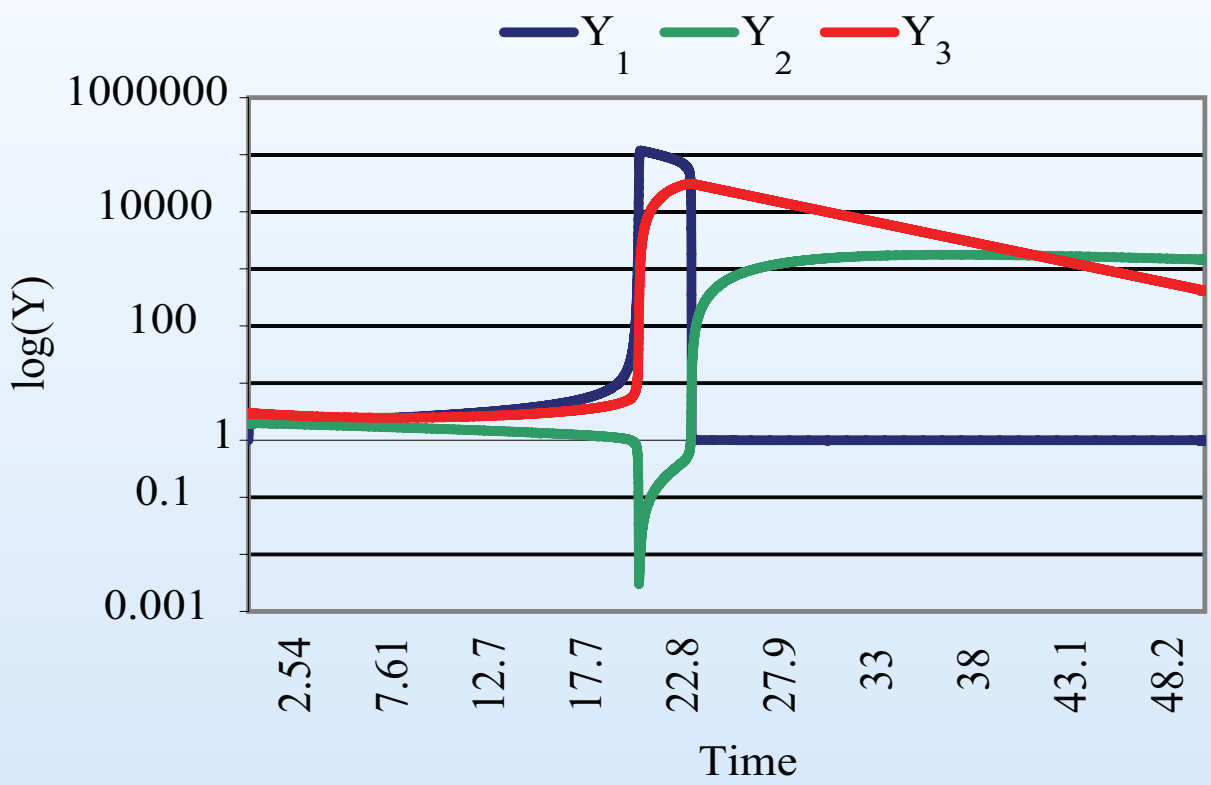
Probabilistic Adaptivity for the Oregonator

We consider the Oregonator problem

$$\begin{cases} \dot{y}_1 = 2(y_2 - y_1 y_2 + y_1 - q y_1^2) & y_1(0) = 1 \\ \dot{y}_2 = \frac{1}{s}(-y_2 - y_1 y_2 + y_3), & y_2(0) = 0, \\ \dot{y}_3 = w(y_1 - y_3), & y_3(0) = 0, \\ s = 77.27,\ w = .161,\ q = 8.375 \times 10^{-6} \end{cases}$$

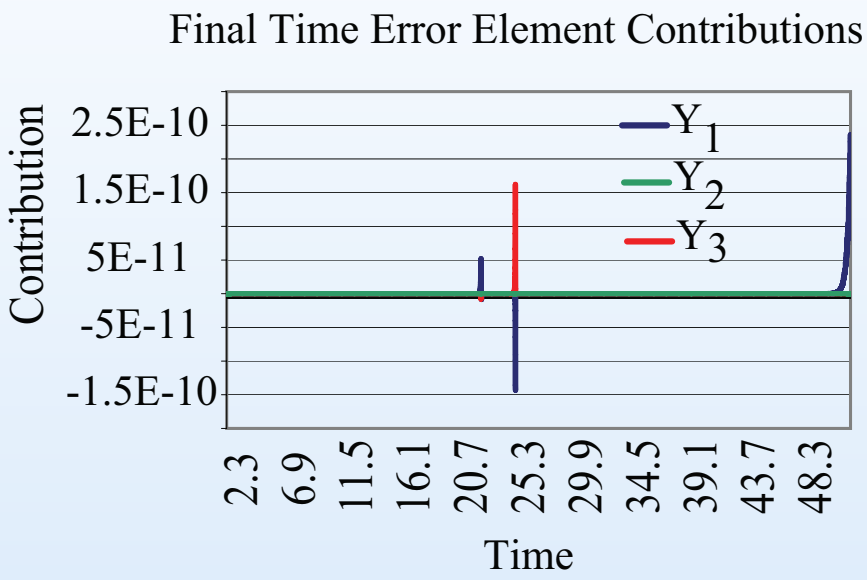
We compute with $TOL = 10^{-8}$ over time $T = 50$

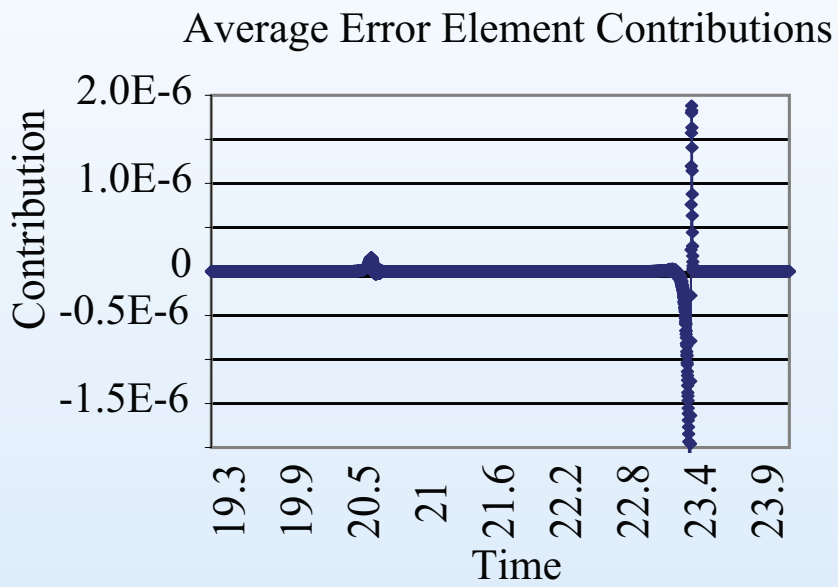
Probabilistic Adaptivity for the Oregonator



A typical Oregonator solution has long periods during which the solution is moderately valued and relatively “quiet” punctuated by rapid transients during which the solution values change enormously. We show a solution during one of the transient periods.

