Argonne NATIONAL LABORATORY

... for a brighter future

Hybrid Sampling-Sensitivity Methods for Uncertainty Quantification.

Mihai Anitescu

With Oleg Roderick, Paul Fischer, Won-Sik Yang

ANS 2009

U.S. Department of Energy

UChicago Argonne LLC

Office of Science
U.S. DEPARTMENT OF ENERGY

A U.S. Department of Energy laboratory managed by UChicago Argonne, LLC

Validation: Certifying code by code or code by data

- Models cannot perfectly match data. An error *model* is needed.
  - Uncertainty Model – “Uncertainty Quantification” (physical parameters, discretization errors, geometrical parameters, model error).
  - Uncertainty-to-output propagation: characterize the system outputs over the entire uncertainty range – some form of sampling/sensitivity?
- Objective of our small study – Rough answers to following questions:
  - Are current uncertainty models suitable for high fidelity calculations (particularly thermohydraulics)?
  - How do I reduce the number of expensive code samples?
  - Where should I spend my effort in uncertainty reduction?

Salient Question: High Dimensional Uncertainty Propagation in Very Large Scale System

- How to approximate the stochastic distribution of functions over very large uncertain spaces?

$$0 = F(x, p): R^n \times R^q \rightarrow R^n \Rightarrow x = x(p)$$

$$p = p(\omega) \sim D(\mu(\omega)) \Rightarrow E_\omega \left[G(x(p)) \right] = ?$$

- Example: F can be the equations of TH+Neutronics, p the physical parameters. (p ~ 1000 … 10000 … 100000).
- G can be (note that x(p) cannot even be stored):
  - Max output temperature, max centerline temperature.
  - A characteristic function $G = \chi\left[a \le T \le b\right]$ which computes the probability of the max T to be in a given range.

High dimensional Approximation… The curse of dimensionality

- Computing the statistics is strongly connected to approximating the function G over the high-dimensional space.
- In a high dimensional problem, this is subject to the “curse of dimensionality” – the fact that the sample density is decreasing exponentially with the dimension of the problem.
- Classical solutions:
  - Sampling : global but slow – note that it does not answer the COD.
  - Sensitivity analysis fast but local and hard to adjust. What if the precision in assessment is insufficient? Higher-order? Hard to develop and implement.
- Can we create a method that efficiently uses the advantages of both methods, and is adjustable?

Our answer: Output learning with derivative information

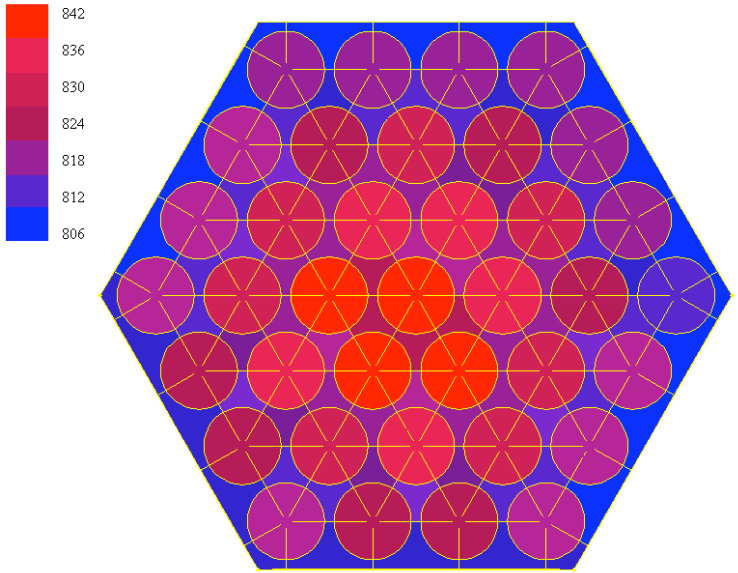
- Setup: outputs (100) << uncertain parameters (10^5-10^6)<< physical state space size (10^9-10^12).
- Outputs, e.g.
  - *Coolant fuel temperature.*
  - *Peak fuel temperature*
- Output (parameters)=? – it is a machine learning problem.
- The main observation: gradients of an output can be computed in 5*the effort (sometimes FAR less) for the output by adjoints, but provide NP times more information !!!
- This is radically different compared to machine learning from non-computing experiment data.
- Hypothesis: Using the derivative information, we can reduce the number of samples needed to compute the entire mapping.
- Hybrid sampling-sensitivity approach.

Outline

- Test case
- Uncertainty quantification for sodium-cooled thermohydraulics.
- Polynomial Regression with derivatives
- Numerical Results.
- Global sensitivity calculation.
- Automatic differentiation issues

Test case: a simplied model of the nuclear reactor core.

- A coupled system, taking into account multiple physical effects (steady state):



- 3D (finite volumes) model of heat transport by diffusion, convection.
- 1D model of neutronic diffusion in the fuel elements, 4 energy groups.
- 2D model of heat distribution in the the fuel elements.

- Physical properties, such as conductivity, heat capacity, heat transfer coefficient, neutronic macroscopic cross-sections, are dependent on temperature. We attribute the uncertainty of the model to experimental uncertainties in temperature dependencies.

Test case: a simplied model of the nuclear reactor core.

- Heat transport:

$$\int_{\Omega} q'''dV = \int_{\Omega} K\nabla T(h)\cdot\vec{n}dS + \int_{\partial\Omega} \rho c_p T\vec{u}\cdot\vec{n}dS$$

- Neutron interaction:

$$-\nabla\cdot D_g\nabla\phi_g + \Sigma_{tg}\phi_g - \sum_{g\neq g'}\Sigma_{g\to g'}\phi_{g'} = -\frac{1}{\kappa}\sum_{g}\chi_g\sum_{g'}\nu_{g'}\Sigma_{f\,g}\phi_{g'}$$

- Heat distribution:

$$\nabla\cdot K\nabla T_{pin} = q'''$$

- Temperature dependencies:

$$R := R(T)(1+\Delta R(T))$$

$$\Sigma := \Sigma(T)(1+\Delta R(T))$$

Test case: Equations

$$0 = -\nabla \cdot K \nabla T - \rho c_p \vec{u} \nabla T + q'''$$

Interface

- When discretized we obtain

Fuel

$$\Phi = h(T_J - T_{surface}) = c_p \frac{T_{surface} - T_I}{D/2}$$

$$\int_\Omega q''' dV = \sum_J 2 \frac{hc_p}{hD + 2c_p} T_J - \sum_J 2 \frac{hc_p}{hD + 2c_p} T_I + \sum_{I-,I+} K \frac{1}{H_{I \to I-,+}} \int_{\partial\Omega} n dS T_{I-,+} + \ldots$$

$$\ldots - \sum_{I-,I+} K \frac{1}{H_{I \to I-,+}} \int_{\partial\Omega} n dS T_I$$

Fluid

$$0 = \sum_J K \frac{1}{H_{I \to J}} \int_{\partial\Omega} n dS T_J + \sum_{J+,J-} K \frac{1}{H_{I \to J}} \int_{\partial\Omega} n dS T_{J+,J-} - \sum_J K \frac{1}{H_{I \to J}} \int_{\partial\Omega} n dS T_I + \ldots$$

$$- \sum_{J+,J-} K \frac{1}{H_{I \to J}} \int_{\partial\Omega} n dS T_I + \frac{1}{2} \sum_J \rho c_p \int_{\partial\Omega} \vec{u} n dS T_J + \frac{1}{2} \sum_{J+,J-} \rho c_p \int_{\partial\Omega} \vec{u} n dS T_{J+,J-} + \ldots$$

$$+ \frac{1}{2} \sum_J \rho c_p \int_{\partial\Omega} \vec{u} n dS T_I + \frac{1}{2} \sum_{J+,J-} \rho c_p \int_{\partial\Omega} \vec{u} n dS T_I + \sum_{J^*} 2 \frac{hc_p}{hD + 2c_p} T_{J^*} - \sum_{J^*} 2 \frac{hc_p}{hD + 2c_p} T_I$$

