SAND2012-2796P

Uncertainty Quantification: UQTk example problems

Bert Debusschere[1] Roger Ghanem[2]

[1]Sandia National Laboratories
Livermore, CA, USA
bjdebus@sandia.gov

[2]University of Southern California
Los Angeles, CA, USA

SIAM 2012 Conference on Uncertainty Quantification

Acknowledgement

K. Sargsyan, C. Safta, H.N. Najm
— Sandia National Laboratories, CA
O.M. Knio — Duke Univ., Durham, NC
A. Alexanderian — Johns Hopkins Univ., Baltimore, MD

This work was supported by:

- US Department of Energy (DOE), Office of Advanced Scientific Computing Research (ASCR), Scientific Discovery through Advanced Computing (SciDAC)
- DOE ASCR Applied Mathematics program.

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 Energys National Nuclear Security Administration under contract DE-AC04-94AL85000.

Outline

Outline

Uncertainty Quantification Toolkit (UQTk)

- A library of C++ and Matlab functions for propagation of uncertainty through computational models
- Mainly relies on spectral Polynomial Chaos Expansions (PCEs) for representing random variables and stochastic processes
- Target usage:
  - Rapid prototyping
  - Algorithmic research
  - Tutorials / Educational
- Version 1.0 released under the GNU Lesser General Public License
- Downloadable from http://www.sandia.gov/UQToolkit/

UQTk contents and development/release plans

- Currently released (http://www.sandia.gov/UQToolkit/)
  - C++ Tools for intrusive UQ with PCEs
- Under production, planned release Fall 2012
  - C++ Tools for non-intrusive UQ
  - Matlab tools for intrusive and non-intrusive UQ
  - Karhunen-Loève decomposition
  - Bayesian inference tools
  - Many more examples and documentation
- Under development
  - Adding support for multiwavelet based stochastic domain decompositions
  - Support for arbitrary basis types

Outline

Polynomial Chaos Expansions (PCEs)

- Represent random variables with finite variance as polynomials of standard random variables

$$U(\boldsymbol{\theta}) \simeq \sum_{k=0}^{P} u_k \Psi_k(\boldsymbol{\xi})$$

  - Truncated to finite dimension n and order p
  - Numer of terms $P+1 = \frac{(n+p)!}{n!p!}$
  - $\boldsymbol{\xi} = (\xi_1, \xi_2, \cdots, \xi_n)$ standard i.i.d. random variables
  - Ψ_k standard orthogonal polynomials
  - u_k spectral modes or PC coefficients
- Most common standard Polynomial-Variable pairs:
  - (continuous) Gauss-Hermite, Legendre-Uniform
  - (discrete) Poisson-Charlier

[Wiener, 1938; Ghanem & Spanos, 1991; Xiu & Karniadakis, 2002]

PCEs in the UQToolkit

```
// Initialize PC class
int ord = 5;   // Order of PCE
int dim = 1;   // Number of uncertain parameters
PCSet myPCSet("ISP",ord,dim,"LU"); // Legendre-Uniform PCEs

// Initialize PC class
int ord = 5;   // Order of PCE
int dim = 1;   // Number of uncertain parameters
PCSet myPCSet("NISP",ord,dim,"LU"); // Legendre-Uniform PCEs
```

- Currently support Wiener-Hermite, Legendre-Uniform, and Gamma-Laguerre (limited), Jacobi-Beta (development version)
- PCSet class initializes PC basis type and pre-computes information needed for working with PC expansions

Operations on PCEs in the UQToolkit

```
// PC coefficients in double*
double* a = new double[npc];
double* b = new double[npc];
double* c = new double[npc];

// Initialization
a[0] = 2.0;
a[1] = 0.1;
...
// Perform some arithmetic
myPCSet.Subtract(a,b,c);
myPCSet.Prod(a,b,c);
myPCSet.Exp(a,c);
myPCSet.Log(a,c);
```

```
// PC coefficients in Arrays
Array1D<double> aa(npc,0.e0);
Array1D<double> ab(npc,0.e0);
Array1D<double> ac(npc,0.e0);

// Initialization
aa(0) = 2.0;
aa(1) = 0.1;
...
// Perform arithmetic
myPCSet.Subtract(aa,ab,ac);
myPCSet.Prod(aa,ab,ac);
myPCSet.Exp(aa,ac);
myPCSet.Log(aa,ac);
```

- PC coefficients are either stored in `double*` vectors or in more advanced custom `Array1D<double>` classes
- Functions can take either data type as argument

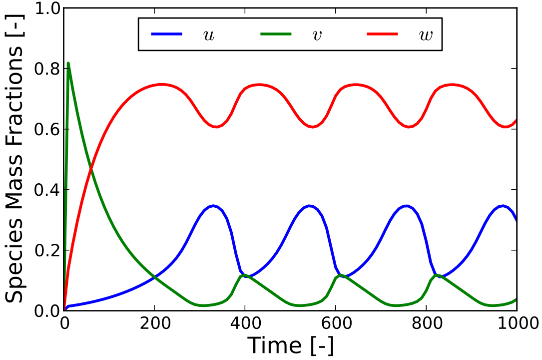
Outline

Surface Reaction Model

3 ODEs for a monomer (u), dimer (v), and inert species (w) adsorbing onto a surface out of gas phase.

$$\frac{\mathrm{d}u}{\mathrm{d}t} = az - cu - 4duv$$
$$\frac{\mathrm{d}v}{\mathrm{d}t} = 2bz^2 - 4duv$$
$$\frac{\mathrm{d}w}{\mathrm{d}t} = ez - fw$$
$$z = 1 - u - v - w$$

$$u(0) = v(0) = w(0) = 0.0$$



Oscillatory behavior for $b \in [20.2, 21.2]$

$a = 1.6$ $b = 20.75$ $c = 0.04$ $d = 1.0$ $e = 0.36$ $f = 0.016$

(Vigil *et al.*, Phys. Rev. E., 1996; Makeev *et al.*, J. Chem. Phys., 2002)

Surface Reaction Model: Intrusive Spectral Propagation (ISP) of Uncertainty

- Assume PCE for uncertain parameter b and for the output variables, u, v, w
- Substitute PCEs into the governing equations
- Project the governing equations onto the PC basis functions
  - Multiply with Ψ_k and take the expectation
- Apply pseudo-spectral approximations where necessary
- UQTk elementary operations

Surface Reaction Model: Specify PCEs for inputs and outputs

Represent uncertain inputs with PCEs with known coefficients:

$$b = \sum_{i=0}^{P} b_i \Psi_i(\xi)$$

Represent all uncertain variables with PCEs with unknown coefficients:

$$u = \sum_{i=0}^{P} u_i \Psi_i(\xi) \quad v = \sum_{i=0}^{P} v_i \Psi_i(\xi) \quad w = \sum_{i=0}^{P} w_i \Psi_i(\xi) \quad z = \sum_{i=0}^{P} z_i \Psi_i(\xi)$$

Surface Reaction Model: Substitute PCEs into governing equations and project onto basis functions

$$\begin{aligned}
\frac{\mathrm{d}u}{\mathrm{d}t} &= az - cu - 4duv \\
\frac{\mathrm{d}}{\mathrm{d}t}\sum_{i=0}^{P} u_i\Psi_i &= a\sum_{i=0}^{P} z_i\Psi_i - c\sum_{i=0}^{P} u_i\Psi_i - 4d\sum_{i=0}^{P} u_i\Psi_i\sum_{j=0}^{P} v_j\Psi_j \\
\left\langle \Psi_k\frac{\mathrm{d}}{\mathrm{d}t}\sum_{i=0}^{P} u_i\Psi_i \right\rangle &= \left\langle a\Psi_k\sum_{i=0}^{P} z_i\Psi_i \right\rangle - \left\langle c\Psi_k\sum_{i=0}^{P} u_i\Psi_i \right\rangle \\
&- \left\langle 4d\Psi_k\sum_{i=0}^{P} u_i\Psi_i\sum_{j=0}^{P} v_j\Psi_j \right\rangle
\end{aligned}$$

Surface Reaction Model: Reorganize terms

$$\frac{\mathrm{d}}{\mathrm{d}t} u_k \left\langle \Psi_k^2 \right\rangle = a z_k \left\langle \Psi_k^2 \right\rangle - c u_k \left\langle \Psi_k^2 \right\rangle - 4d \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j \left\langle \Psi_i \Psi_j \Psi_k \right\rangle$$

$$\frac{\mathrm{d}}{\mathrm{d}t} u_k = a z_k - c u_k - 4d \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j \frac{\left\langle \Psi_i \Psi_j \Psi_k \right\rangle}{\left\langle \Psi_k^2 \right\rangle}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} u_k = a z_k - c u_k - 4d \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j C_{ijk}$$

- Triple products $C_{ijk} = \frac{\left\langle \Psi_i \Psi_j \Psi_k \right\rangle}{\left\langle \Psi_k^2 \right\rangle}$ can be pre-computed and stored for repeated use

Surface Reaction Model: Substitute PCEs into governing equations and project onto basis functions

$$
\begin{aligned}
\frac{\mathrm{d}v}{\mathrm{d}t} &= 2bz^2 - 4duv \\
\frac{\mathrm{d}}{\mathrm{d}t}\sum_{i=0}^{P} v_i\Psi_i &= 2\sum_{h=0}^{P} b_h\Psi_h \sum_{i=0}^{P} z_i\Psi_i \sum_{j=0}^{P} z_j\Psi_j - 4d\sum_{i=0}^{P} u_i\Psi_i \sum_{j=0}^{P} v_j\Psi_j \\
\left\langle \Psi_k \frac{\mathrm{d}}{\mathrm{d}t}\sum_{i=0}^{P} v_i\Psi_i \right\rangle &= \left\langle 2\Psi_k \sum_{h=0}^{P} b_h\Psi_h \sum_{i=0}^{P} z_i\Psi_i \sum_{j=0}^{P} z_j\Psi_j \right\rangle \\
&- \left\langle 4d\Psi_k \sum_{i=0}^{P} u_i\Psi_i \sum_{j=0}^{P} v_j\Psi_j \right\rangle
\end{aligned}
$$

Surface Reaction Model: Reorganize terms

$$\frac{\mathrm{d}}{\mathrm{d}t} v_k \left\langle \Psi_k^2 \right\rangle = 2 \sum_{h=0}^{P} \sum_{i=0}^{P} \sum_{j=0}^{P} b_h z_i z_j \left\langle \Psi_h \Psi_i \Psi_j \Psi_k \right\rangle - 4d \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j \left\langle \Psi_i \Psi_j \Psi_k \right\rangle$$

$$\frac{\mathrm{d}}{\mathrm{d}t} v_k = 2 \sum_{h=0}^{P} \sum_{i=0}^{P} \sum_{j=0}^{P} b_h z_i z_j \frac{\left\langle \Psi_h \Psi_i \Psi_j \Psi_k \right\rangle}{\left\langle \Psi_k^2 \right\rangle} - 4d \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j \frac{\left\langle \Psi_i \Psi_j \Psi_k \right\rangle}{\left\langle \Psi_k^2 \right\rangle}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} v_k = 2 \sum_{h=0}^{P} \sum_{i=0}^{P} \sum_{j=0}^{P} b_h z_i z_j D_{hijk} - 4d \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j C_{ijk}$$

- Pre-computing and storing the quad product D_{hijk} becomes cumbersome
- Use pseudo-spectral approach instead