Probabilistic Adaptivity for the Oregonator





Element contributions for two quantities of interest: value at the final time (left) and average error over time (right). The contributions near transient points are much larger.

The classic dual-weighted optimal control approach requires 188,279 time steps

The probability approach requires 20108 time steps

Should We Think of Numerical Error as Certain?

Operator Decomposition

Operator decomposition is widely used for solving multiphysics, multiscale problems

The problem is decomposed into components representing relatively simple physics and/or single scales and a global solution is sought by iterating solutions of the components

Operator decomposition discretizes the instantaneous interaction between the physical components and scales

This creates several new sources of error and instability

I demonstrate with three examples

A Coupled Parabolic Problem

A parabolic interface problem

$$\begin{cases} \frac{\partial T_1}{\partial t} - \nabla \cdot (\chi_1 \nabla T_1) = f_1, & x \in \Omega_1 = (0,1) \times (0,1) \\ \frac{\partial T_2}{\partial t} - \nabla \cdot (\chi_2 \nabla T_2) = f_2, & x \in \Omega_2 = (1,2) \times (0,1) \end{cases}$$

with coupling conditions

$$\begin{cases} T_1 = T_2, \\ \chi_1 \nabla T_1 \cdot \boldsymbol{n} = \chi_2 \nabla T_2 \cdot \boldsymbol{n}, \end{cases} \qquad x \in \Gamma = 1 \times (0,1)$$

We use second order finite elements in space and an implicit integration method in time for each component

A Typical Operator Decomposition Solution

Let T_i^n denote an approximation at time t_n.

A basic block "Gauss-Seidel" algorithm:

- Given T_2^{n-1}, solve for T_1^n with $T_1^n = T_2^{n-1}$ on the interface.
- Given T_1^n, solve for T_2^n with $\chi_2 \nabla T_2^n \cdot \boldsymbol{n} = \chi_1 \nabla T_1^n \cdot \boldsymbol{n}$ on the interface
- Move on to t_{n+1}

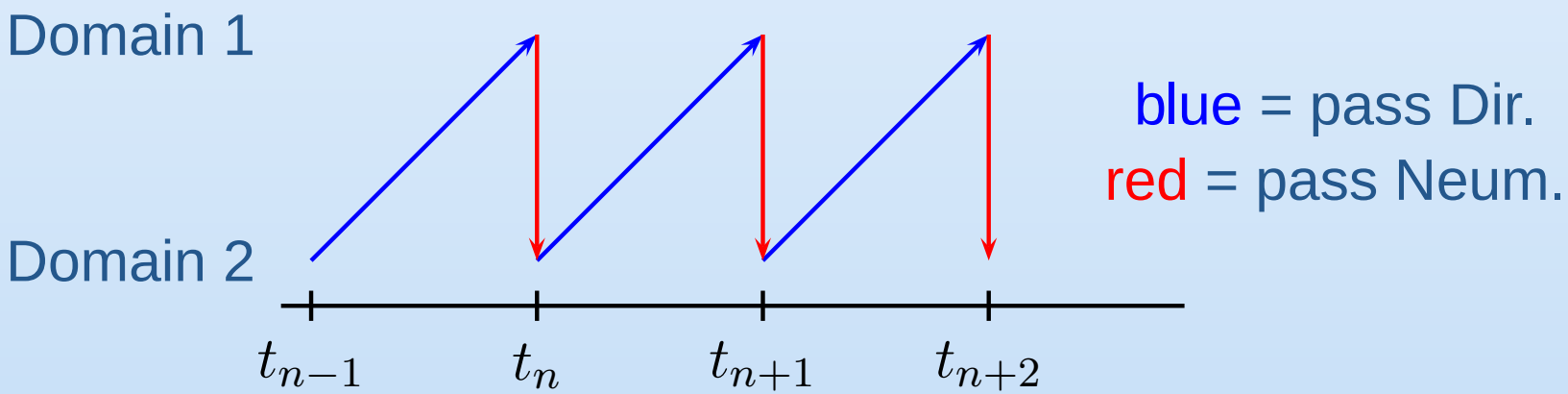
A Typical Operator Decomposition Solution

Let T_i^n denote an approximation at time t_n.

A basic block "Gauss-Seidel" algorithm:

- Given T_2^{n-1}, solve for T_1^n with $T_1^n = T_2^{n-1}$ on the interface.
- Given T_1^n, solve for T_2^n with $\chi_2 \nabla T_2^n \cdot \boldsymbol{n} = \chi_1 \nabla T_1^n \cdot \boldsymbol{n}$ on the interface
- Move on to t_{n+1}

Illustration



A Typical Operator Decomposition Solution

Let T_i^n denote an approximation at time t_n.

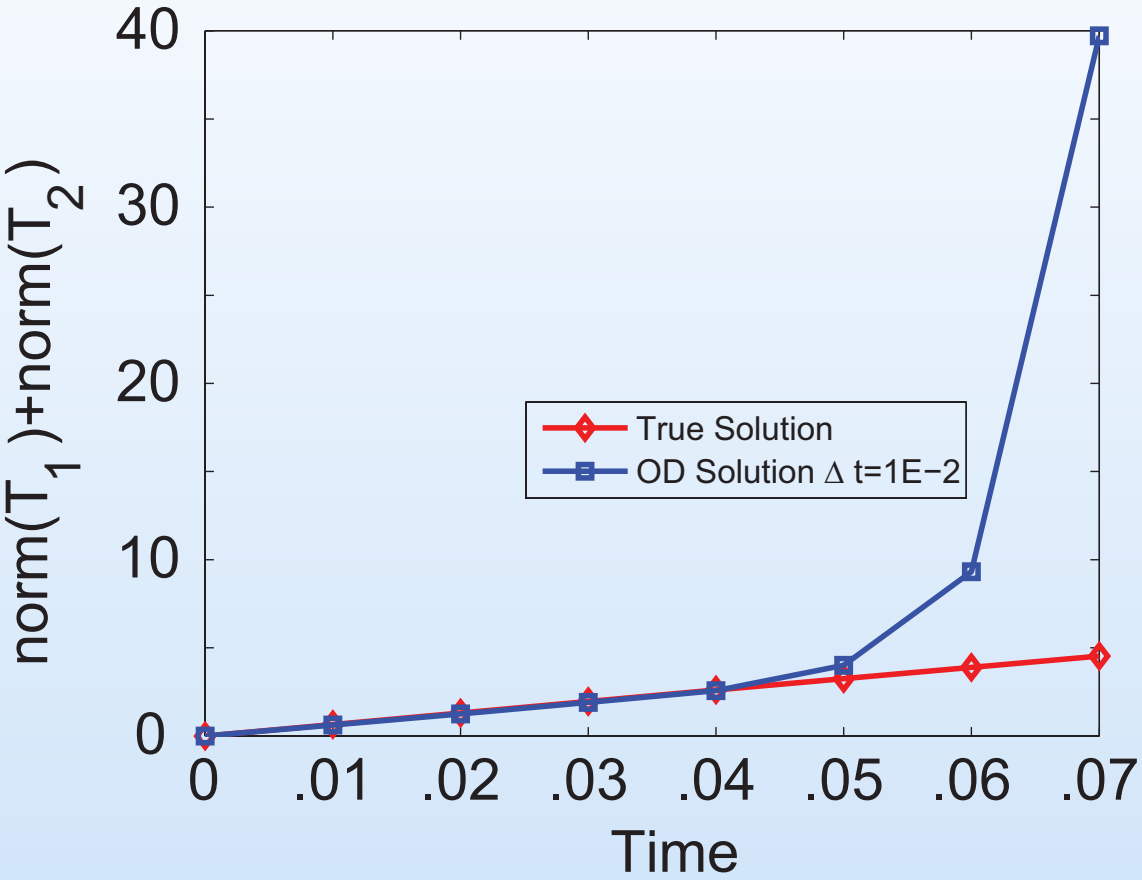
A basic block "Gauss-Seidel" algorithm:

- Given T_2^{n-1}, solve for T_1^n with $T_1^n = T_2^{n-1}$ on the interface.
- Given T_1^n, solve for T_2^n with $\chi_2 \nabla T_2^n \cdot \boldsymbol{n} = \chi_1 \nabla T_1^n \cdot \boldsymbol{n}$ on the interface
- Move on to t_{n+1}

There are many other possibilities of course

Accumulation of the Iteration Error for Gauss-Seidel

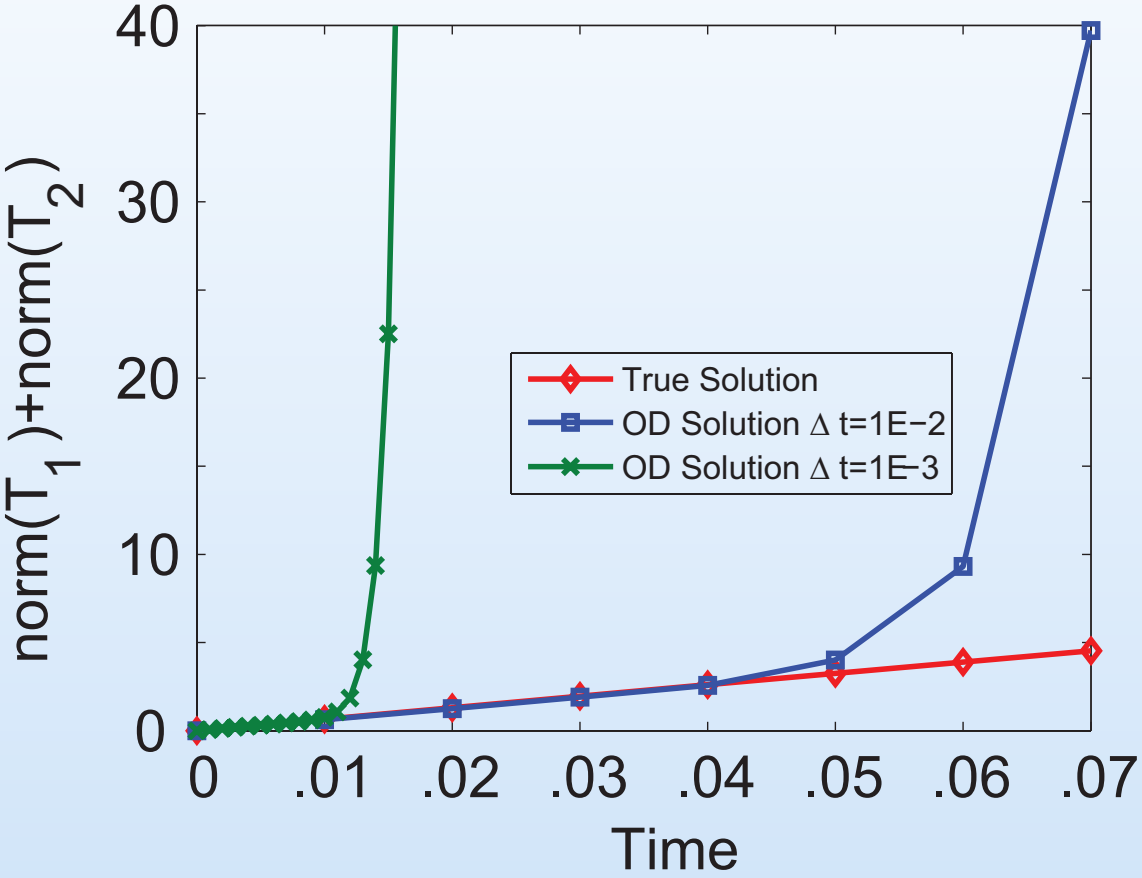
With $\Delta t = 1 \times 10^{-2}$,



The contribution to the error from incomplete iteration dominates the error and it accumulate as time passes