Test case: Uncertainty in the Physical Parameters
1) Temperature dependent

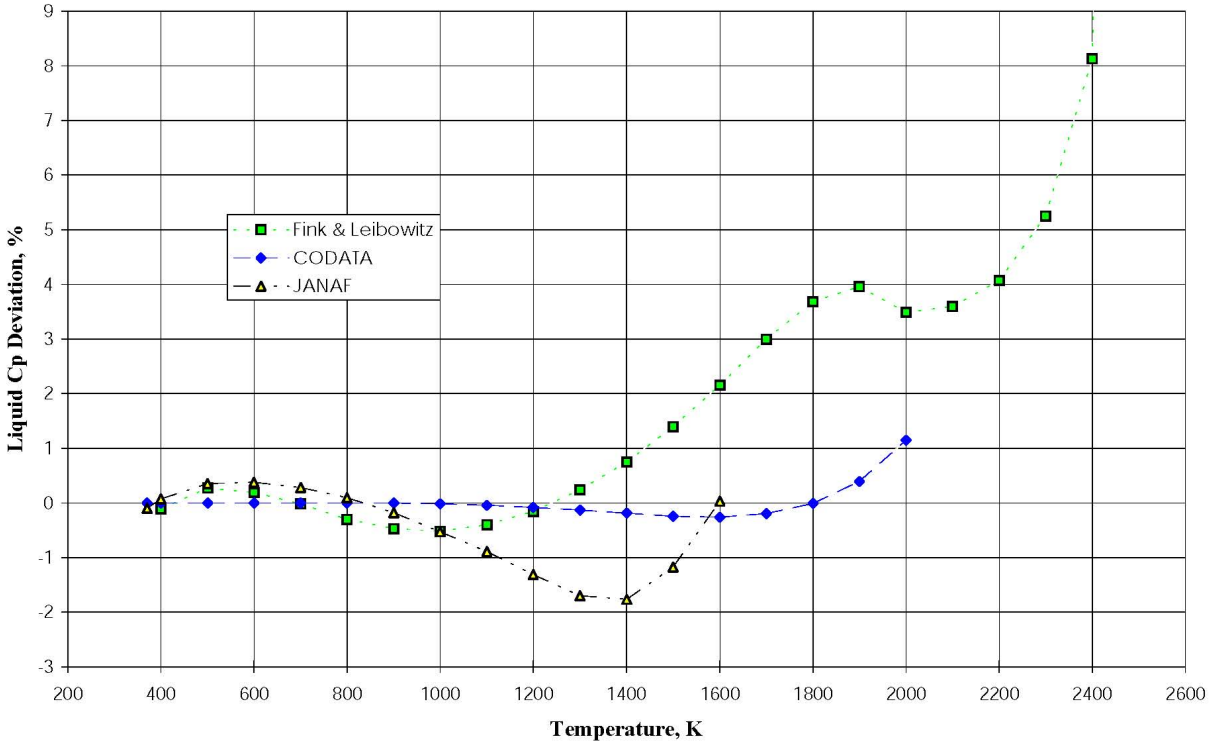
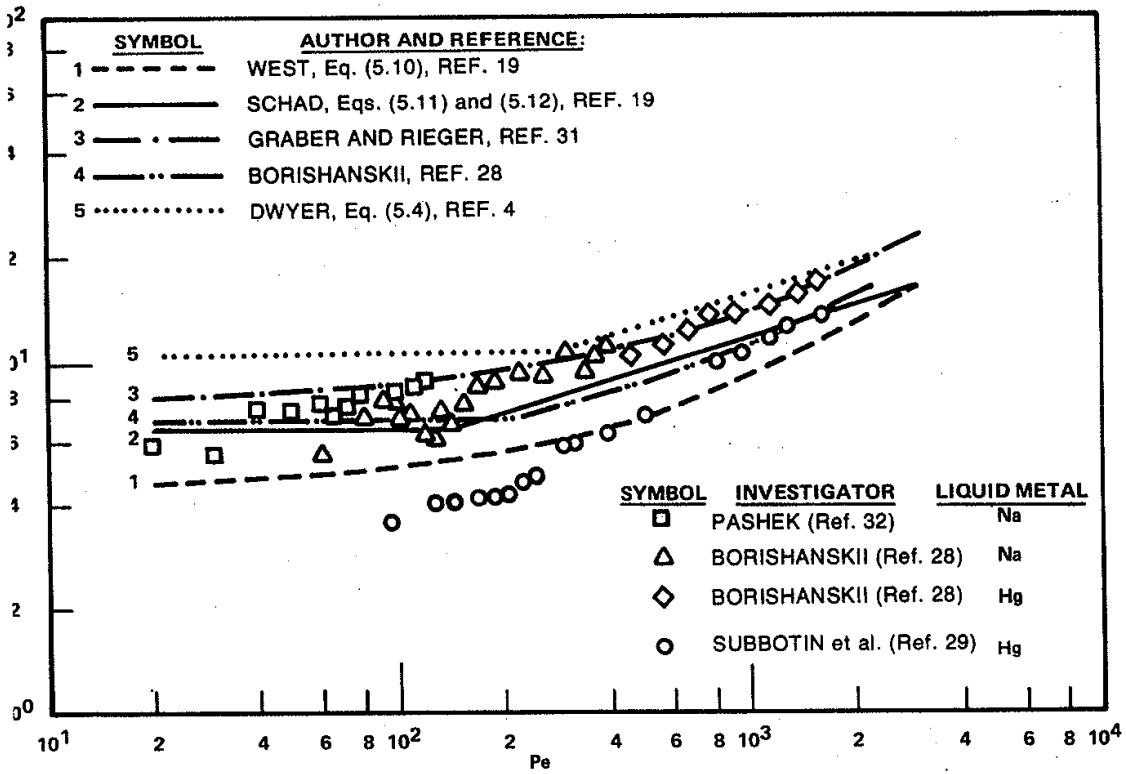


Fig. 1.1-11 Deviations of Values from Other Assessments from the Recomended Values for the Heat Capacity at Constant Pressure of Liquid Sodium

Test case: uncertainty in heat transfer coefficient(Nusselt) 2) Other parameter dependent (Pe)



. 5-6. Comparison of predicted and experimentally measured heat transfer results (P/D =).

- Sometimes as bad as a factor of 2!