Surface Reaction Model: Pseudo-Spectral approach for products

- Introduce auxiliary variable $g = z^2$

$$\begin{aligned} g &= z^2 \\ f = 2bz^2 &= 2bg \end{aligned}$$

$$\begin{aligned} g_k &= \sum_{i=0}^{P}\sum_{j=0}^{P} z_i z_j C_{ijk} \\ f_k &= 2\sum_{i=0}^{P}\sum_{j=0}^{P} b_i g_j C_{ijk} \end{aligned}$$

- Limits the complexity of computing product terms
  - Higher products can be computed by repeated use of the same binary product rule
- Does introduce errors if order of PCE is not large enough

Surface Reaction Model: UQTk implementation

```
// Build du/dt = a*z - c*u - 4.0*d*u*v
aPCSet.Multiply(z,a,dummy1);            // dummy1 = a*z
aPCSet.Multiply(u,c,dummy2);            // dummy2 = c*u
aPCSet.SubtractInPlace(dummy1,dummy2); // dummy1 = a*z - c*u
aPCSet.Prod(u,v,dummy2);               // dummy2 = u*v
aPCSet.MultiplyInPlace(dummy2,4.e0*d); // dummy2 = 4.0*d*u*v
aPCSet.Subtract(dummy1,dummy2,dudt);   // dudt = a*z - c*u - 4.0*d*u*v
```

- All operations are replaced with their equivalent intrusive UQ counterparts
- Results in a set of coupled ODEs for the PC coefficients
  - u, v, w, z represent vector of PC coefficients
- This set of equations is integrated to get the evolution of the PC coefficients in time

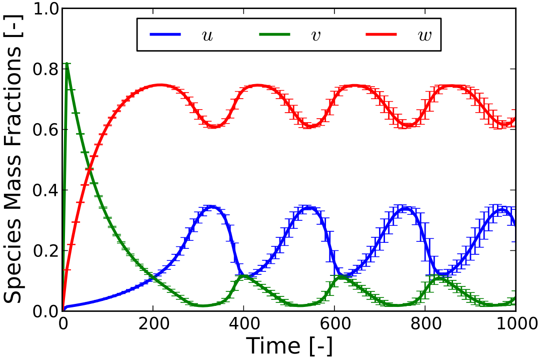
Surface Reaction Model: Second equation implementation

```
// Build dv/dt = 2.0*b*z*z - 4.0*d*u*v
aPCSet.Prod(z,z,dummy1);                  // dummy1 = z*z
aPCSet.Prod(dummy1,b,dummy2);             // dummy2 = b*z*z
aPCSet.Multiply(dummy2,2.e0,dummy1);      // dummy1 = 2.0*b*z*z
aPCSet.Prod(u,v,dummy2);                  // dummy2 = u*v
aPCSet.MultiplyInPlace(dummy2,4.e0*d);    // dummy2 = 4.0*d*u*v
aPCSet.Subtract(dummy1,dummy2,dvdt);      // dvdt = 2.0*b*z*z - 4.0*d*u*v

// Build dw/dt = e*z - f*w
aPCSet.Multiply(z,e,dummy1);              // dummy1 = e*z
aPCSet.Multiply(w,f,dummy2);              // dummy2 = f*w
aPCSet.Subtract(dummy1,dummy2,dwdt);      // dwdt = e*z - f*w
```

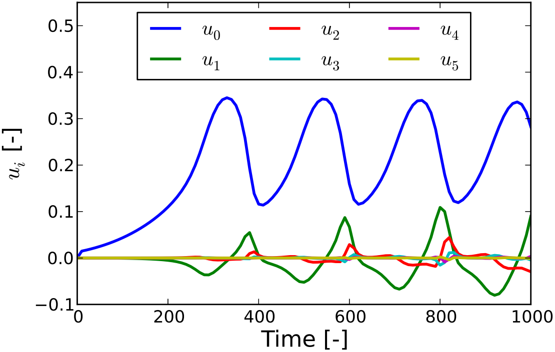
- Dummy variables used where needed to build the terms in the equations
- Data structure is currently being enhanced to provide the operation result as the function return value
  - Will allow more elegant inline replacement of operators with their stochastic counterparts

Surface Reaction Model: ISP results



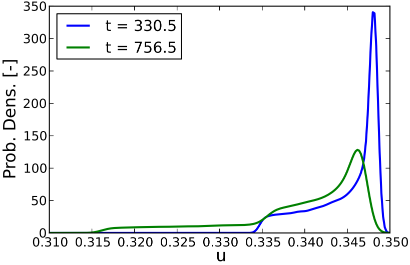
- Assume 0.5% uncertainty in b around nominal value
- Legendre-Uniform intrusive PC
- Mean and standard deviation for u, v, and w
- Uncertainty grows in time

Surface Reaction Model: ISP results



- Modes of u
- Modes decay with higher order
- Amplitudes of oscillations of higher order modes grow in time

Surface Reaction Model: ISP results: PDFs



- Pdfs of u at maximum mean (left) and maximum standard deviation (right)
- Distributions get broader and multimodal as time increases
  - Effect of accumulating phase errors

Surface Reaction Model: Non-Intrusive Propagation of Uncertainty

- Consider two approaches for Non-Intrusive Spectral Projection (NISP):
  - Quadrature
  - Monte Carlo or random-sampling
- Assume PCE for uncertain parameter b and the output variables
- Sample input parameters at quadratrure/random sample points
- Run deterministic forward model for each of the sampled input parameters
- Perform Galerkin projection $u_k = \frac{\langle u\Psi_k\rangle}{\langle \Psi_k^2\rangle}$
  - Quadrature: $\langle u\Psi_k\rangle = \sum_{i=1}^{N_q} w_i u(b_i)$
  - Monte Carlo: $\langle u\Psi_k\rangle = \frac{1}{N_s}\sum_{i=1}^{N_s} u(b_i)$

Surface Reaction Model: NISP implementation in UQTk

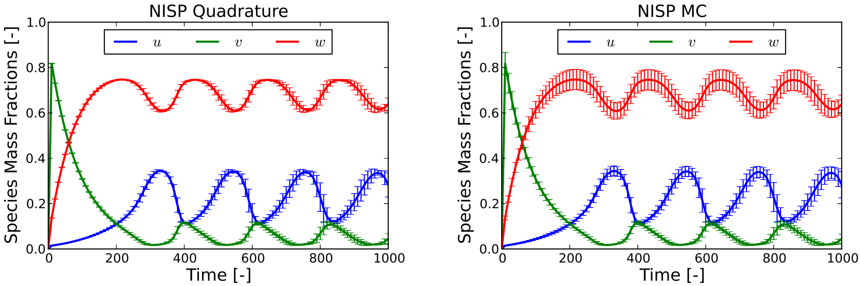
Quadrature:

```
// Get the quadrature points
int nQdpts=myPCSet.GetNQuadPoints();
double* qdpts=new double[nQdpts];
myPCSet.GetQuadPoints(qdpts);
...
// Evaluate parameter at quad pts
for(int i=0;i<nQdpts;i++){
  bval[i]=myPCSet.EvalPC(b,&qdpts[i]);
}
...
// Run model for all samples
for(int i=0;i<nQdpts;i++){
  u_val[i] = ...
}
// Spectral projection
myPCSet.GalerkProjection(u_val,u);
myPCSet.GalerkProjection(v_val,v);
myPCSet.GalerkProjection(w_val,w);
```

Monte-Carlo Sampling:

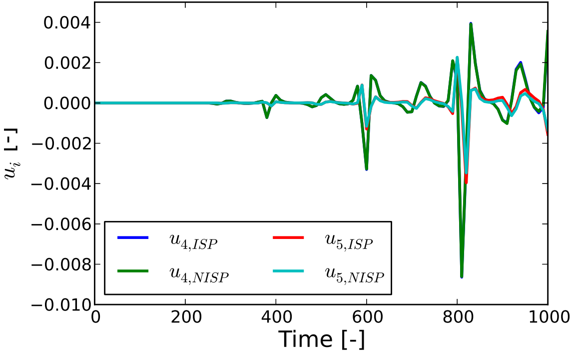
```
// Get the sample points
int nSamples=1000;
Array2D<double> samPts(nSamples,dim);
myPCSet.DrawSampleVar(samPts);
...
// Evaluate parameter at sample pts
for(int i=0;i<nSamples;i++){
  ... // select samPt from samPts
  bval[i]=myPCSet.EvalPC(b,&samPt)
}
...
// Run model for all samples
for(int i=0;i<nSamples;i++){
  u_val[i] = ...
}
// Spectral projection
myPCSet.GalerkProjectionMC(samPts,u_val,u);
myPCSet.GalerkProjectionMC(samPts,v_val,v);
myPCSet.GalerkProjectionMC(samPts,w_val,w);
```

Surface Reaction Model: NISP results



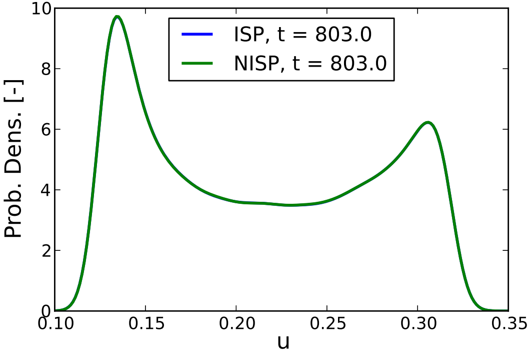
- Mean and standard deviation for u, v, and w
- Quadrature approach agrees well with ISP approach using 6 quadrature points
- Monte Carlo sampling approach converges slowly
  - With a 1000 samples, results are quite different from ISP and NISP

Surface Reaction Model: Comparison ISP and NISP



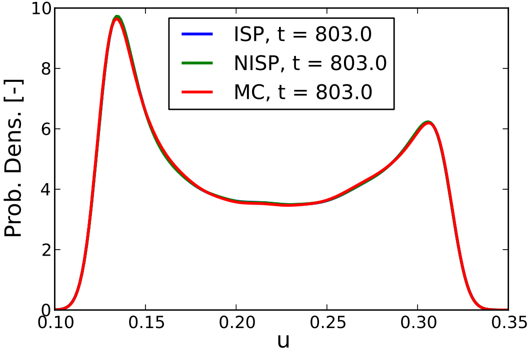
- Lower order modes agree perfectly
- Very small differences in higher order modes
  - Difference increases with time

Surface Reaction Model: Comparison ISP and NISP



- All pdf's based on 50K samples each and evaluated with Kernel Density Estimation (KDE)
- No difference in PDFs of sampled PCEs between NISP and ISP

Surface Reaction Model: Comparison ISP, NISP, and MC



- All pdf's based on 50K samples each and evaluated with Kernel Density Estimation (KDE)
- Good agreement between intrusive, non-intrusive projection, and Monte Carlo sampling

ISP pros and cons

- Pros:
  - Elegant
  - One time solution of system of equations for the PC coefficients fully characterizes uncertainty in all variables at all times
  - Tailored solvers can (potentially) take advantage of new hardware developments
- Cons:
  - Often requires re-write of the original code
  - Reformulated system is factor (P+1) larger than the original system and can be challenging to solve
  - Challenges with increasing time-horizon for ODEs
- Many efforts in the community to automate ISP
  - UQToolkit http://www.sandia.gov/UQToolkit/
  - Sundance http://www.math.ttu.edu/ klong/Sundance/html/
  - Stokhos http://trilinos.sandia.gov/packages/stokhos/
  - ...