Accumulation of the Iteration Error for Gauss-Seidel

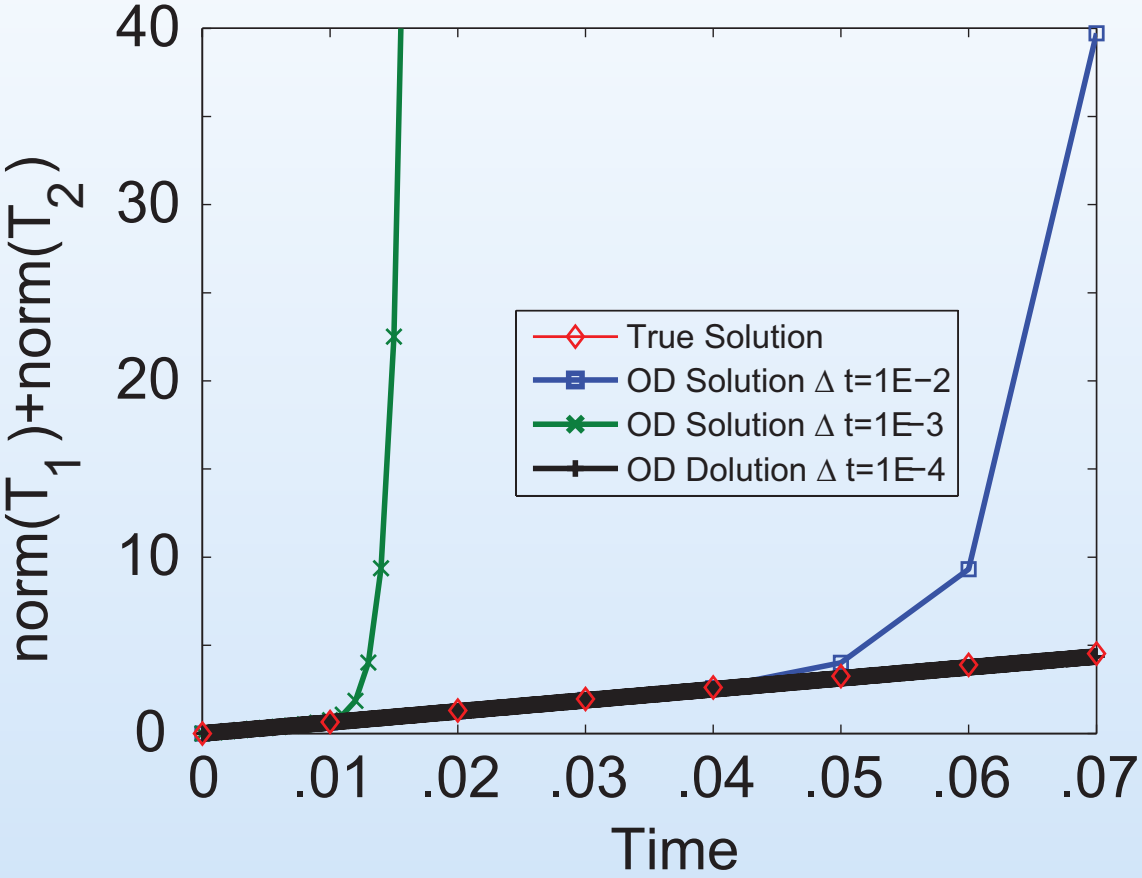
Reducing Δt to 1×10^{-3},



The iteration error accumulates more rapidly when we decrease the time step by a factor of 10

Accumulation of the Iteration Error for Gauss-Seidel

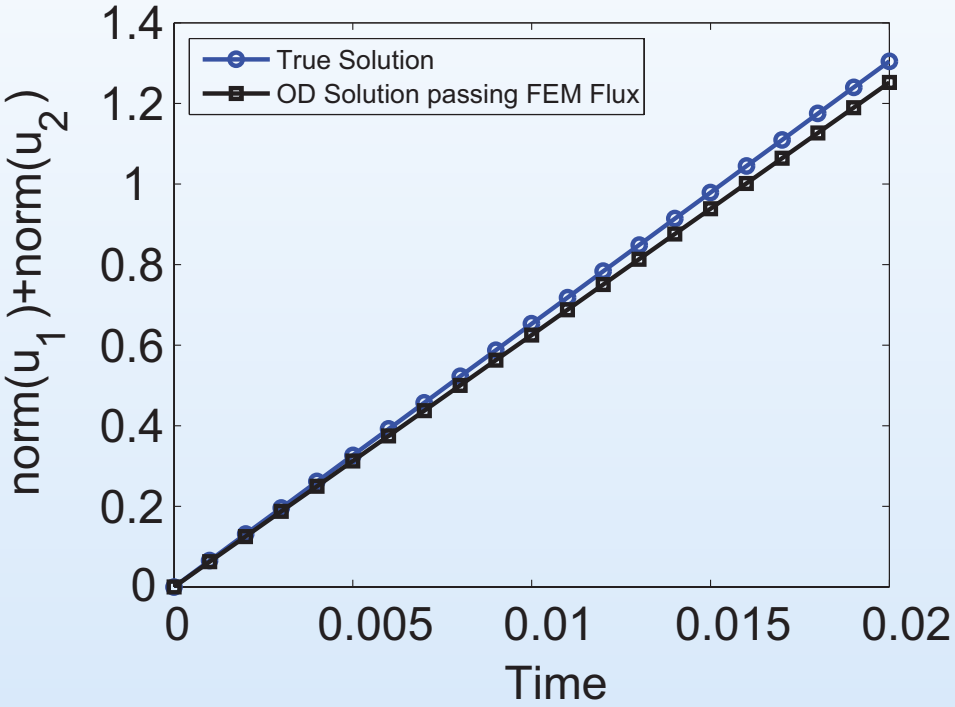
Reducing Δt to 1×10^{-4},



Decreasing the step further cause the iteration error to drop significantly

Transfer Error

For $\Delta t = 10^{-3}$, we iterate at each step until the residual is 10^{-8}



However, there is still a significant source of error remaining for small time steps. The error has reached 5% on a short time interval and worsens considerably as time passes.

The source of error is passing the numerical flux $\chi_1 \nabla T_1^n \cdot \boldsymbol{n}$, which reduces the accuracy from $O(\Delta t + \Delta x^2)$ to $O(\Delta t + \Delta x)$

Operator Decomposition for an Elliptic System

Consider

$$\begin{cases} -\Delta u_1 = \sin(4\pi x)\sin(\pi y), & x \in \Omega \\ -\Delta u_2 = b \cdot \nabla u_1 = 0, & x \in \Omega, \\ u_1 = u_2 = 0, & x \in \partial\Omega, \end{cases} \qquad b = \frac{2}{\pi}\begin{pmatrix} 25\sin(4\pi x) \\ \sin(\pi x) \end{pmatrix}$$

where information from u_1 is passed to the equation for u_2

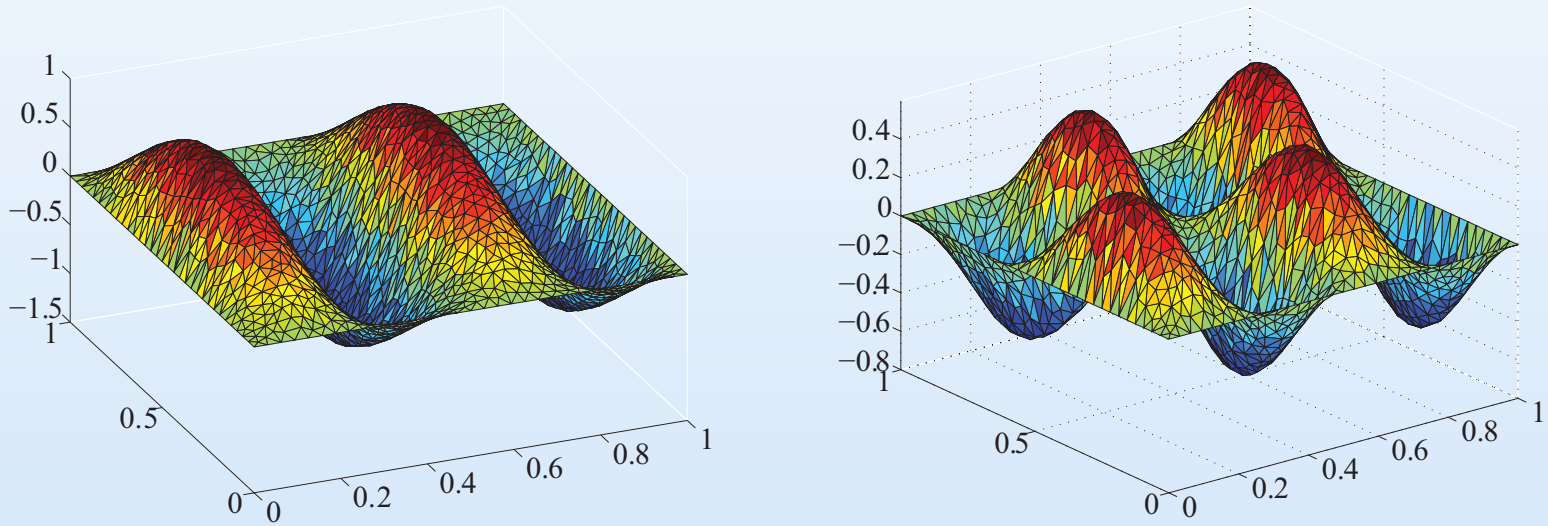
We consider the quantity of interest

$$u_2(.25, .25) \approx \langle \delta_{\text{reg}}(.25, .25), u_2 \rangle$$

We solve for u_1 first and then solve for u_2

Operator Decomposition for an Elliptic System

For the error in $u(.25, .25)$ computed on uniformly fine meshes
discretization contribution $\approx .0042$
decomposition contribution $\approx .0006$