Uncertainty Quantification

- Representing the uncertainty in the physical parameters

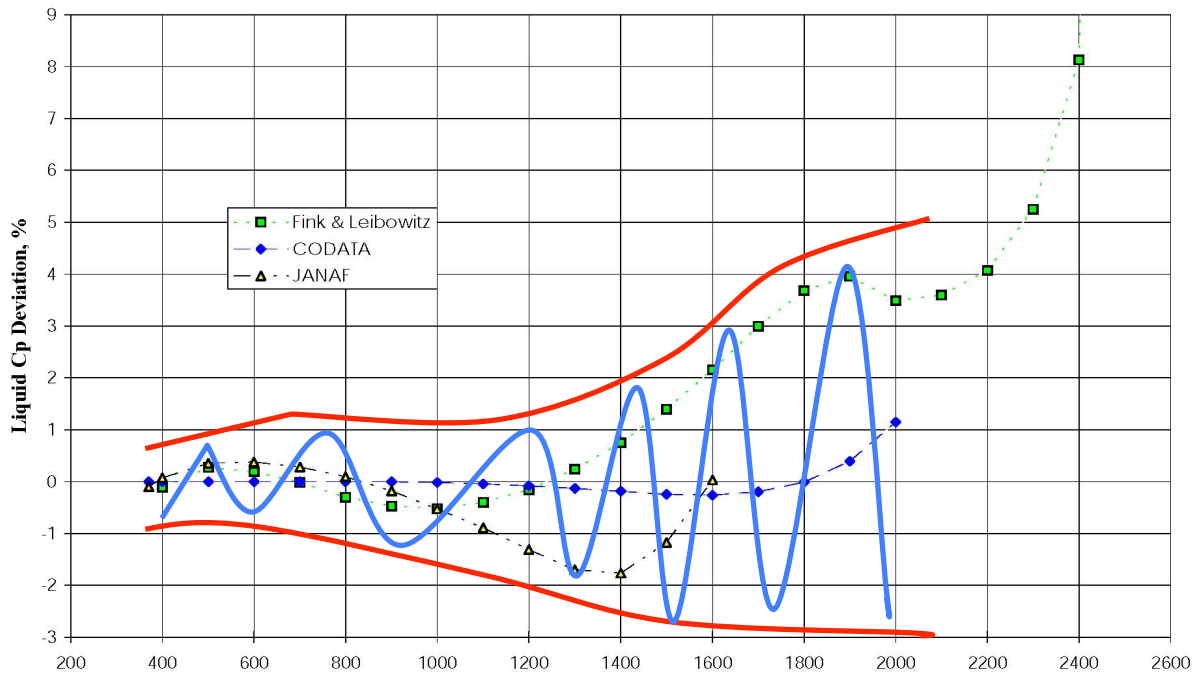
Uncertainty in fluid flow core

- The parameters\material properties *R* of the model include heat capacity c_p, heat conductivity *K for the coolant and fuel; convective heat transfer coefficient h.*; and 4-group crossection.
- A fixed-point iteration procedure *R:=R(T) , T:=T(R)* is used to couple the dependence of the temperature distribution on the material parameters, and the dependence of the material parameters on temperature.
- Uncertainty in the performance *J(T)* of the nuclear reactor is attributed to the uncertainty in the values of parameters *R, and it is temperature-dependent*.
- Note: the available temperature-dependencies are built as a best fit to experimental data + uncertainties, but this results in an uncertainty band, even if the data warrants finer representation, such as no/little oscillation ..

$$c_p \approx 1.6582 - 8.470 \cdot 10^{-4} T + 4.4541 \cdot 10^{-7} T^2 - 2992.6 T^{-2}$$

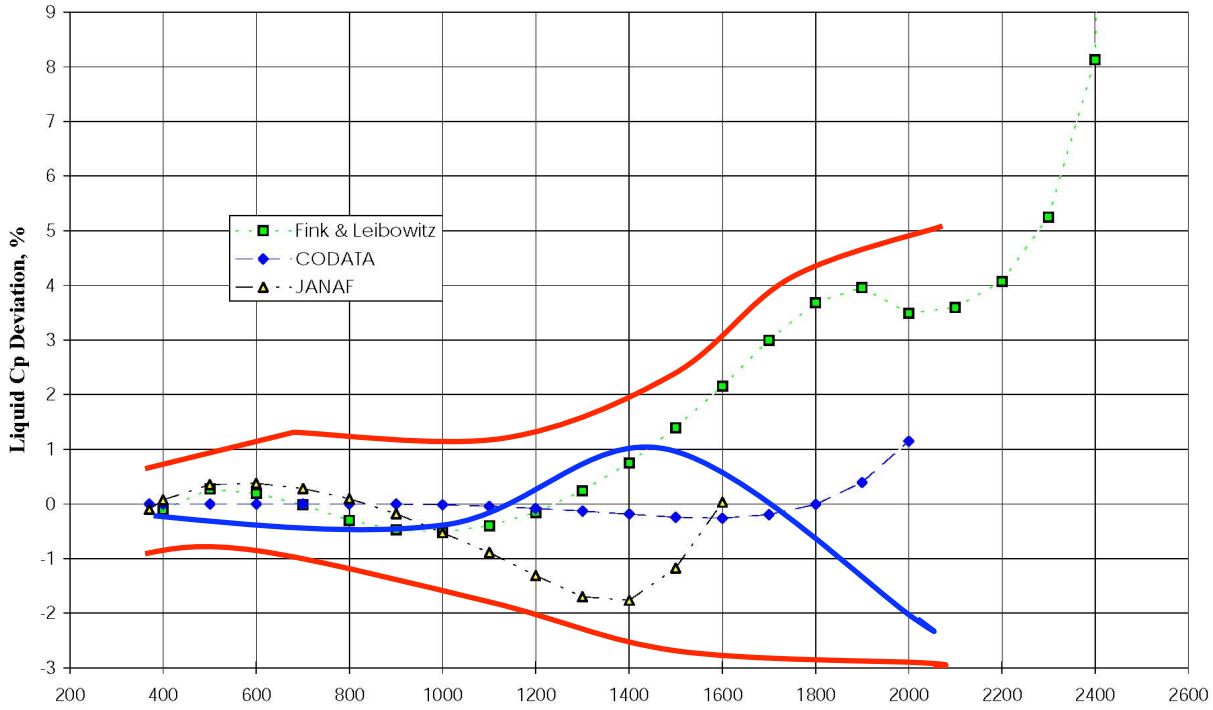
with uncertainty $\frac{\Delta c_p}{c_p}$ estimated at 0.1% at 300 K, 3% at 1000 K, 8% at 2000 K.

Current Representation of Uncertainty



- Allows for realizations that CLEARLY cannot occur, with a phenomenal unjustified growth in uncertainty space.
- Also note that for high resolution, it may include spatial complexity by coupling where you do not expect – so not suitable for high fidelity calc.
- May not want to solve it with hifi and should not solve it.

A more reasonable model



- Capture smoothness --- significant reduction of the uncertainty space.
- Effectively low-order interpolation.

Representation of temperature-dependent uncertainty

- Assume a temperature-dependent structure for the uncertainty:

$$R = \left(\sum_i r^{(i)} T^i \right) \cdot \left(1 + \alpha^{(0)} C^{(0)}(T) + \alpha^{(1)} C^{(1)}(T) + \alpha^{(2)} C^{(2)}(T)) + \ldots\right)$$

$$h = \left(\sum_i r^{(i)} T^i \right) \cdot \left(1 + \alpha^{(0)} C^{(0)}(Pe(T)) + \alpha^{(1)} C^{(1)}(Pe(T)) + \alpha^{(2)} C^{(2)}(Pe(T))) + \ldots\right)$$

in the Chebyshev polynomial basis $C^{(0)}(T) = 1 \quad C^{(1)}(T) = T + 1 \quad C^{(2)}(T) = 2T^2 - 1$

$$C^{(3)}(T) = 4T^3 - 3T^2$$

- With no oscillations in uncertainty, use 2nd order expansion, resulting in 3 uncertainty parameters per thermo-dynamical property.

$$R = \sum_i (r^{(i)} + \alpha^{(i)}) T^i$$

Representation of temperature-dependent uncertainty.