NISP pros and cons

- Pros:
  - Easy to use as wrappers around existing codes
  - Embarassingly parallel
- Cons:
  - Most methods suffer from curse of dimensionality $N_q = n^{N_d}$
- Many develoment efforts for smarter sampling approaches and dimensionality reduction
  - (Adaptive) Sparse Quadrature approaches
  - Compressive Sensing
  - ...
- Sampling methods have found very wide spread use in the community
  - DAKOTA http://dakota.sandia.gov/
  - ...

Outline

Uncertain Input Characterization

- Use standard distribution
  - Normal distribution: often a good choice based on Central Limit Theorem
  - Lognormal: when positivity is required
- Infer model parameters from data with inverse problem
- Determine RV from available samples of RV
  - Fit standard distribution to data, e.g. MultiVariate Normal (MVN) approximation
  - Inverse Cumulative Distribution Function (CDF) mapping; Rosenblatt transformation
- Dimensionality reduction for stochastic processes, Karhunen-Loève Expansion (KLE)

Bayesian Inference

- Bayes formula $$\overbrace{P(\mathbf{c}|\mathcal{D})}^{\text{Posterior}} \propto \overbrace{P(\mathcal{D}|\mathbf{c})}^{\text{Likelihood}} \overbrace{P(\mathbf{c})}^{\text{Prior}}$$
- Update prior distribution/knowledge about parameter $\mathbf{c}$ to posterior distribution given data $\mathcal{D}$, using likelihood function $\mathcal{L}(\mathbf{c}) \equiv P(\mathcal{D}|\mathbf{c})$.
- Data $\mathcal{D} = \{d_i\}_{i=1}^N$ - measurements of *some* quantities of interest (QoIs).
- Prior distribution $P(\mathbf{c})$ is based on expert opinion/previous literature.
- Likelihood function measures goodness-of-fit and is the key component that connects the model inputs to measured QoIs, e.g.

$$\mathcal{L}(\mathbf{c}) = P(\mathcal{D}|\mathbf{c}) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\sum_{i=1}^{N} \frac{(d_i - f_i(\mathbf{c}))^2}{2\sigma^2}\right)$$

- Input parameter $\rightarrow$ output QoI functions $f_i(\cdot)$ could be expensive or not even available.
- Usually posterior distribution is not analytically tractable: need to resort to Markov Chain Monte Carlo sampling.

Markov Chain Monte Carlo

Single-site MCMC

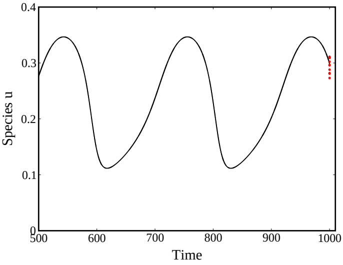
- Set the current chain state $\mathbf{c}$ at an *initial chain state* $\mathbf{c}^{(0)}$,
- Repeat for a predefined number (N_{MCMC}) of times,
  - For $k = 1, \ldots, K$,
    - generate a single-site proposal c'_k from a Gaussian distribution centered at the current chain state value of site c_k with *proposal width* σ_k,
    - compute $\alpha = \min\{1, P(\mathbf{c}'|D)/P(\mathbf{c}|D)\}$,
    - update the current chain state's k-th element $c_k = c'_k$ with probability α,
  - End
- End

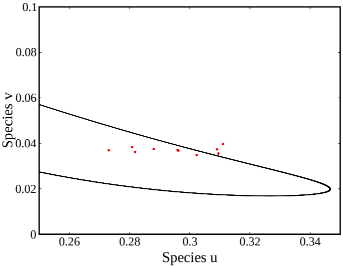
Markov Chain Monte Carlo

Adaptive MCMC [Haario,2002]

- Set the current chain state $\mathbf{c}$ at an *initial chain state* $\mathbf{c}^{(0)}$,
- Repeat for a predefined number (N_{MCMC}) of times,
  - generate a proposal $\mathbf{c}'$ from a multivariate Gaussian distribution centered at the current chain state value $\mathbf{c}$ with *proposal covariance* that is learnt from previous chain states,
  - compute $\alpha = \min\{1, P(\mathbf{c}'|D)/P(\mathbf{c}|D)\}$,
  - update the current chain state $\mathbf{c} = \mathbf{c}'$ with probability α,
- End

Surface Reaction Model: parameter inference



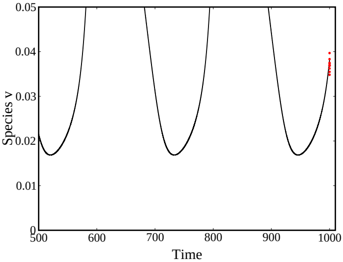


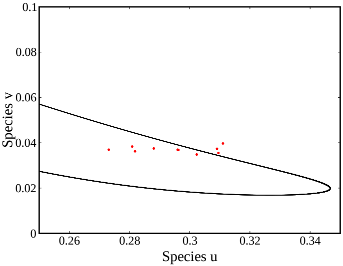
- Synthetic data is generated for model outputs u,v at $T = 1000$.
- Inferring two input parameters a and b using 10 samples on u and v
- Likelihood function

$$P(\mathcal{D}|a,b) = \frac{1}{2\pi\sigma_u\sigma_v} \exp\left(-\sum_{i=1}^{N} \left(\frac{(d_u^{(i)} - u(T;a,b))^2}{2\sigma_u^2} + \frac{(d_v^{(i)} - v(T;a,b))^2}{2\sigma_v^2} \right) \right)$$

- Uncorrelated Gaussian noise model is assumed with standard deviations proportional to the model value $\sigma_u = 0.1u, \sigma_v = 0.1v$.

Surface Reaction Model: parameter inference



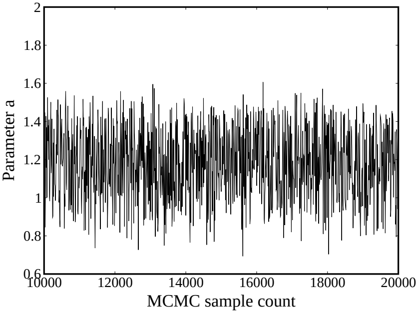


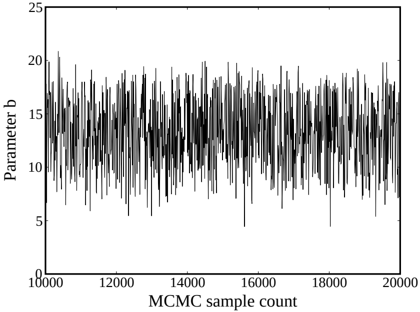
- Synthetic data is generated for model outputs u,v at $T = 1000$.
- Inferring two input parameters a and b using 10 samples on u and v
- Likelihood function

$$P(\mathcal{D}|a,b) = \frac{1}{2\pi\sigma_u\sigma_v}\exp\left(-\sum_{i=1}^{N}\left(\frac{(d_u^{(i)} - u(T;a,b))^2}{2\sigma_u^2} + \frac{(d_v^{(i)} - v(T;a,b))^2}{2\sigma_v^2}\right)\right)$$

- Uncorrelated Gaussian noise model is assumed with standard deviations proportional to the model value $\sigma_u = 0.1u, \sigma_v = 0.1v$.

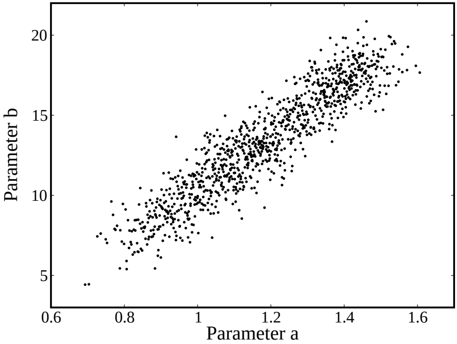
Surface Reaction Model: parameter inference





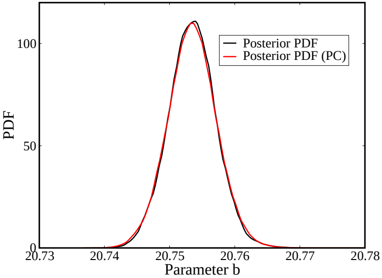
- Posterior distribution based on 20000 adaptive MCMC samples.
- First 4000 samples discarded
- Shown only the second half of the chains

Posterior on inferred parameters



- Width of posterior indicates the amount of uncertainty in the inferred parameters
- Uncertainty can be reduced by taking more data or by reducing noise in measurements

Surface Reaction Model: posterior to PC in 1d



- The posterior describes random variable a with CDF $F(\cdot)$
- CDF transformation $F(a) = \eta$ maps random variable a to uniform$[0,1]$ random variable η.
- $\eta = \Phi(\xi)$ maps uniform η to normal RV ξ
- The inverse CDF enables NISP projection

$$a = \sum_{k=0}^{P} a_k \Psi_k(\xi) \qquad a_k \propto \langle a\Psi_k(\xi)\rangle = \int \underbrace{F^{-1}(\Phi(\xi))}_{a} \Psi_k(\xi) d\xi$$

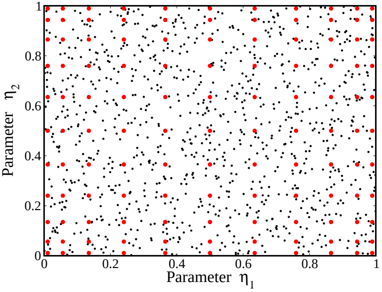
Surface Reaction Model: posterior to PC in multi-d

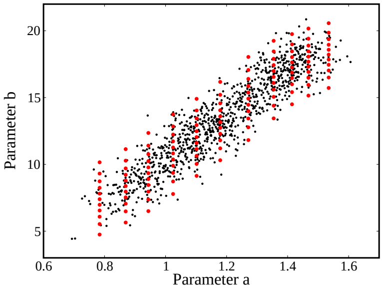
- Rosenblatt transformation maps any (not necessarily independent) set of random variables $(\lambda_1, \ldots, \lambda_n)$ to uniform i.i.d.'s $\{\eta_i\}_{i=1}^n$ (Rosenblatt, 1952).

$$\begin{aligned}
\eta_1 &= F_1(\lambda_1) \\
\eta_2 &= F_{2|1}(\lambda_2|\lambda_1) \\
&\vdots \\
\eta_n &= F_{n|n-1,\ldots,1}(\lambda_n|\lambda_{n-1}, \ldots, \lambda_1)
\end{aligned}$$

- Rosenblatt transformation is a multi-D generalization of 1D CDF mapping.
- Conditional CDFs are harder to evaluate in high dimensions

Projection of Rosenblatt transformed vars onto PCEs



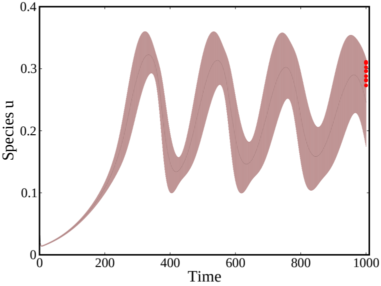


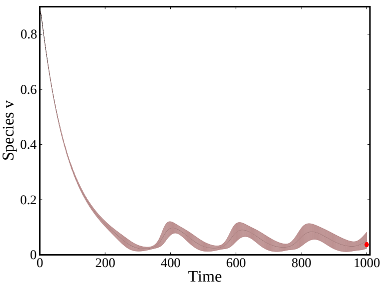
- NISP projection is enabled by inverse Rosenblatt transformation $(a, b) = R^{-1}(\xi_1, \xi_2)$ ensures a well-defined quadrature integration

$$a = \sum_{k=0}^{P} a_k \Psi_k(\xi) \qquad a_k \propto \int \underbrace{R_a^{-1}(\xi)}_{a} \Psi_k(\xi) d\xi$$

$$b = \sum_{k=0}^{P} b_k \Psi_k(\xi) \qquad b_k \propto \int \underbrace{R_b^{-1}(\xi)}_{b} \Psi_k(\xi) d\xi$$