Solutions of components 1 and 2

The decomposition contribution to the error is small but significant

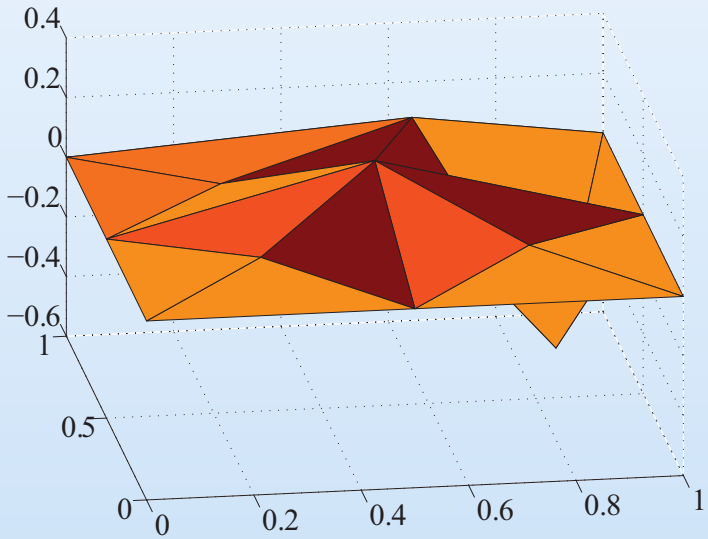
Operator Decomposition for an Elliptic System

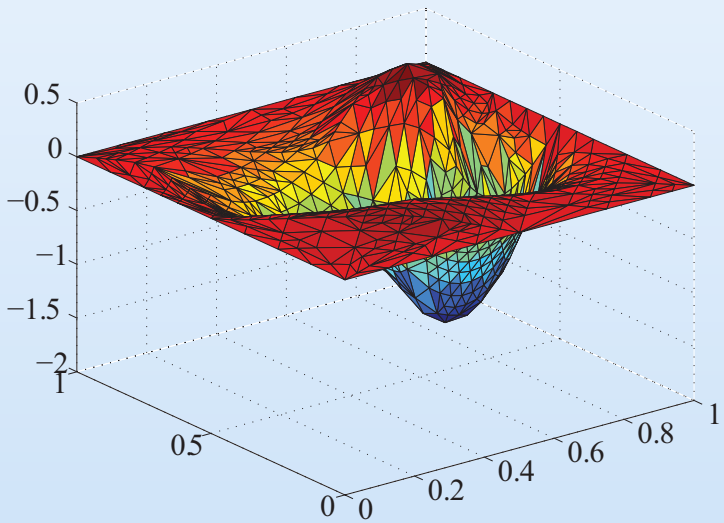
We adapt the mesh while ignoring the contributions to the error from operator decomposition

The error from decomposition becomes worse

discretization error $\approx .0001$

decomposition contribution $\approx .2244$





U_1 on a coarse mesh and the total error of U_2

Decomposition for a Reaction-Diffusion Problem

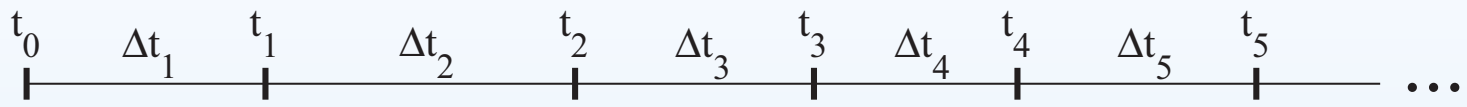
We consider the reaction-diffusion problem

$$\begin{cases} \frac{du}{dt} = \Delta u + F(u), \quad 0 < t, \\ u(0) = u_0 \end{cases}$$

The diffusion component Δu induces stability and change over long time scales

The reaction component F induces instability and change over short time scales

Decomposition for a Reaction-Diffusion Problem



On $(t_{n-1}, t_n]$, we numerically solve

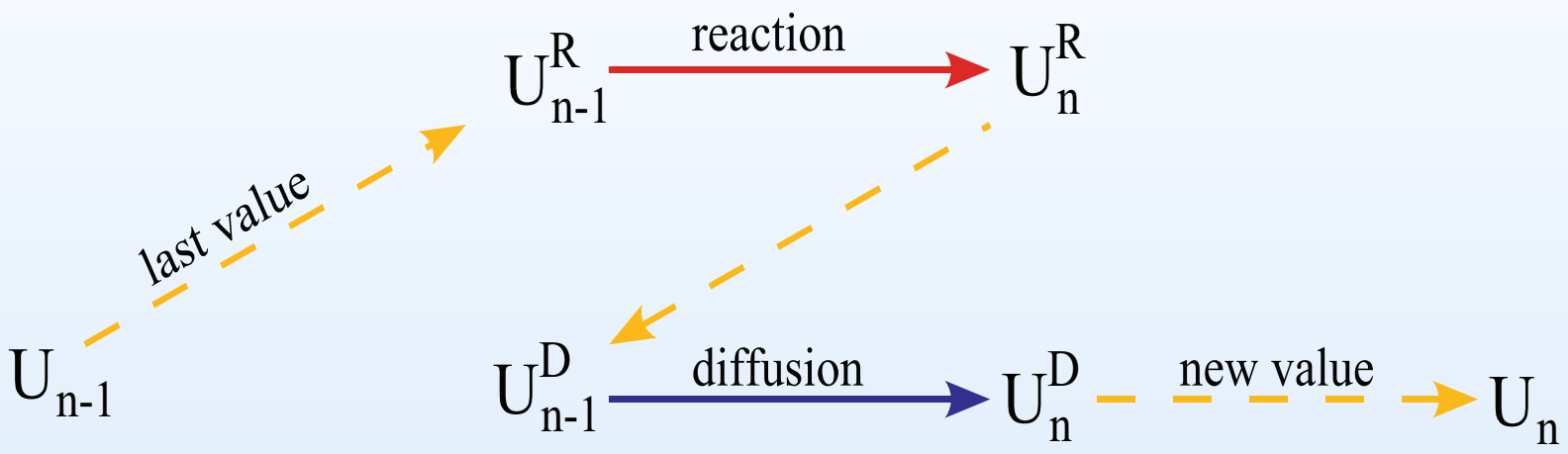
$$\begin{cases} \frac{du^R}{dt} = F(u^R), & t_{n-1} < t \leq t_n, \\ u^R(t_{n-1}) = u^D(t_{n-1}) \end{cases}$$