- Find the validity region for the uncertainty coefficients {α} by random sampling. Start with a large uniform sample of values, reject the points that violate the uncertainty condition $\Delta R / R \le \xi\%(T)$

$$c_p \approx 1.6582 - 8.470 \cdot 10^{-4} T + 4.4541 \cdot 10^{-7} T^2 - 2992.6 T^{-2}$$

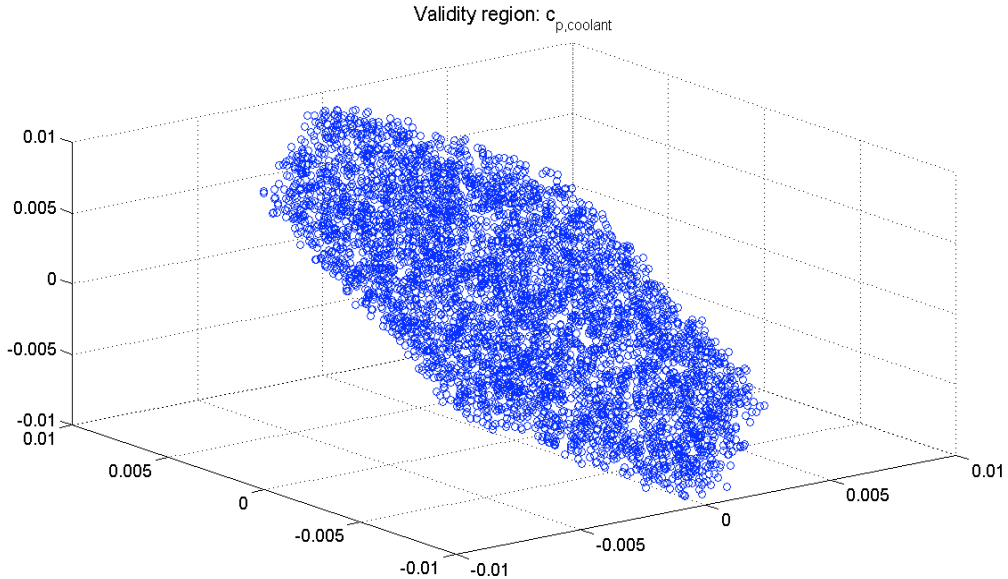
with uncertainty $\frac{\Delta c_p}{c_p}$ estimated at 0.1% at 300 K, 3% at 1000 K, 8% at 2000 K.

- In the multiplicative uncertainty model,

$$C = C(T), C = C(Pe(T))$$

- In the new representation the parameters become but the validity region is not necessarily rectangular

$$\{\alpha_{C_1}^{(i)}\}\{\alpha_{C_2}^{(i)}\}\ldots\{\alpha_{Cn}^{(i)}\}$$



- Uncertainty region with a uniform prior.
- Arguable, but all uncertainty analyses have a subjective component.

Polynomial regression with derivatives (PRD)

- Truly, a novel machine learning technique for massive computer simulations environments.
- Objective: learn the multidimensional function J(α) from well-chosen sample simulations. Choose α run code compute J and its derivatives.

Basis construction: Stochastic finite element method

- A surrogate model is an explicit approximation $J \approx \hat{J}(\alpha)$ in some basis $\Psi(\alpha)$.
- For uncertainty representation: Stochastic Finite Element Method (SFEM, truly spectral element for the uncertainty):

- Choose a set of multi-variable orthogonal polynomials Ψ. Use some subset $\{\Psi_q\}$ to approximate the output function: $J \approx \hat{J} = \sum_q x_q \Psi_q$

- The coefficients P in the definition of each polynomial are chosen to satisfy the orthogonality condition in some measure π:

$$\int_\Omega \Psi_p \Psi_q d\pi = 0 \qquad p \neq q$$

$$\Psi(\alpha_1, \alpha_2, \ldots) = \prod_i H^{(k_i)}(\alpha_i)$$

- For Gaussian probability measure, the basis is a set of Hermite polynomials:

$$H^{(0)}(\alpha) = 1 \qquad H^{(1)}(\alpha) = 2\alpha \qquad H^{(2)}(\alpha) = 4\alpha^2 - 1$$

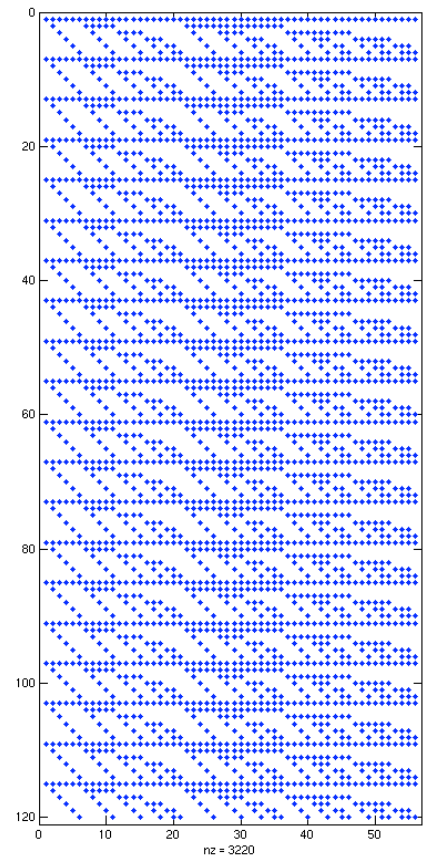
$$H^{(3)}(\alpha) = 8\alpha^3 - 12\alpha \qquad H^{(4)}(\alpha) = 16\alpha^4 - 48\alpha^{12} + 12$$

- The coefficients x_q are found by collocation/regression.

Related idea: Collocation for SFEM in uncertainty quantification.

- Collocation equation: evaluate the basis polynomials at the sample points in the parameter space, run full model to compute the outputs *S* at the sample points, assemble the collocation system Ψx=S:

$$\begin{pmatrix} \Psi(S_1) \\ \Psi(S_2) \\ \vdots \\ \Psi(S_m) \end{pmatrix} X = \begin{pmatrix} J(S_1) \\ J(S_{2)} \\ \vdots \\ J(S_m) \end{pmatrix}$$

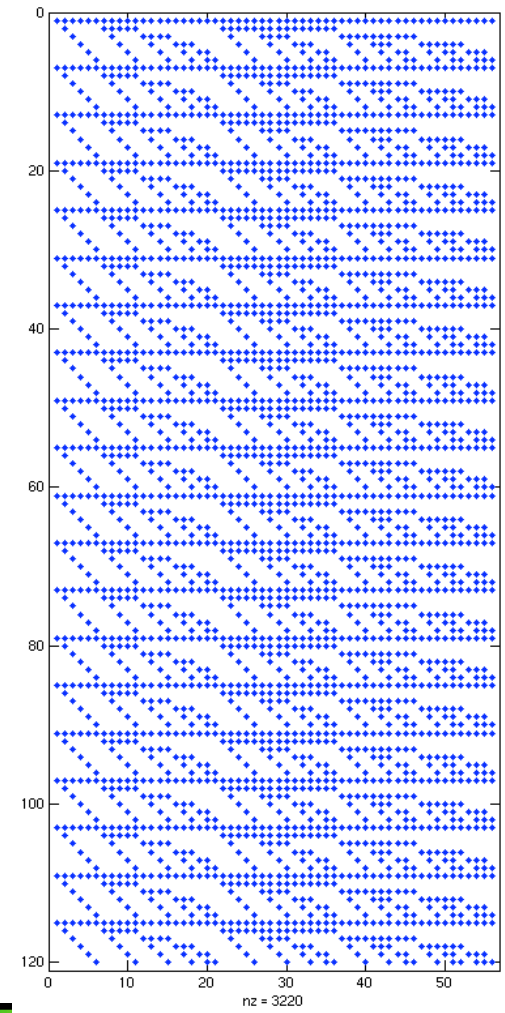


- It is truly a surface response approach.
- Issue: we would like to use high-order polynomials. The number of sample points required to assemble Ψ grows rapidly.

Using Derivatives

$$\begin{pmatrix} \Psi(S_1) \\ \frac{\partial}{\partial \alpha}\Psi(S_1) \\ \Psi(S_2) \\ \frac{\partial}{\partial \alpha}\Psi(S_1) \\ \vdots \\ \Psi(S_n) \\ \frac{\partial}{\partial \alpha}\Psi(S_n) \end{pmatrix} x = \begin{pmatrix} J(S_1) \\ \frac{\partial}{\partial \alpha}J(S_1) \\ J(S_2) \\ \frac{\partial}{\partial \alpha}J(S_1) \\ \vdots \\ J(S_n) \\ \frac{\partial}{\partial \alpha}J(S_n) \end{pmatrix}$$

- Not square, solve by regression (PRD)
- The only interaction with the physics code is only by the right hand side.
- If we implement the adjoint wisely, we can get NP times more information for not even on extra function evaluation cost !!!