Surface Reaction Model: predictive confidence





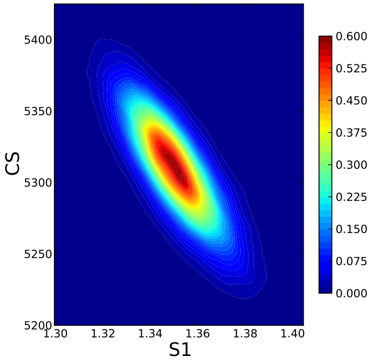
- Uncertainty in inferred input parameters *a* and *b* is pushed through the forward model
  - Using 2D Wiener Hermite NISP Quadrature approach
- Accounts for parametric uncertainty due to data noise

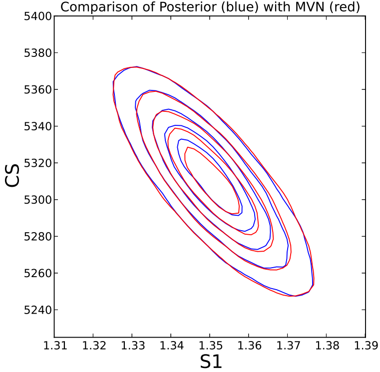
Multivariate Normal Approximation

- Many distributions are unimodal and somewhat shaped like Gaussians
- MultiVariate Normal (MVN) approximations capture the mean and correlation structure of the random variables
- Easy to extract from a set of samples
  - In 1D: just compute mean and standard deviation: $u = u_0 + u_1\xi$
  - Multi-D: Cholesky factorization of covariance

```
# Compute mean parameter values
par_mean = numpy.mean(samples,axis=0)
# Compute the covariance
par_cov = numpy.cov(samples,rowvar=0)
# Compute the Cholesky Decomposition
chol_lower = numpy.linalg.cholesky(par_cov)
```

MVN approximation of Bayesian posterior from MCMC samples





$$S_1 = 1.351 + 0.01367\xi_1$$
$$CS = 5310 - 26.25\xi_1 + 20.26\xi_2$$

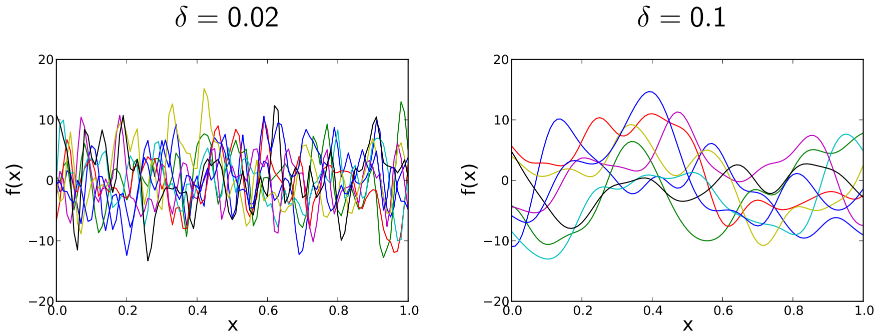
Karhunen-Loève (KL) Expansions

- Assume stochastic process $F(x, \theta)$
- With covariance function $Cov(x_1, x_2)$
- F can be written as

$$F(x, \theta) = \langle F(x, \theta) \rangle_\theta + \sum_{k=1}^{\infty} \sqrt{\lambda_k} F_k(x) \xi_k$$

- $F_k(x)$: eigenfunctions of $Cov(x_1, x_2)$
- λ_k: corresponding eigenvalues, all positive
- ξ_k: uncorrelated random variables, unit variance
  - Samples are obtained by projecting realizations of F onto F_k
  - Generally not independent
    - Special case: for Gaussian F, ξ_k are i.i.d. normal random variables

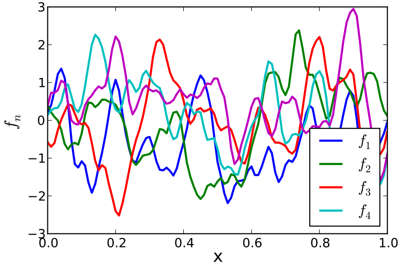
1D Gaussian Process: Realizations



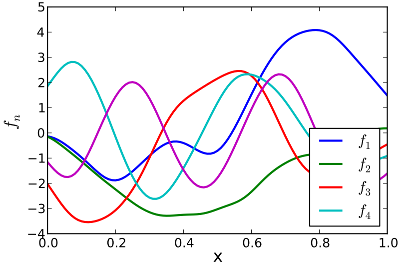
- Covariance $Cov(x_1, x_2) = exp(-(x_1 - x_2)^2/\delta^2)$
- Sample realizations are noisier as correlation length decreases

1D Gaussian Process: KL modes

$\delta = 0.02$

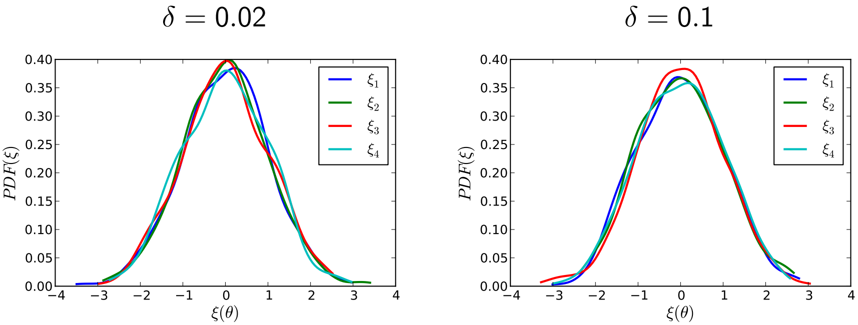


$\delta = 0.1$



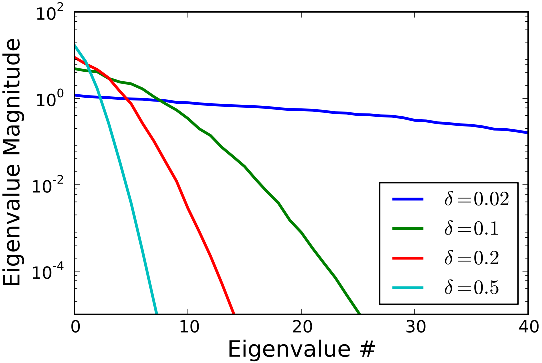
- Eigenmodes of the covariance matrix
  - Data covariance matrix constructed from 512 Gaussian process realizations
- Higher modes are more oscillatory

1D Gaussian Process: KL random variables



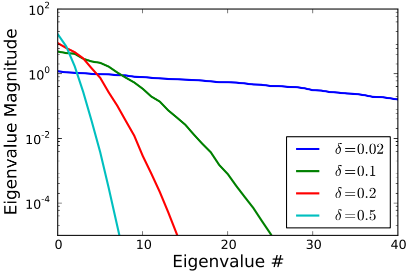
- Random variables obtained by projecting realizations onto KL modes
- Uncorrelated by construction
  - Also independent due to nature of Gaussian Process

1D Gaussian Process: Eigenvalue spectrum



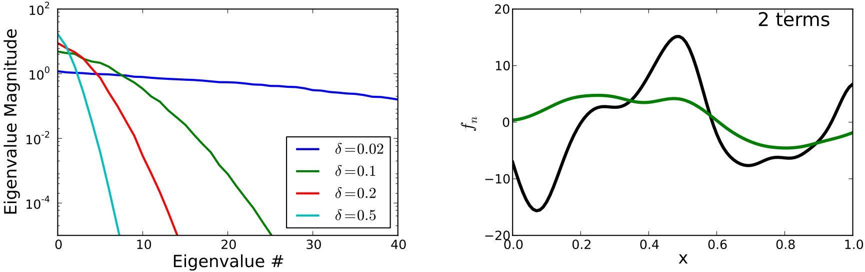
- Eigenvalue spectrum decays more slowly as correlation length decreases
  - More oscillatory modes needed to represent fluctuations in x
- KL expansion generally is truncated after enough modes are included to capture a specified fraction of the total variance

1D Gaussian Process: Reconstructed realizations



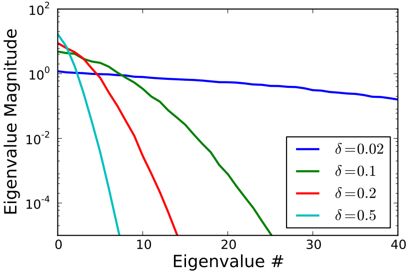
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

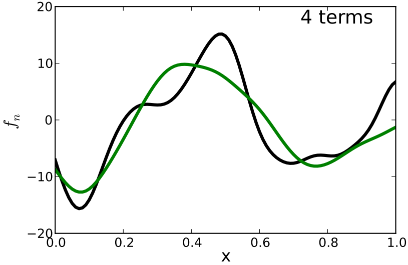
1D Gaussian Process: Reconstructed realizations



- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

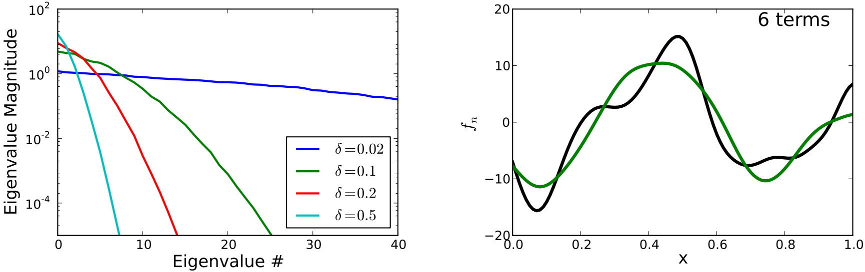
1D Gaussian Process: Reconstructed realizations





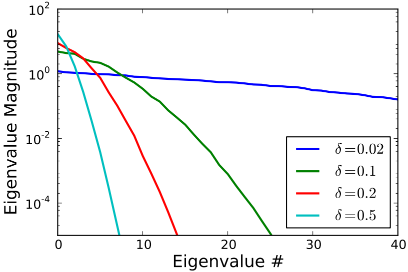
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

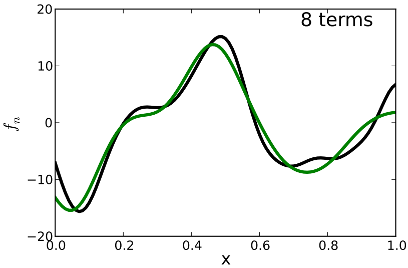
1D Gaussian Process: Reconstructed realizations



- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

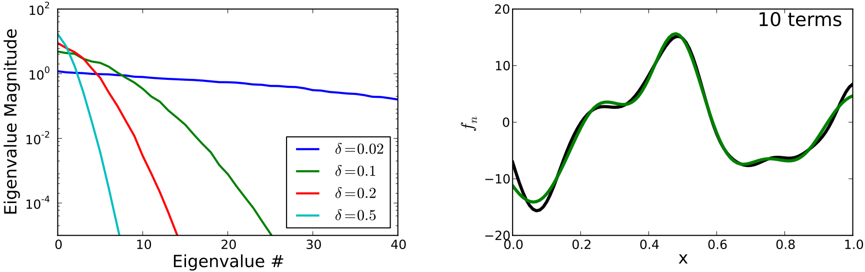
1D Gaussian Process: Reconstructed realizations