Then on $(t_{n-1}, t_n]$, we numerically solve

$$\begin{cases} \frac{du^D}{dt} - \Delta(u^D), & t_{n-1} < t \leq t_n, \\ u^D(t_{n-1}) = u^R(t_n) \end{cases}$$

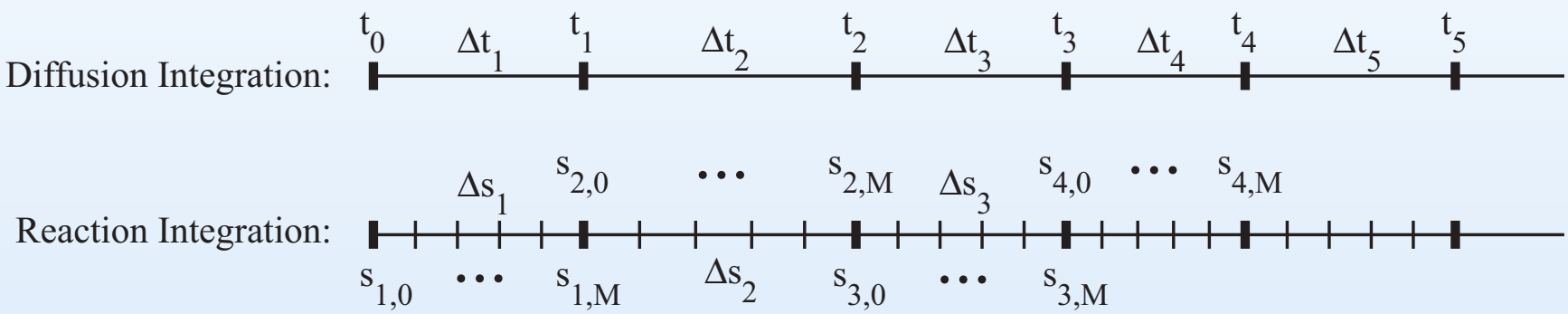
The operator decomposition approximation is $u(t_n) \approx u^D(t_n)$

Decomposition for a Reaction-Diffusion Problem



Decomposition for a Reaction-Diffusion Problem

To account for the fast reaction, we approximate u^r using many time steps inside each diffusion step



This is a multiscale time integration scheme

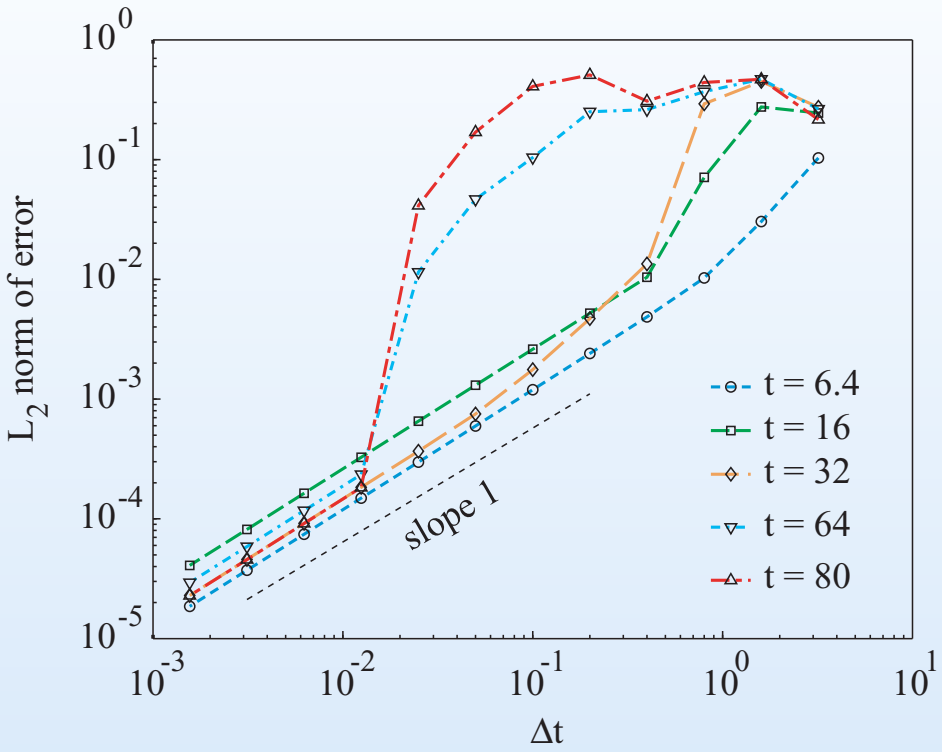
Decomposition for a Reaction-Diffusion Problem

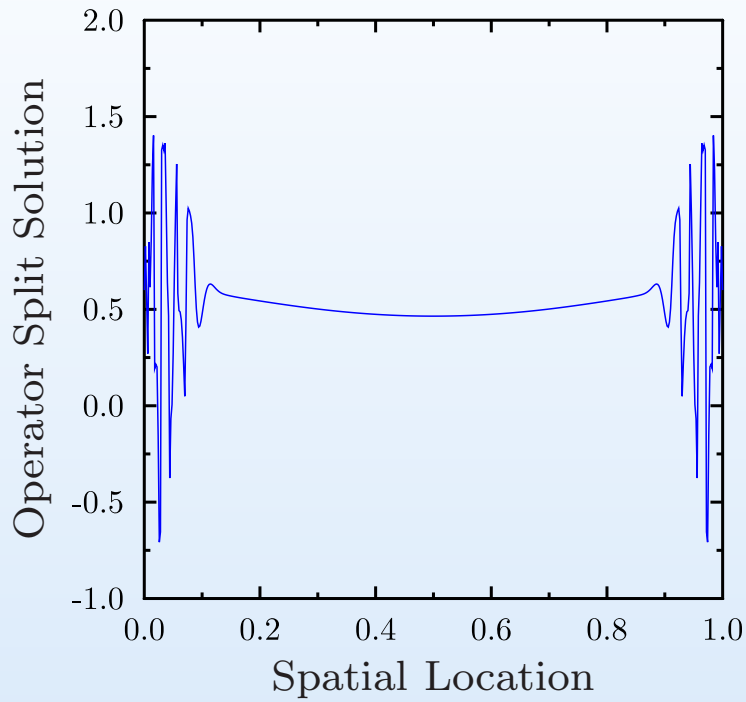
The Brusselator problem

$$\begin{cases} \frac{\partial u_i}{\partial t} - 0.025\,\frac{\partial^2 u_i}{\partial x^2} = f_i(u_1, u_2) \;\; i = 1, 2 \\ f_1(u_1, u_2) = 0.6 - 2u_1 + u_1^2 u_2 \\ f_2(u_1, u_2) = 2u_1 - u_1^2 u_2 \end{cases}$$

- Use a linear finite element method in space with 500 elements
- Use a standard first order splitting scheme
- Use Trapezoidal Rule with time step of $.2$ for the diffusion and Backward Euler with time step of $.004$ for the reaction

Decomposition for a Reaction-Diffusion Problem





On the left, we plot the accuracy versus the time step at a collection of times. For long times, there is a critical time step above which there is no convergence. This is due to instability arising from operator decomposition. On the right, we plot a typical unstable solution. The oscillations are **not** on the scale of the space mesh.