Basis pruning/ how to sample

- Still subject to “curse of dimensionality”, (degree^d)
- Collocation/regression choices:
  - “Full” basis vs. “truncated” basis.
  - “Tall” Ψ with over-sampling vs. “square” matrix Ψ with a minimal number of sample points

- Goal-oriented basis: polynomials of high degree are only included for “important” variables. Importance is defined as sensitivity of the output function to a particular parameter.

- Goal-oriented sample set: mostly an open question, especially when derivative is also involved. Sample points may be chosen: in the directions of highest sensitivity of the output function; for the best condition of Ψ; for optimal approximation error; for the best condition of Λ.

Approximating the output of the model

- For a moderate number of parameters (3-15), a good choice is “tall” matrix, “truncated” basis.

- Possible definitions of “importance” of a parameter $r^{(i)}$:
  - Derivative (at some “typical” temperature distribution): $\left|\partial J / \partial r^{(i)}\right|$
  - Derivative adjusted by parameter variance: $\left|\partial J / \partial r^{(i)}\right| \sigma_i$

- We start with a full basis of order 3, separate the variables, by “importance”, into groups I, II and III of sizes $n_I > n_{II} >> n_{III}$. We allow polynomials that include variables from group III to have degree 3; allow the polynomials that include variables from group II have degree 2; only keep polynomials of degree 1 in the variables from group I.

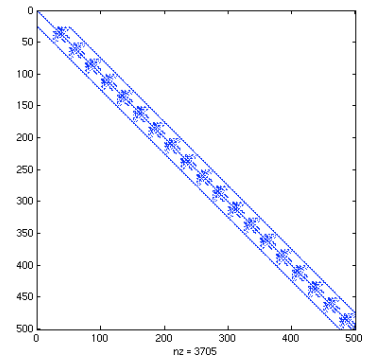
Computing derivatives using adjoints

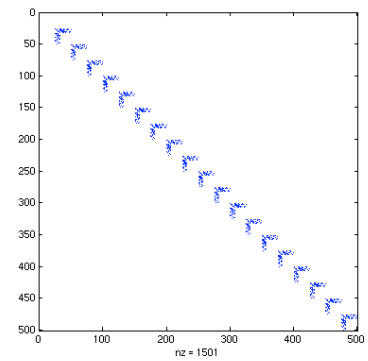
- The dependencies $J(T), T(\Lambda), \Lambda(R,T), R(T,\alpha)$ can be studied directly, by random sampling.

- The derivative $\nabla_\alpha J$ can be used for sensitivity analysis.

- Derivative using the adjoint method:
  - Start with an algebraic form of the flux equilibrium equation: $F(T,\alpha) = 0$ with
  - Assemble a system for the adjoint variable λ:
  - Evaluate the expression:

$$\nabla_\alpha J = -\lambda^T \nabla_\alpha F$$



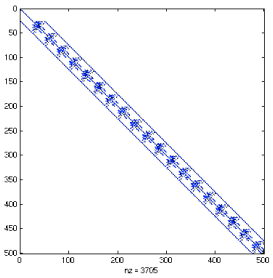


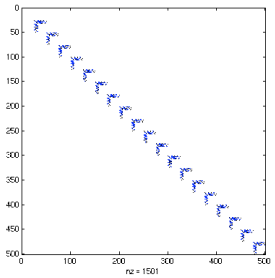
Adjoints.

- Consider the finite volumes equation in the form $F(T_n(\boldsymbol{\alpha}), R(T_{n-1}, \boldsymbol{\alpha})) = 0$

$$F(T_n, R(T_{n-1})) = \boldsymbol{\Lambda}(R) \cdot T_n - \mathbf{B} = 0$$

$$T_n \approx T_{n-1}$$





- Differentiate to obtain

$$\left(\frac{\partial F}{\partial T_n} + \frac{\partial F}{\partial R} \cdot \frac{\partial R}{\partial T_n}\right) \cdot \frac{dT_{n-1}}{d\boldsymbol{\alpha}} + \frac{\partial F}{\partial R} \cdot \frac{\partial R}{\partial \boldsymbol{\alpha}} = 0$$

- We need two partial derivatives: $\dfrac{\partial F}{\partial T_n} = \boldsymbol{\Lambda}$

$$\frac{\partial F}{\partial R} = \frac{\partial \boldsymbol{\Lambda}}{\partial R} \cdot T_n \,.$$

- We have assembled the adjoint variable:

$$\frac{dT_n}{d\boldsymbol{\alpha}} = -\left(\frac{\partial F}{\partial T_n} + \frac{\partial F}{\partial R} \cdot \frac{\partial R}{\partial T_n}\right)^{-1} \cdot \frac{\partial F}{\partial R} \cdot \frac{\partial R}{\partial \boldsymbol{\alpha}} = \boxed{-\lambda^T \cdot \frac{\partial F}{\partial R}} \cdot \frac{\partial R}{\partial \boldsymbol{\alpha}}$$

Adjoints

- The required components of the derivatives arrays

$$\frac{\partial \Lambda}{\partial R^{(I,J)}{}_j}, \quad \frac{\partial R^{(I,J)}{}_j}{\partial \alpha_k}, \quad \frac{\partial R^{(I)}{}_j}{\partial \alpha_k}, \quad \frac{\partial R^{(I,J)}{}_j}{\partial T^{(I)}}$$

  for the volume cells I,J and parameter components R_j, α_k are defined and stored during the last step of the iteration $R_n := R(T_{n-1})$, $T_n := T(R_n)$.

- Finally, the derivative is expressed as: $\frac{dJ}{d\alpha} = \frac{\partial J}{\partial T_n} \cdot \frac{dT_n}{d\alpha}$

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial T} \cdot -\left(\Lambda + \frac{\partial \Lambda}{\partial R} \cdot T_{n\cdot} \cdot \frac{\partial R}{\partial T_n} \right)^{-1} \cdot \frac{\partial \Lambda}{\partial R} \cdot T_{n\cdot} \cdot \frac{\partial R}{\partial \alpha} \Big|_{T_n = T, T_{n-1} = T}$$

- Note: in Matlab, computing all derivatives for a single output typically produces an overhead of 10-40 %.