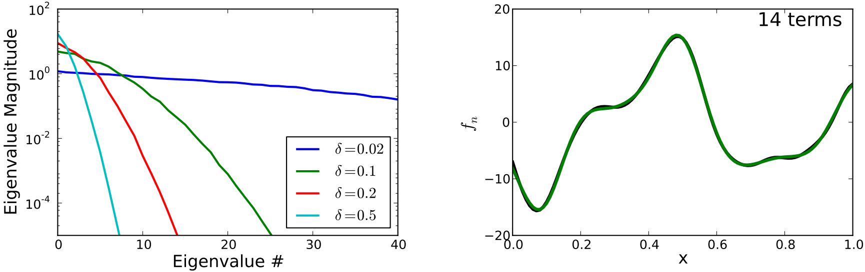
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

1D Gaussian Process: Reconstructed realizations



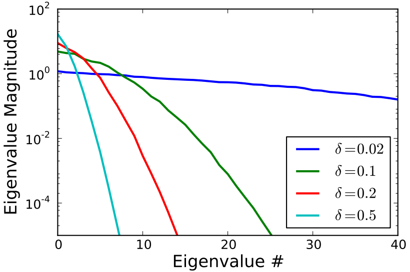
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

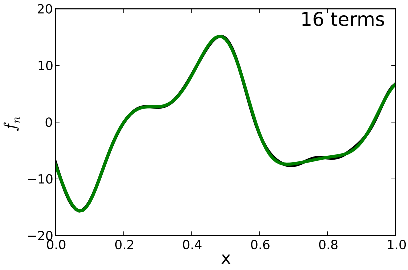
1D Gaussian Process: Reconstructed realizations



- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

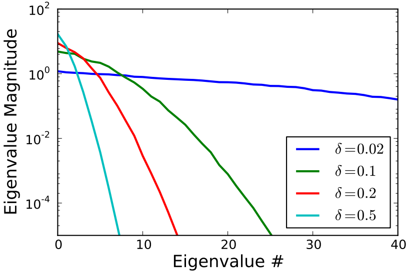
1D Gaussian Process: Reconstructed realizations

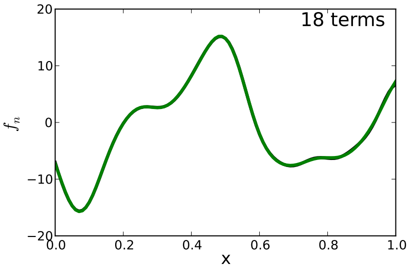




- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

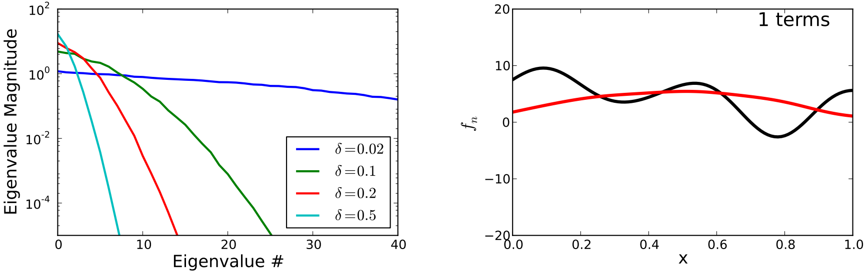
1D Gaussian Process: Reconstructed realizations





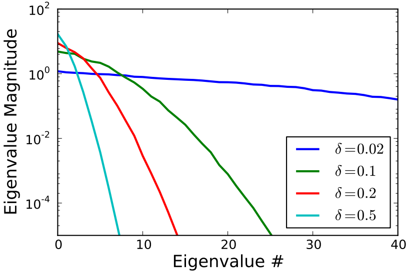
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

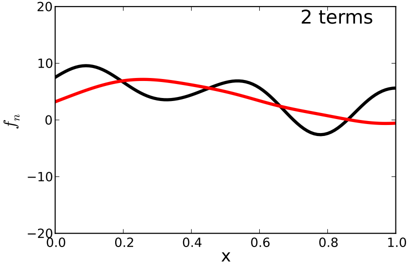
1D Gaussian Process: Reconstructed realizations



- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

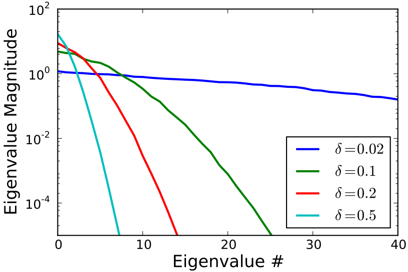
1D Gaussian Process: Reconstructed realizations

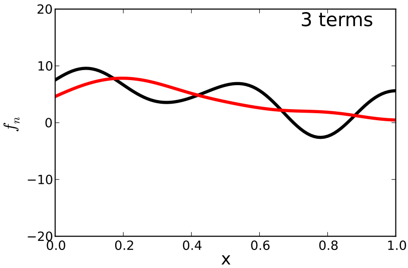




- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

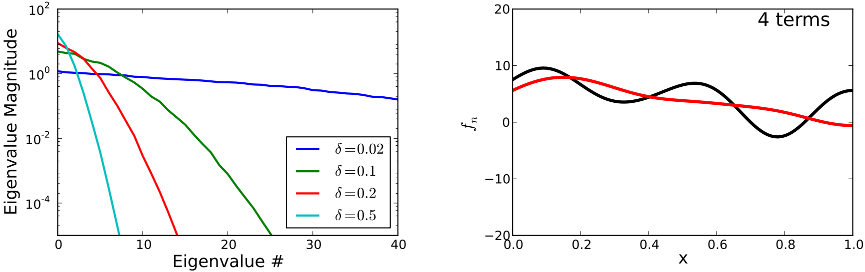
1D Gaussian Process: Reconstructed realizations





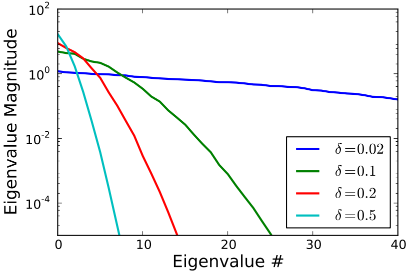
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

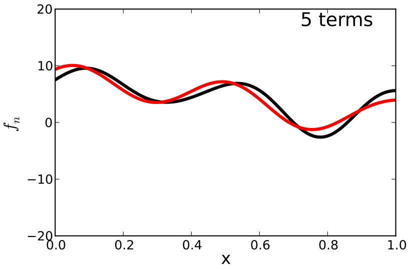
1D Gaussian Process: Reconstructed realizations



- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

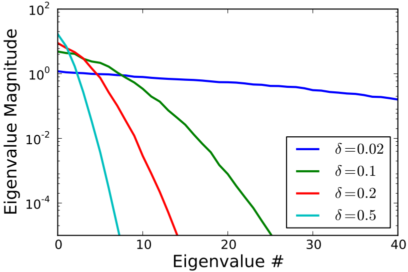
1D Gaussian Process: Reconstructed realizations

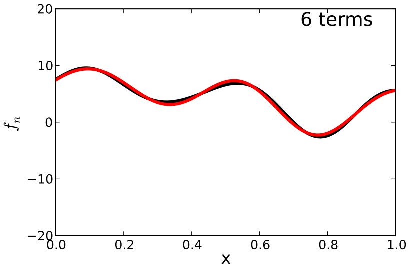




- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

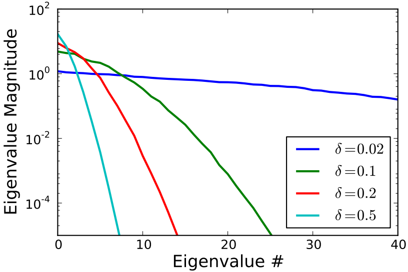
1D Gaussian Process: Reconstructed realizations

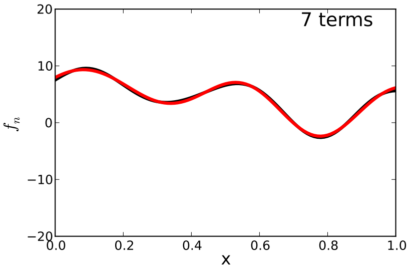




- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

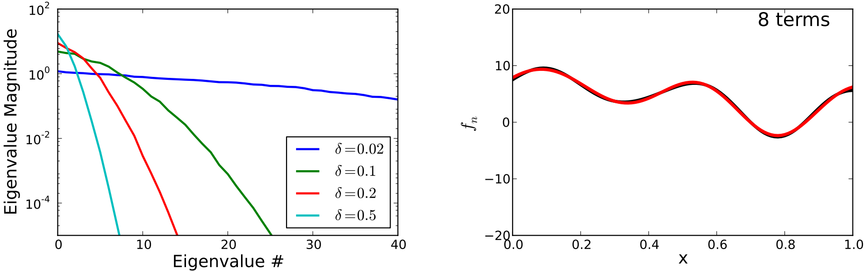
1D Gaussian Process: Reconstructed realizations





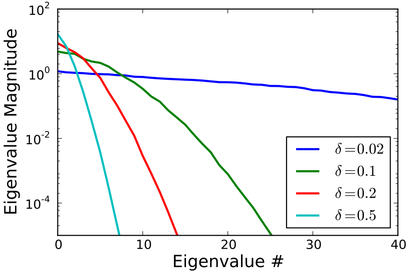
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

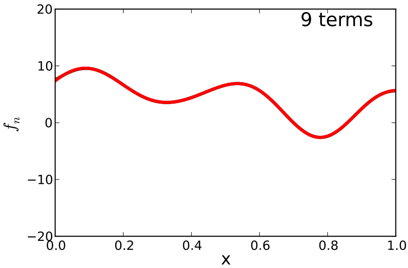
1D Gaussian Process: Reconstructed realizations



- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

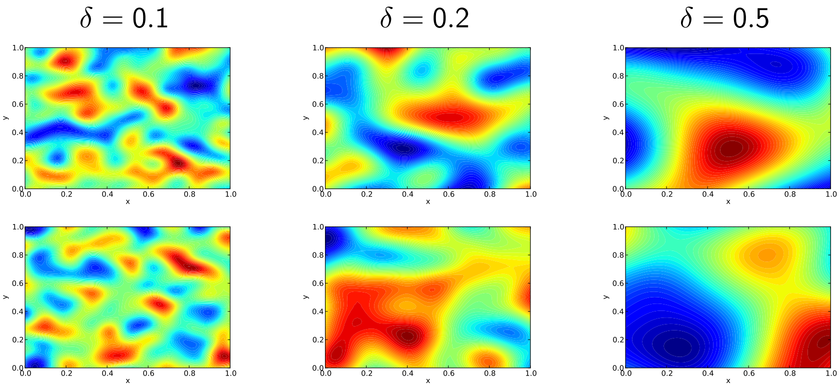
1D Gaussian Process: Reconstructed realizations