A Posteriori Analysis of Operator Decomposition

The conclusion is that numerical solution of multiphysics, multiscale problems can lead to unpredictable numerical errors

Note: We have adapted a posteriori analysis to operator decomposition and we can estimate the error in all three cases

The ingredients are

- Estimating the error arising from incomplete iteration
- Estimating the error in information passed between components
- Estimating the changes to the adjoint arising from operator decomposition

Mitigating the effects of operator decomposition is usually not a question of straightforward mesh refinement

Data Assimilation

Data Assimilation

Basic problem:

Given the current and past state of a physical system, what is the best prediction for future behavior?

Data assimilation is a method for fusing data collected from observation of a system to mathematical models of the dynamics of the system

Data assimilation is becoming increasingly important in a wide range of applications

There are enormous financial consequences riding on the reliability of data assimilation predictions, e.g. climate modeling

Data Assimilation

A model describing the processes in the system is defined in terms of parameters and data that characterize a particular state of the system

The parameters and data in a model are determined by matching quantities of interest computed from model output to observations of the state of the modeled system

The model is then integrated forward in time to predict the future behavior using the “best fit” parameter values

There are many approaches to assimilation

- Estimation theory, e.g. Kalman filtering
- Control theory approaches
- Direct minimization, e.g. simulated annealing, gradient search

Data Assimilation

There are many interesting theoretical and computational problems underlying data assimilation that are waiting to be tackled

It is probably fair to say that data assimilation has received the most attention from the statistics-oriented community

One consequence is that current data assimilation practice focuses on assimilating against state values

This discards information about the sensitivity and stability of model output determined by the characteristics of differential equations

Assimilating for the Lorenz Model

The Lorenz equations are

$$\begin{cases} \dot{x}_1 = \sigma(x_2 - x_1), \\ \dot{x}_2 = rx_1 - x_2 - x_1x_3, \\ \dot{x}_3 = x_1x_2 - bx_3, \\ x_1(0) = a_1,\ x_2(0) = a_2,\ x_3(0) = a_3, \end{cases} \qquad t > 0,$$

We compute a true solution for $r = 24.2$, $\sigma = 10$, $b = 8/3$ accurately then introduce small independent random errors to each solution value to model “measurement” error

We consider parameters drawn from a region $r \sim U(22.2, 26.2)$, $\sigma \sim U(9, 11)$, $b \sim (5/3, 11/3)$

Assimilating for the Lorenz Model

We assimilate against the norm

$$\left(\sum_{i=1}^{N} \left(U_i^{\text{true}} - U_i^{\text{model}} \right)^2 \right)^{1/2}$$

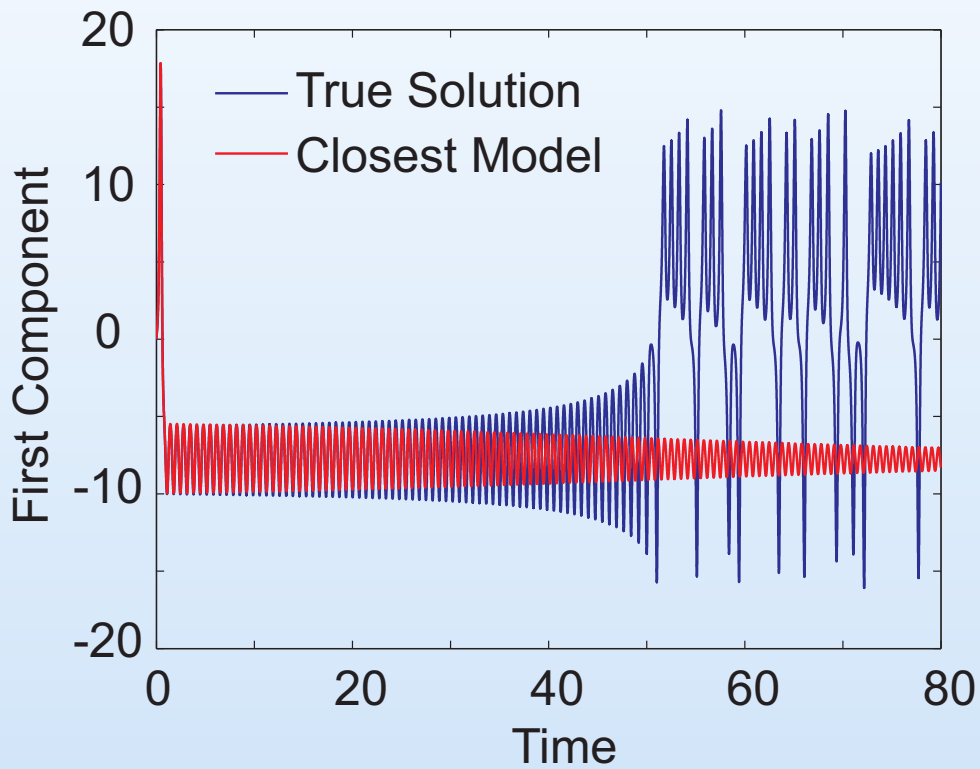
where $\{U_i^{\text{true}}\}$ and $\{U_i^{\text{model}}\}$ are the true and model observations

We assimilate over $[0, 5]$ using 5000 points and over $[0, 20]$ using 20000 points

As a crude solution to the assimilation problem, we compute a random selection (1000) of model solutions and choose the closest