Performance of PRD model

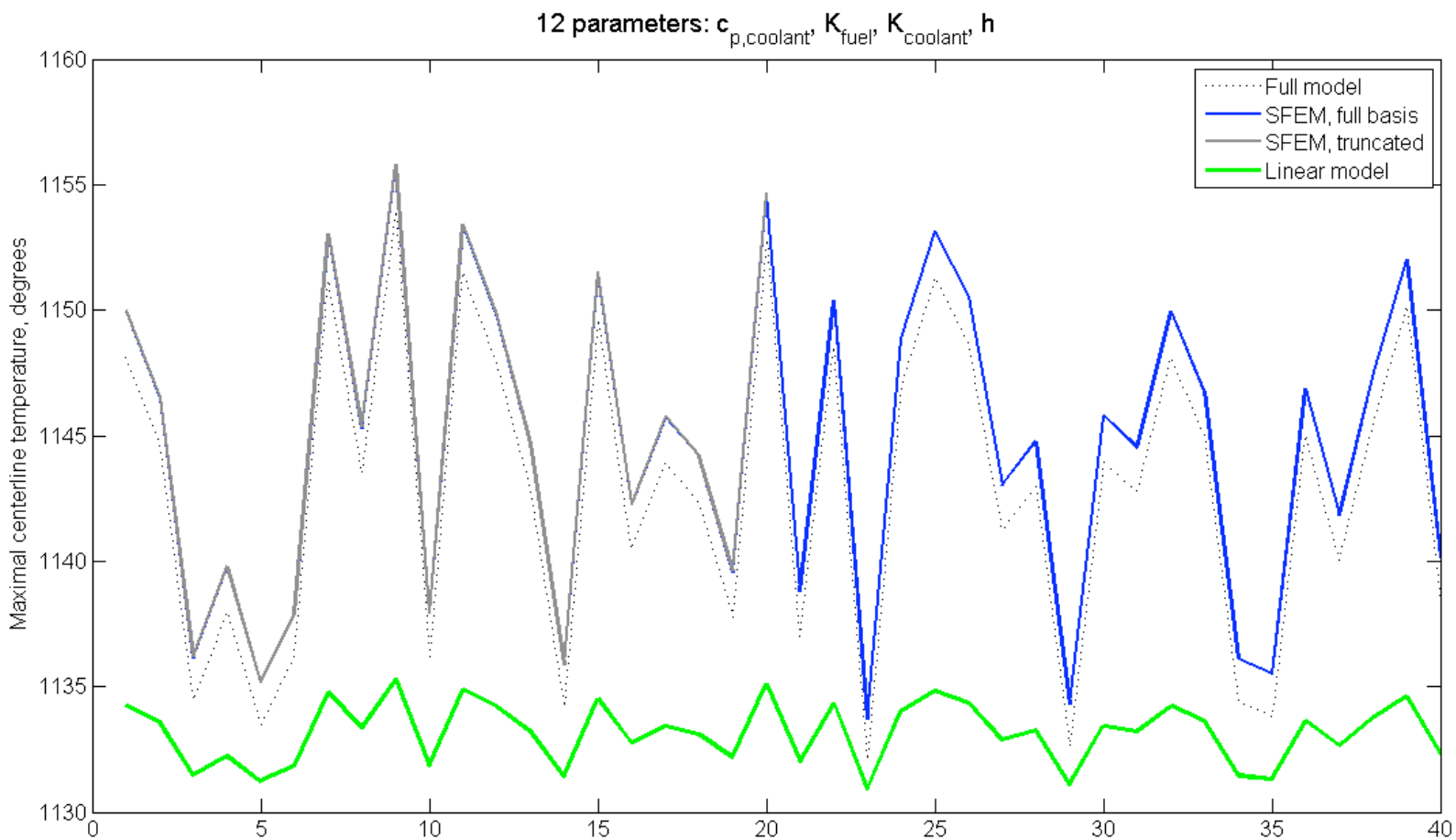
- Size of the finite volume model: 7 pins, 20 horizontal layers.
- The output function is a measure of temperature in the outflow layer:

$$J(T) = \frac{1}{const} \cdot \|T_{outflow}\|_p$$

- We will compare with the linear (sensitivity) model.

$$\hat{J} = J_0 + \sum_i \frac{\partial J}{\partial a_i} a_i$$

Typical performance of the polynomial model: max centerline temperature



Performance of PRD approach, max centerline temperature

	Sampling	Linear Model	SFEM Full	SFEM Truncated
Sample size	40	1	72	12
Range	2237.82 - 2460.54	2227.43 - 2450.09	2237.82 - 2460.55	2237.51 - 2459.63
Model σ^2	3487.08	3495.38	3486.87	3476.01
Model σ	59.05	59.12	59.05	58.96
Error σ^2	*	8.94	0.00015	0.084
Error σ	*	2.99	0.01	0.29

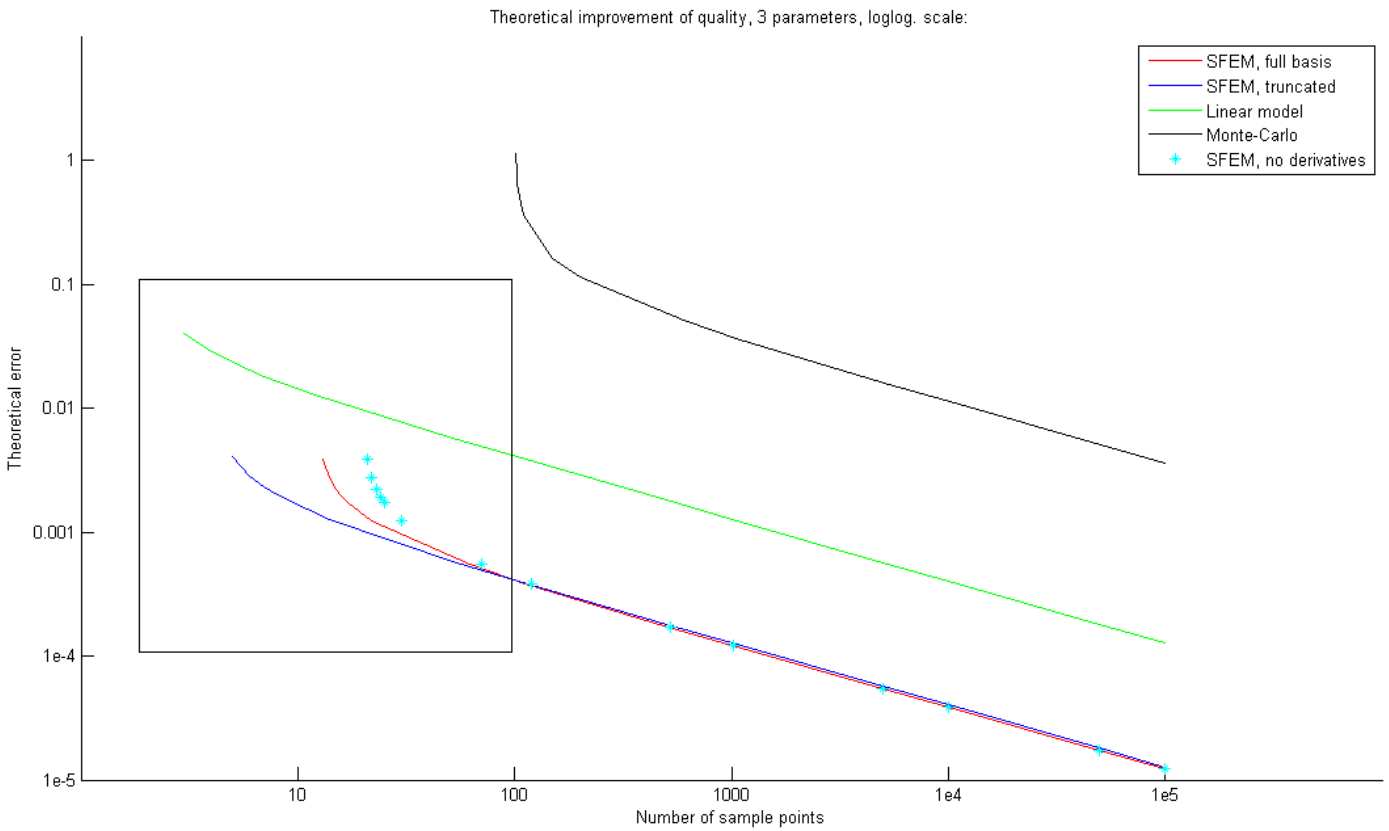
"Preconditioning" with SFEM for assessment,