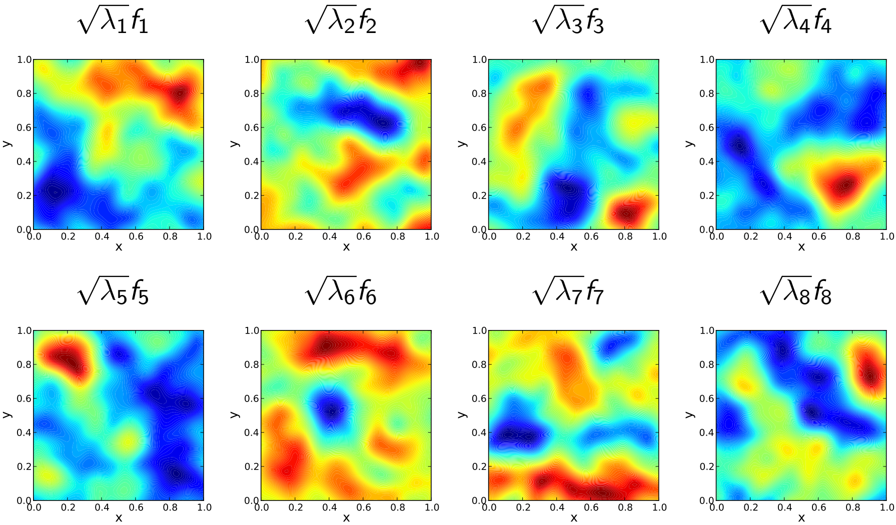
- Large scale features can be resolved with small number of modes
- Smaller scale features require higher modes

KL of 2D Gaussian Process

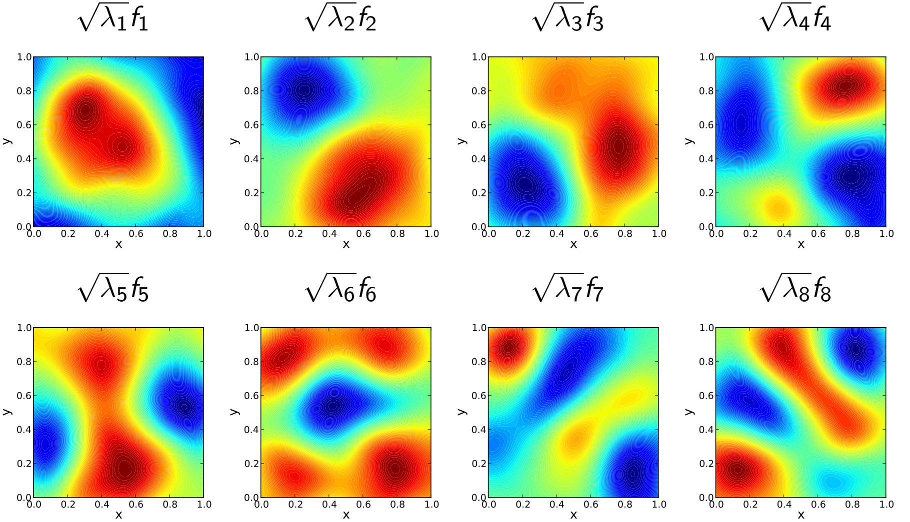


- 2D Gaussian Process with covariance:
  $Cov(x_1, x_2) = exp(-||x_1 - x_2||^2/\delta^2)$
- Realizations are smoother as covariance length δ increases

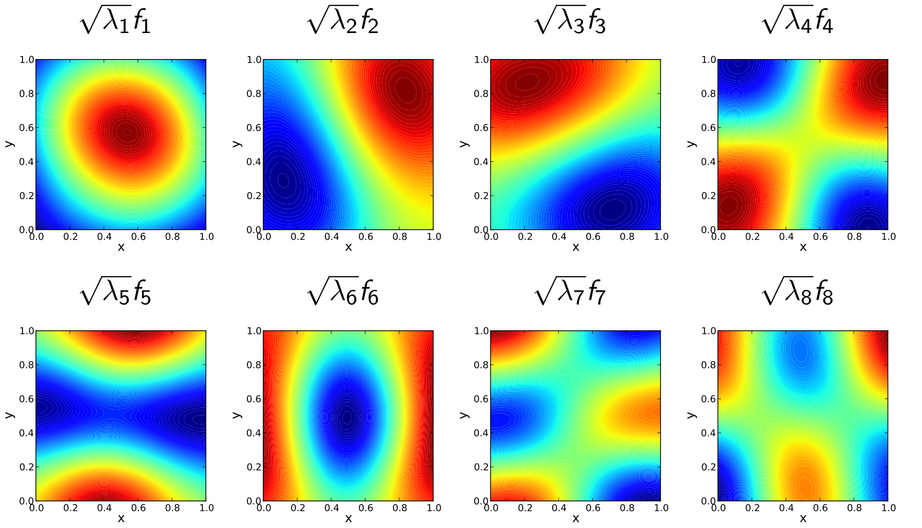
2D KL - Modes for $\delta = 0.1$



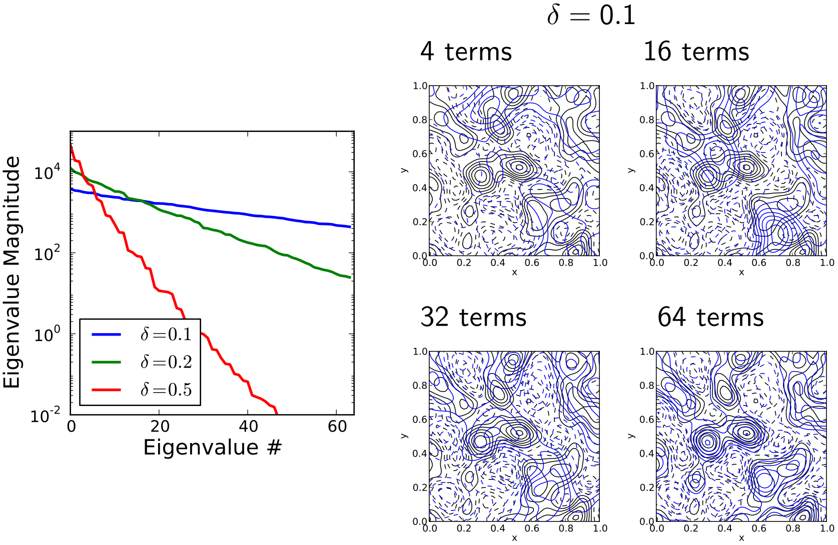
2D KL - Modes for $\delta = 0.2$



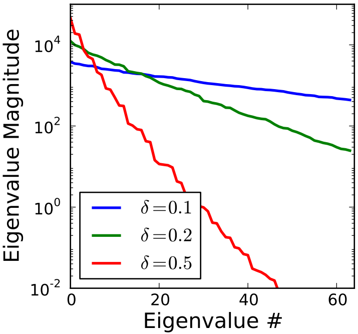
2D KL - Modes for $\delta = 0.5$

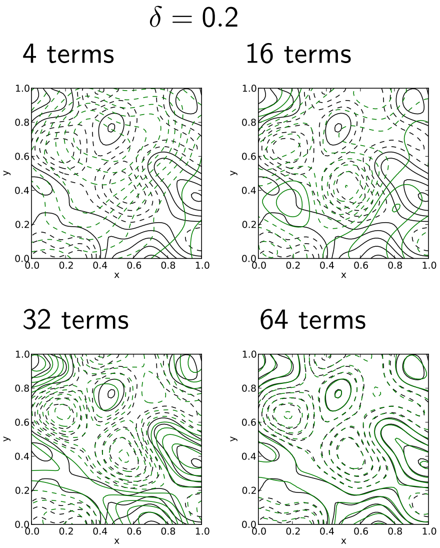


2D KL - eigenvalue spectrum

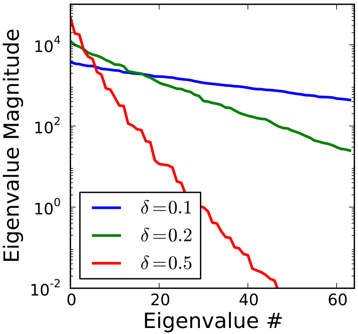


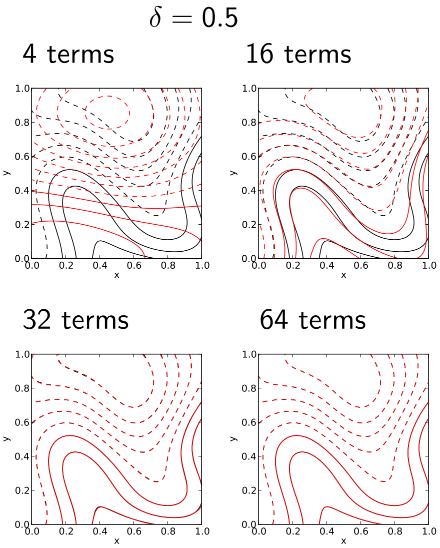
2D KL - eigenvalue spectrum





2D KL - eigenvalue spectrum



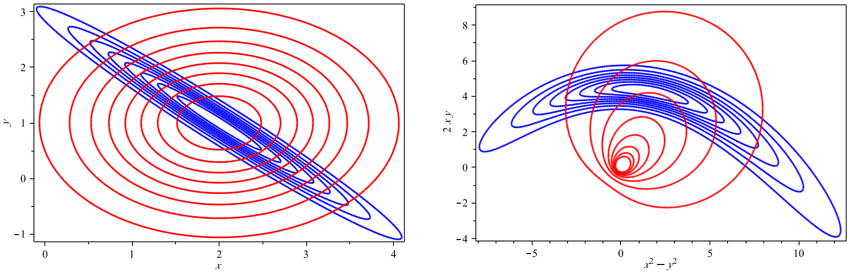


Outline

Uncertainty in Model Inputs

- Probabilistic UQ requires specification of uncertain inputs
- Require joint PDF on input space
- PDF can be found given data
- Typically such PDFs are not available from the literature
  - Summary information, e.g. nominals and bounds, is usually available
- Uncertainty in computational predictions can depend strongly on detailed structure of the missing parametric PDF
- Need a procedure to reconstruct a PDF consistent with available information in the absence of the raw data
  - "Data Free" Inference (DFI) (Berry *et al.*, JCP 2012)

The strong role of detailed input PDF structure

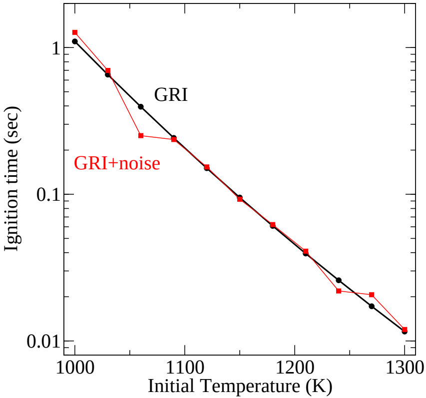


- Simple nonlinear algebraic model $(u, v) = (x^2 - y^2, 2xy)$
- Two input PDFs, $p(x, y)$
  - same nominals/bounds
  - different correlation structure
- Drastically different output PDFs
  - different nominals and bounds

Generate ignition "data" using a detailed model+noise

- Ignition using a detailed chemical model for methane-air chemistry
- Ignition time versus Initial Temperature
- Multiplicative noise error model
- 11 data points:

$$
\begin{aligned}
d_i &= t_{\mathrm{ig},i}^{\mathrm{GRI}}(1+\sigma\epsilon_i) \\
\epsilon &\sim N(0,1)
\end{aligned}
$$