Assimilation Without Error Control

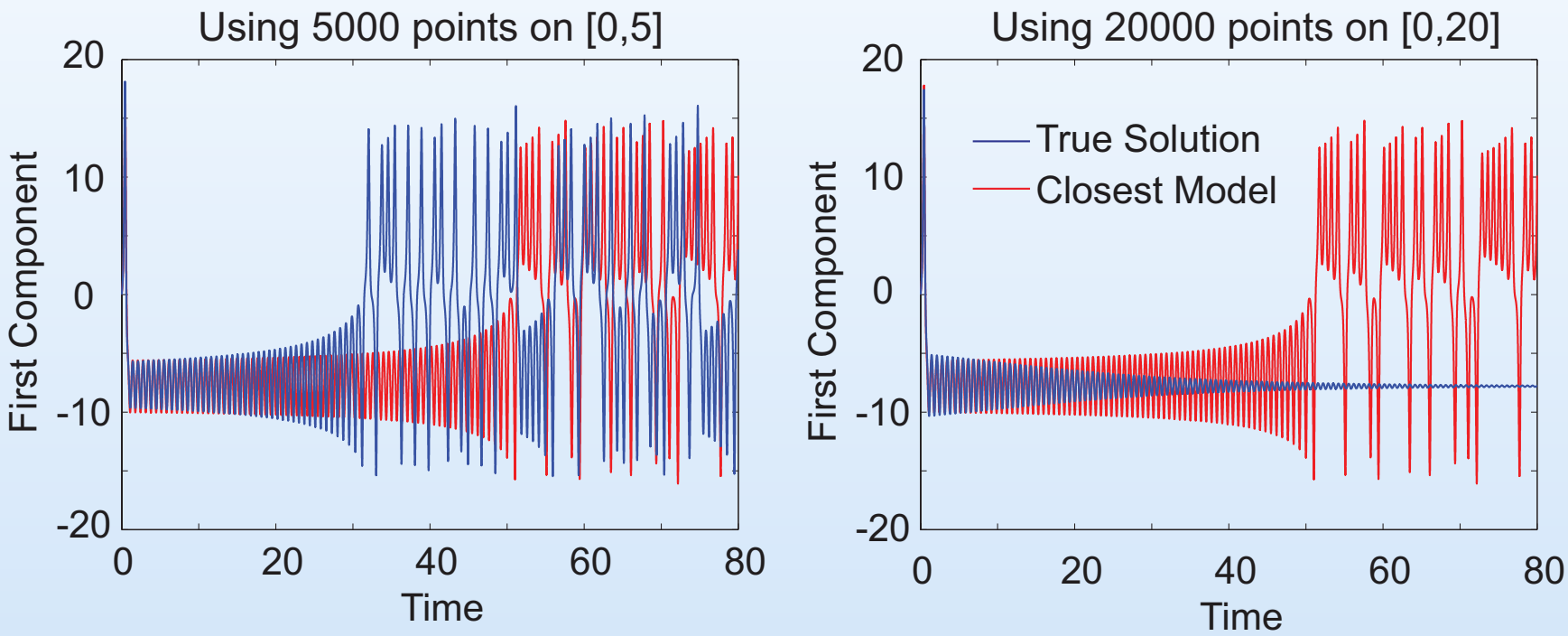
We show results for a numerical solution computed using fixed time steps over $[0, 20]$



The closest parameter fit leads to a solution that converges to a steady-state, i.e. that has the incorrect dynamical behavior. Any prediction of behavior of the solution from time 20 forward will be wrong.

Assimilation With Error Control

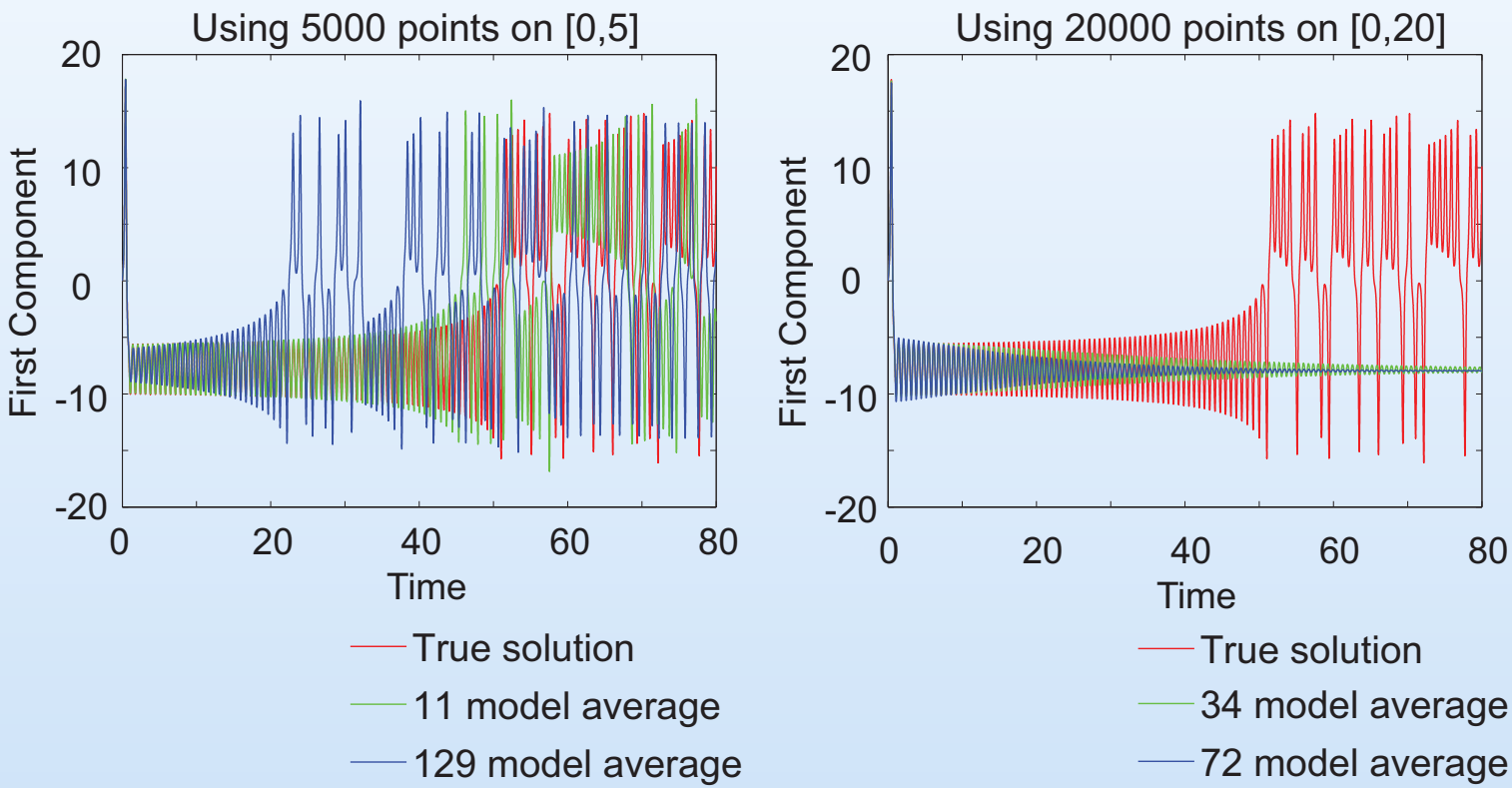
We show results for numerical solutions computed using adaptive error control



Assimilating over $[0, 5]$ leads to a chaotic solution that begins to diverge immediately after time 5. Assimilating over $[0, 20]$ leads to another non-chaotic solution.

Assimilation Using Model Averages

We assimilate against a number of model solutions computed with adaptive error control and then average the parameter values



Averaging parameters from a set of closest solutions leads to the same results

Data Assimilation

Of course, this is a simplistic experiment

It does point to the need to go past the simplest form of data assimilation

A differential equation determines the sensitivities of quantities of interest computed from solutions, i.e. derivatives with respect to data and parameters,

It is interesting to consider assimilating against sensitivity information as well as state

Conclusion

Statistical, Probabilistic, and Deterministic Analysis

Using my limited experience in some fundamental problems in computational modeling, I have tried to support the claim

The problem of quantifying all sources of uncertainty in computational modeling demands a fusion of deterministic, probabilistic, and statistical approaches