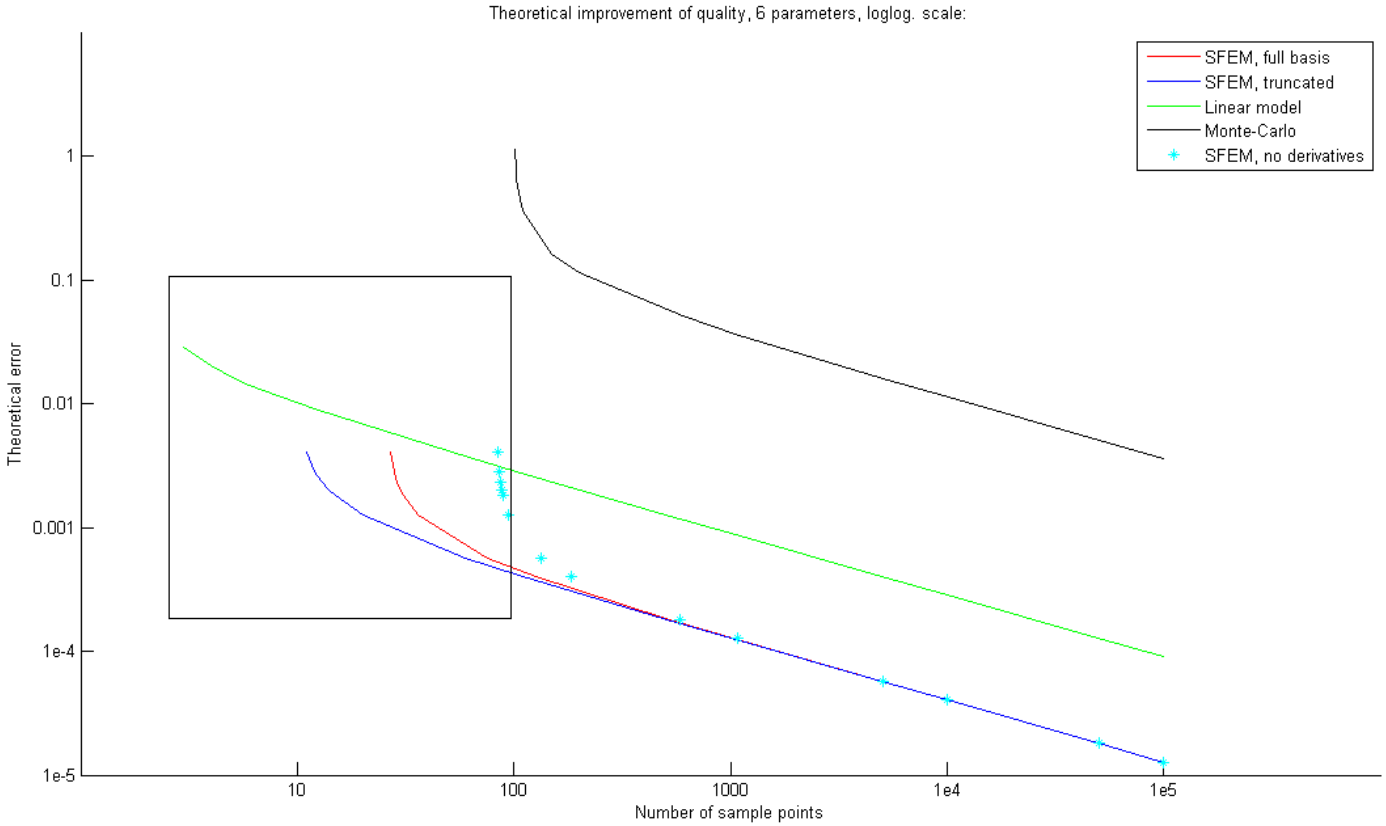
- We can compute our expectation using a control variate technique.

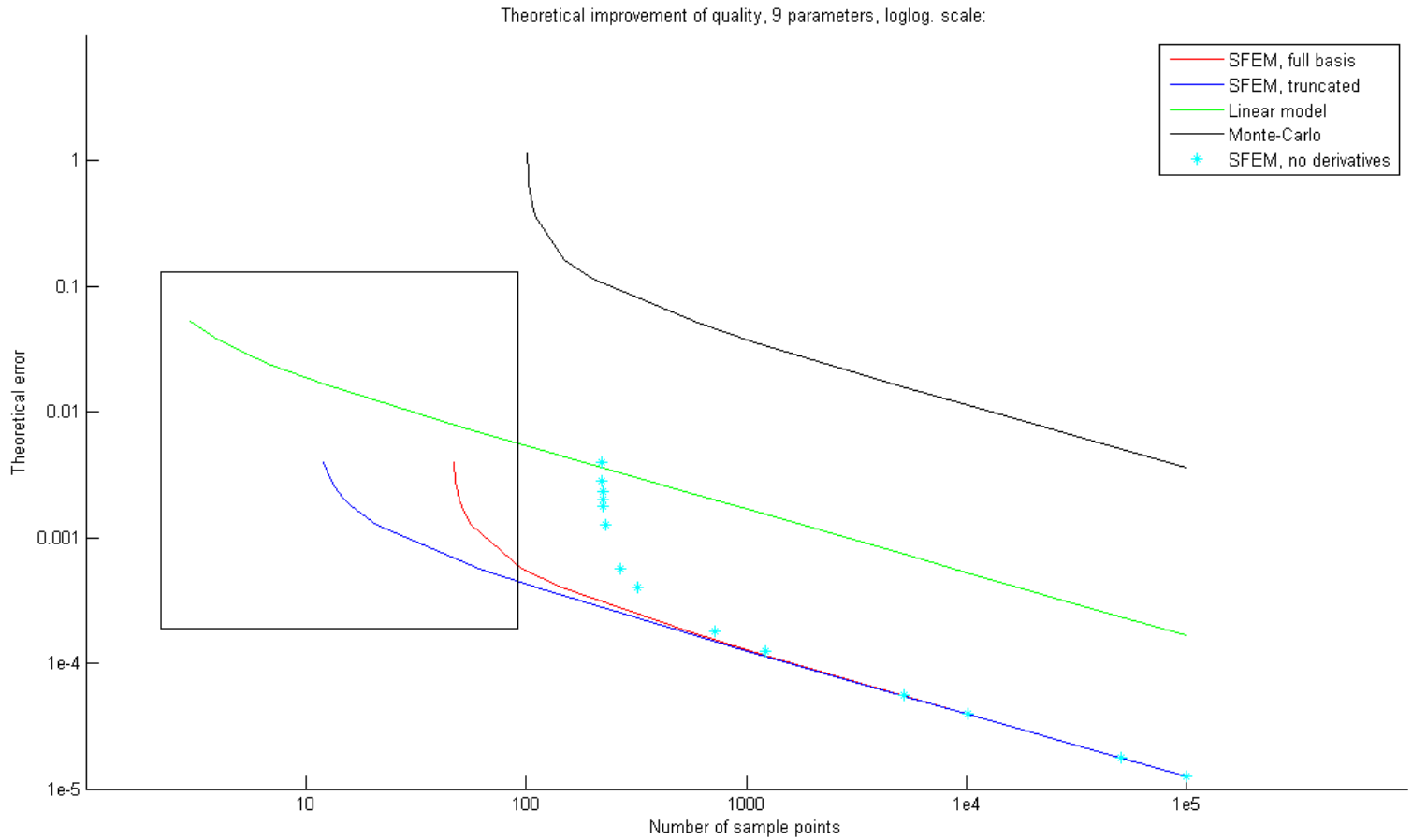
$$E_\omega\left[J\left(x(p)\right)\right] = E_\omega\left[J\left(x(p)\right) - \hat{J}\left(x(p)\right) + E_\omega\left[\hat{J}\left(x(p)\right)\right]\right]$$