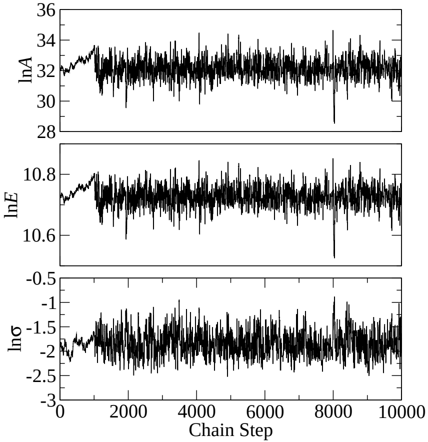
Fitting with a simple chemical model

- Fit a global single-step irreversible chemical model

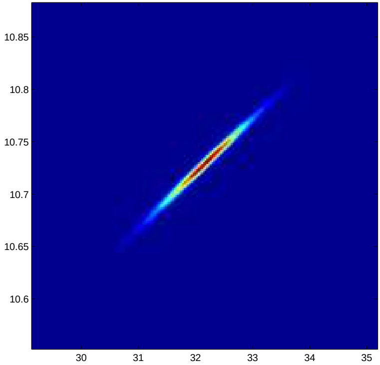
$$\mathrm{CH_4} + 2\mathrm{O_2} \rightarrow \mathrm{CO_2} + 2\mathrm{H_2O}$$

$$\begin{aligned} \mathfrak{R} &= [\mathrm{CH_4}][\mathrm{O_2}]k_f \\ k_f &= A\exp(-E/R^o T) \end{aligned}$$

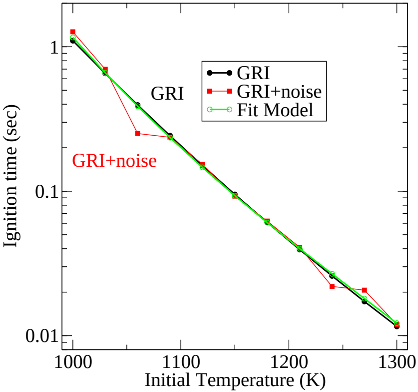
- Infer 3-D parameter vector $(\ln A, \ln E, \ln \sigma)$
- Good mixing with adaptive MCMC when start at MLE



Bayesian Inference Posterior and Nominal Prediction

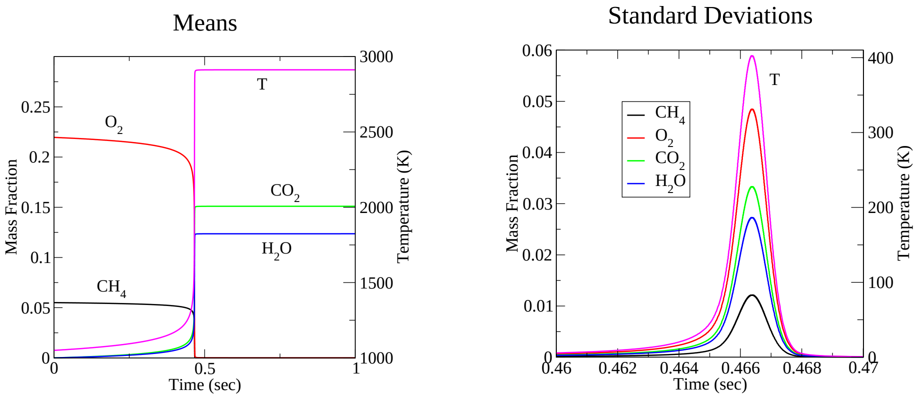


Marginal joint posterior on $(\ln A, \ln E)$ exhibits strong correlation



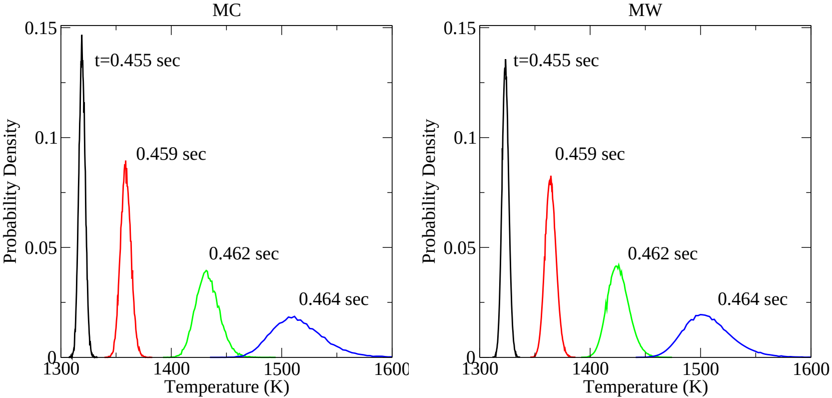
Nominal fit model is consistent with the true model

Correlation Slope χ and Chemical Ignition



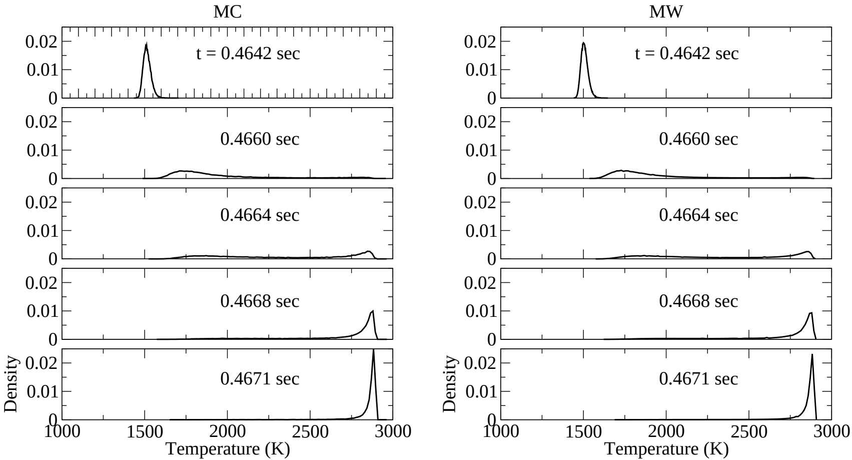
- 4^{th} Order Multiwavelet PC, Multiblock, Adaptive
- $\sigma_{T,\max} \sim 400$ K during ignition transient, $\chi \sim 0.03$

Time evolution of Temperature PDFs in preheat stage



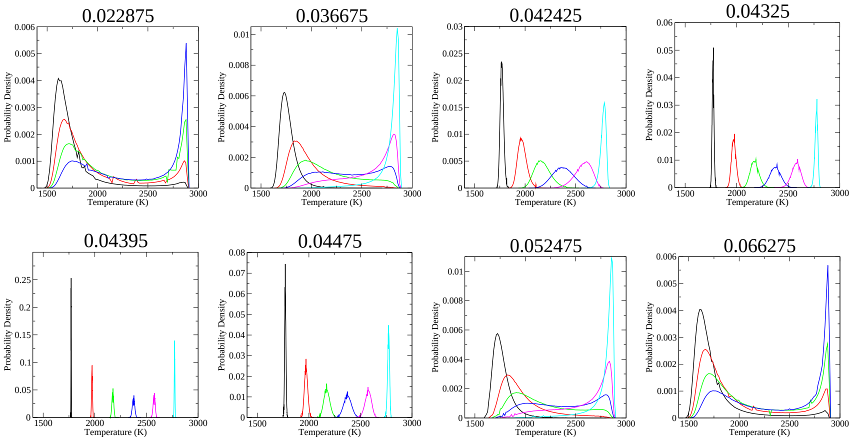
- Similar results from MC (20K samples) and MW PC
- Increased uncertainty, and long high−T PDF tails, in time

Evolution of Temp. PDF – Fast Ignition Transient



- Transition from unimodal to bimodal PDFs
- Leakage of probability mass from pre-heat PDF high–T tail

Time evolution of Temperature PDFs for different χ



- Bimodal solution PDFs for high uncertainty growth
- Unimodal for low uncertainty growth, with $\chi \approx 0.044$

Central Challenge for UQ in Chemical Kinetic Models

- Need joint PDF on model parameters for forward UQ
- Joint PDF structure is crucial
- Joint PDF not available for chemical kinetic parameters
- At best, have
  - Nominal parameter values
  - Bounds, *e.g.* marginal 5%, 95% quantiles
- PDF **can** be constructed by repeating experiments or access to original raw data
  - Neither is feasible
- Is there a way to construct an approximate PDF **without** access to raw data?
  - Yes!

Data Free Inference (DFI)

(Berry *et al.*, JCP, in review)

- Intuition: In the absence of data, the structure of the fit model, combined with the nominals and bounds, implicitly inform the correlation between the parameters
- Goal: Make this information *explicit* in the joint PDF
- DFI: discover a consensus joint PDF on the parameters consistent with given information:
  - Nominal parameter values
  - Bounds
  - The fit model
  - The data range
  - ... potentially other/different constraints

Outline

Schlögl Model is a prototype bistable model

- Reactions

$$A + 2X \underset{a_2}{\overset{a_1}{\rightleftarrows}} 3X$$

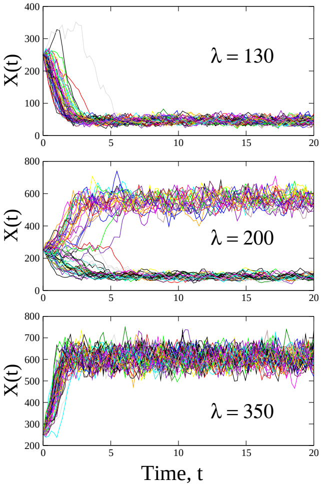
$$B \underset{a_4}{\overset{a_3}{\rightleftarrows}} X$$

- Propensities

$$a_1 = k_1 A X(X-1)/2,$$
$$a_2 = k_2 X(X-1)(X-2)/6,$$
$$a_3 = k_3 B,$$
$$a_4 = k_4 X.$$

- Nominal parameters

$k_1 A$	0.03
k_2	0.0001
$k_3 B = \lambda$	200
k_4	3.5
A	10^5
B	$2 \cdot 10^5$
$X(0)$	250



Polynomial Chaos expansion represents any random variable as a polynomial of a standard random variable

- Truncated PCE: finite dimension n and order p

$$X(\boldsymbol{\theta}) \simeq \sum_{k=0}^{P} c_k \Psi_k(\boldsymbol{\eta})$$

with the number of terms $P+1 = \frac{(n+p)!}{n!p!}$.

- $\boldsymbol{\eta} = (\eta_1, \cdots, \eta_n)$ standard i.i.d. r.v.
  Ψ_k standard orthogonal polynomials
  c_k spectral modes.
- Most common standard Polynomial-Variable pairs:
  (continuous) Gauss-Hermite, Legendre-Uniform,
  (discrete) Poisson-Charlier.

[Wiener, 1938; Ghanem & Spanos, 1991; Xiu & Karniadakis, 2002]

Polynomial Chaos expansion represents any random variable as a polynomial of a standard random variable

- Truncated PCE: finite dimension n and order p

$$X(\theta) \simeq \sum_{k=0}^{P} c_k \Psi_k(\boldsymbol{\eta})$$

with the number of terms $P+1 = \frac{(n+p)!}{n!p!}$.

- $\boldsymbol{\eta} = (\eta_1, \cdots, \eta_n)$ standard i.i.d. r.v.
  Ψ_k standard orthogonal polynomials
  c_k spectral modes.
- Most common standard Polynomial-Variable pairs: (continuous) Gauss-Hermite, Legendre-Uniform, (discrete) Poisson-Charlier.

[Wiener, 1938; Ghanem & Spanos, 1991; Xiu & Karniadakis, 2002]

Polynomial Chaos expansion represents any random variable as a polynomial of a standard random variable

- Truncated PCE: finite dimension n and order p

$$X(\boldsymbol{\theta}) \simeq \sum_{k=0}^{P} c_k \Psi_k(\boldsymbol{\eta})$$

  with the number of terms $P + 1 = \frac{(n+p)!}{n!p!}$.

- $\boldsymbol{\eta} = (\eta_1, \cdots, \eta_n)$ standard i.i.d. r.v.
  Ψ_k standard orthogonal polynomials
  c_k spectral modes.

- Most common standard Polynomial-Variable pairs:
  (continuous) Gauss-Hermite, Legendre-Uniform,
  (discrete) Poisson-Charlier.

[Wiener, 1938; Ghanem & Spanos, 1991; Xiu & Karniadakis, 2002]

Galerkin Projection is typically needed

PC expansion: $X(\boldsymbol{\theta}) \simeq \sum_{k=0}^{P} c_k \Psi_k(\boldsymbol{\eta}) = g_{\mathcal{D}}(\boldsymbol{\eta})$

Orthogonal projection: $c_k = \frac{\langle X(\boldsymbol{\theta})\Psi_k(\boldsymbol{\eta})\rangle}{\langle \Psi_k^2(\boldsymbol{\eta})\rangle}$

- Intrusive Spectral Projection (ISP)
  - ⋆ Direct projection of governing equations
  - ⋆ Leads to deterministic equations for PC coefficients
  - ∗ No explicit governing equation for SRNs
- Non-intrusive Spectral Projection (NISP)
  - ⋆ Sampling based
  - ⋆ No explicit evolution equation for X needed
  - ∗ Galerkin projection not well-defined for SRNs

Galerkin Projection is typically needed

PC expansion: $X(\boldsymbol{\theta}) \simeq \sum_{k=0}^{P} c_k \Psi_k(\boldsymbol{\eta}) = g_{\mathcal{D}}(\boldsymbol{\eta})$

Orthogonal projection: $c_k = \frac{\langle X(\boldsymbol{\theta})\Psi_k(\boldsymbol{\eta})\rangle}{\langle \Psi_k^2(\boldsymbol{\eta})\rangle}$

- Intrusive Spectral Projection (ISP)
  - ⋆ Direct projection of governing equations
  - ⋆ Leads to deterministic equations for PC coefficients
  - ∗ No explicit governing equation for SRNs
- Non-intrusive Spectral Projection (NISP)
  - ⋆ Sampling based
  - ⋆ No explicit evolution equation for X needed
  - ∗ Galerkin projection not well-defined for SRNs

Galerkin Projection is typically needed

PC expansion: $X(\boldsymbol{\theta}) \simeq \sum_{k=0}^{P} c_k \Psi_k(\boldsymbol{\eta}) = g_{\mathcal{D}}(\boldsymbol{\eta})$

Orthogonal projection: $c_k = \frac{\langle X(\boldsymbol{\theta})\Psi_k(\boldsymbol{\eta})\rangle}{\langle \Psi_k^2(\boldsymbol{\eta})\rangle}$

- Intrusive Spectral Projection (ISP)
  - ⋆ Direct projection of governing equations
  - ⋆ Leads to deterministic equations for PC coefficients
  - ∗ No explicit governing equation for SRNs
- Non-intrusive Spectral Projection (NISP)
  - ⋆ Sampling based
  - ⋆ No explicit evolution equation for X needed
  - ∗ Galerkin projection not well-defined for SRNs

Karhunen-Loève decomposition reduces stochastic process to a finite number of random variables

- KL decomposition:

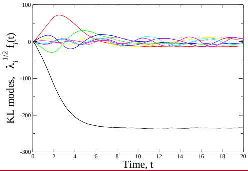
$$X(t,\theta) = \bar{X}(t) + \sum_{n=1}^{\infty} \xi_n(\theta)\sqrt{\lambda_n} f_n(t)$$

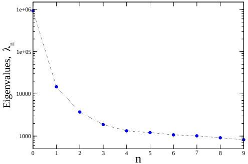
- Uncorrelated, zero-mean KL variables:

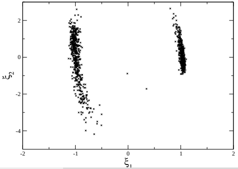
$$\langle \xi_n \rangle = 0, \qquad \langle \xi_n \xi_m \rangle = \delta_{nm}$$

- SSA(continuum) $\longleftrightarrow$ KL(discrete)

$$X(t) \longleftrightarrow \boldsymbol{\xi} = (\xi_1, \xi_2, \dots)$$

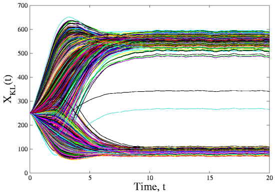




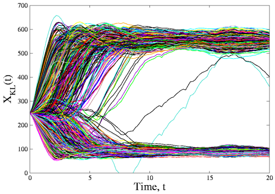


K-L decomposition captures each realization

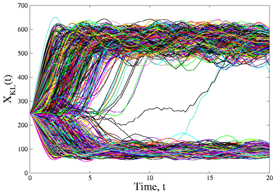
KL decomposition with 2 modes



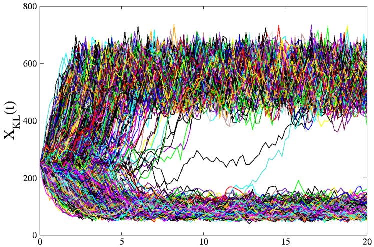
KL decomposition with 5 modes



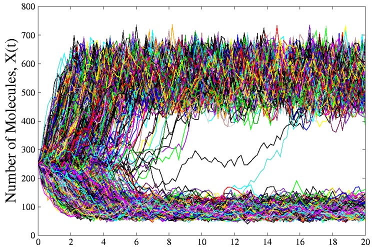
KL decomposition with 10 modes



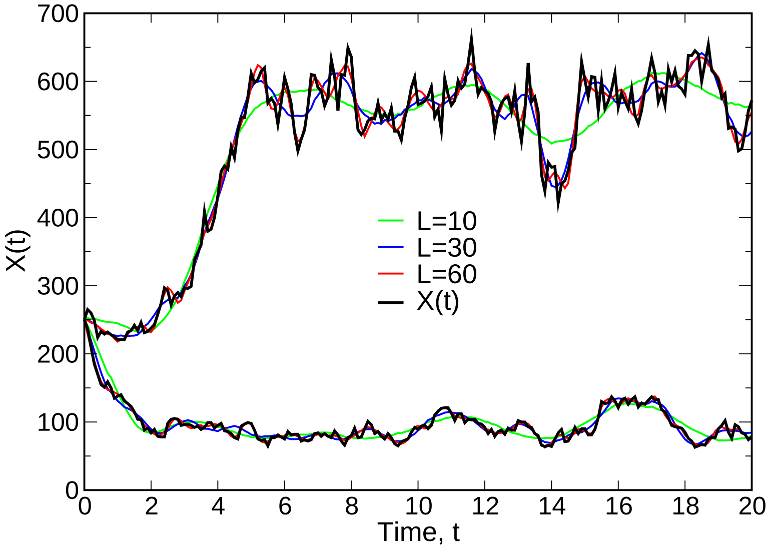
KL decomposition with 100 modes



SSA Realizations



K-L decomposition captures each realization



PC expansion of a random vector

$$\boldsymbol{\xi} = \sum_{k=0}^{P} \mathbf{c}_k \Psi_k(\boldsymbol{\eta})$$

Galerkin projection

$$\mathbf{c}_k = \frac{\langle \boldsymbol{\xi} \Psi_k(\boldsymbol{\eta}) \rangle}{\langle \Psi_k^2(\boldsymbol{\eta}) \rangle}$$

is not well-defined,
since $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ do not belong to the same stochastic space.

PC expansion of a random vector

$$\boldsymbol{\xi} = \sum_{k=0}^{P} \mathbf{c}_k \Psi_k(\boldsymbol{\eta})$$

Galerkin projection

$$\mathbf{c}_k = \frac{\langle \boldsymbol{\xi} \Psi_k(\boldsymbol{\eta}) \rangle}{\langle \Psi_k^2(\boldsymbol{\eta}) \rangle}$$

is not well-defined,
since $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ do not belong to the same stochastic space.

Need a map $\boldsymbol{\xi} \leftrightarrow \boldsymbol{\eta}$.

Rosenblatt transformation

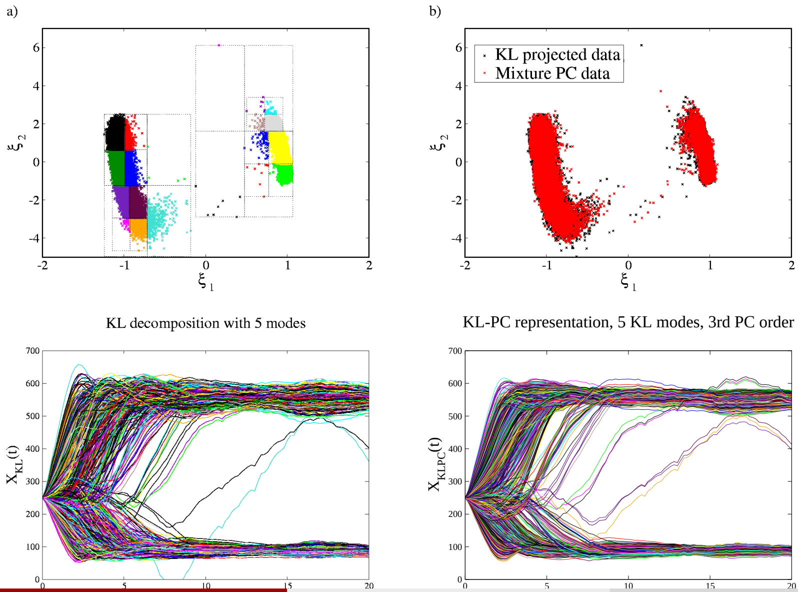
- Rosenblatt transformation maps any (not necessarily independent) set of random variables $(\xi_1, \ldots, \xi_n)$ to uniform i.i.d.'s $\{\eta_i\}_{i=1}^n$ (Rosenblatt, 1952).

$$\begin{aligned}
\eta_1 &= F_1(\xi_1) \\
\eta_2 &= F_{2|1}(\xi_2|\xi_1) \\
\eta_3 &= F_{3|2,1}(\xi_3|\xi_2, \xi_1) \\
&\vdots \\
\eta_n &= F_{n|n-1,\ldots,1}(\xi_n|\xi_{n-1}, \ldots, \xi_1)
\end{aligned}$$

- Inverse Rosenblatt transformation $\boldsymbol{\xi} = R^{-1}(\boldsymbol{\eta})$ ensures a well-defined quadrature integration

$$\langle \xi_i \Psi_k(\boldsymbol{\eta}) \rangle = \int R^{-1}(\boldsymbol{\eta})_i \Psi_k(\boldsymbol{\eta}) d\boldsymbol{\eta}$$

KL+PC+Data Partitioning represent the dynamics of a bimodal process



Outline

Advanced Topics

- Sensitivity analysis
- Domain decomposition methods; multiwavelets
- Adaptive Sparse Quadrature
- Stochastic preconditioning (time rescaling)
- Data Free Inference (DFI)
- Bayesian Compressive Sensing (BCS)
- PCEs with random coefficients versus Gaussian Processes
- Model uncertainty, comparison, selection

... Stay Tuned ...