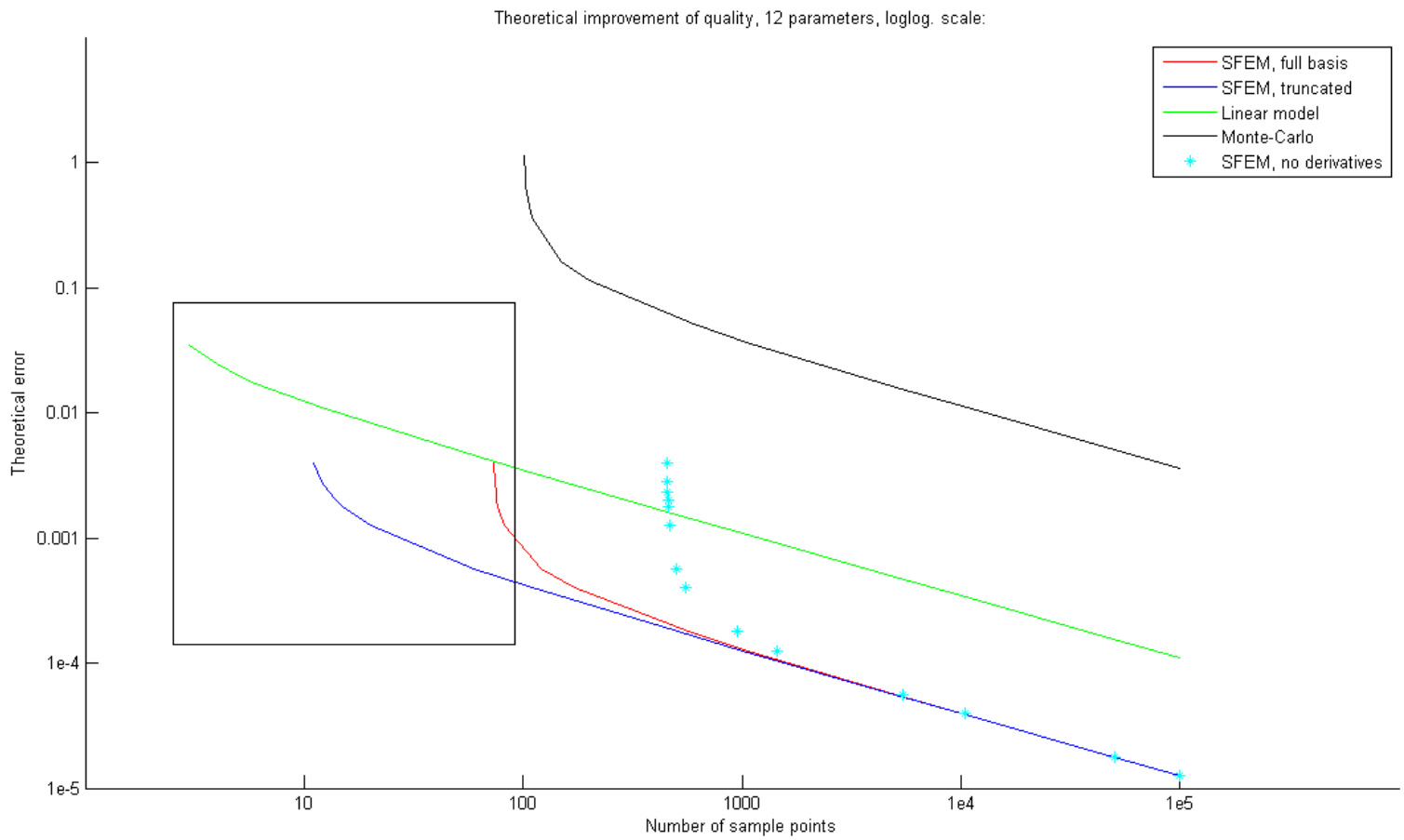
$$Var\left[J\left(x(p)\right) - \hat{J}\left(x(p)\right)\right] \ll Var\left[J\left(x(p)\right)\right]$$

- If the approximation is good, then we need far fewer samples to compute our estimate
- This will answer the question: give me the mean with 90% confidence with very few samples.

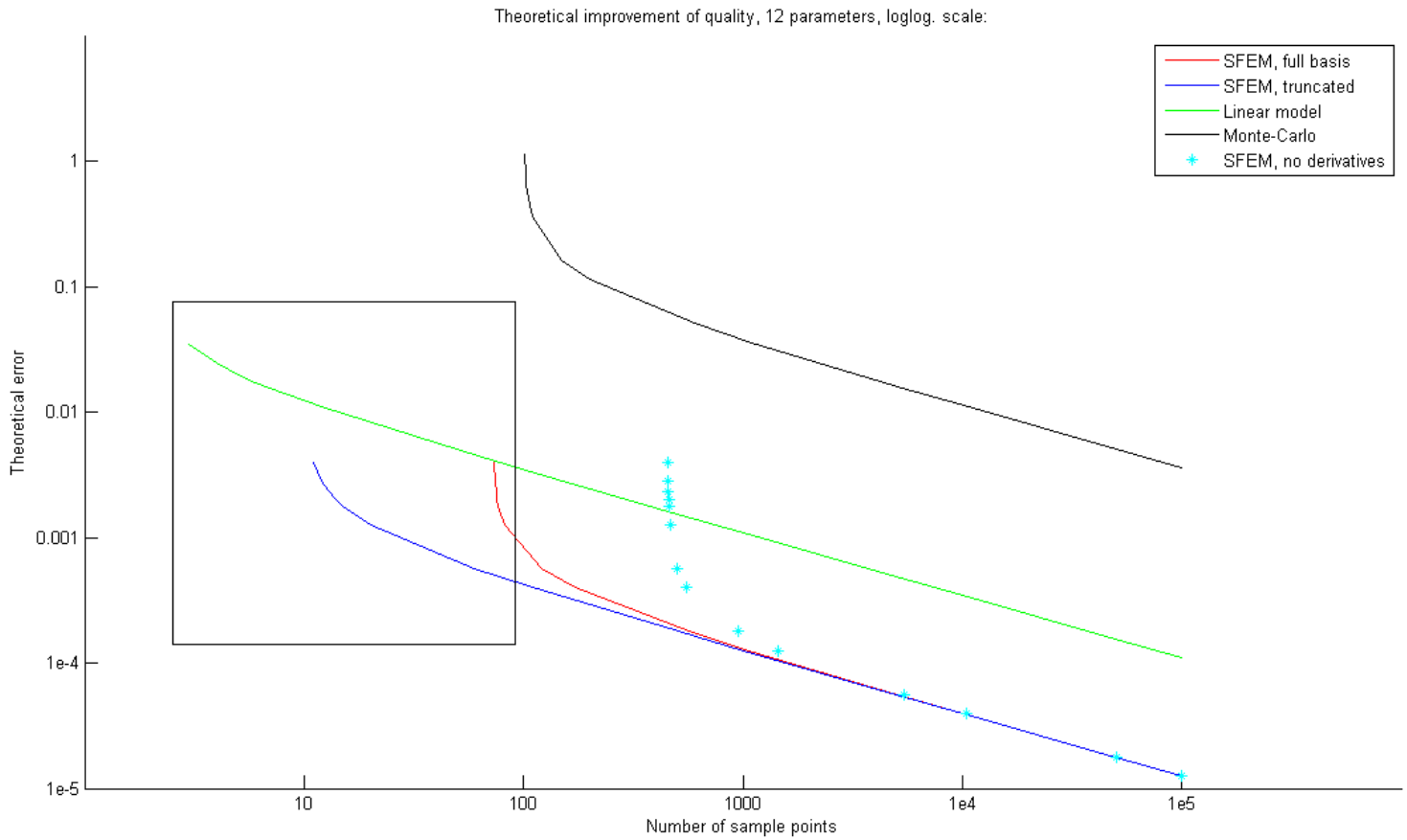

Theoretical improvement of quality, 3 parameters, loglog. scale:
SFEM, full basis
SFEM, truncated
Linear model
Monte-Carlo
SFEM, no derivatives
Theoretical error
1
0.1
0.01
0.001
1e-4
1e-5
10
100
1000
1e4
1e5
Number of sample points


Theoretical improvement of quality, 6 parameters, loglog. scale:
SFEM, full basis
SFEM, truncated
Linear model
Monte-Carlo
SFEM, no derivatives
Theoretical error
1
0.1
0.01
0.001
1e-4
1e-5
10
100
1000
1e4
1e5
Number of sample points


Theoretical improvement of quality, 9 parameters, loglog. scale:
SFEM, full basis
SFEM, truncated
Linear model
Monte-Carlo
SFEM, no derivatives
Theoretical error
1
0.1
0.01
0.001
1e-4
1e-5
10
100
1000
1e4
1e5
Number of sample points


Theoretical improvement of quality, 12 parameters, loglog. scale:
SFEM, full basis
SFEM, truncated
Linear model
Monte-Carlo
SFEM, no derivatives
Theoretical error
1
0.1
0.01
0.001
1e-4
1e-5
10
100
1000
1e4
1e5
Number of sample points

Control variate error versus effort.



How well does PRD capture global nonlinear effects?

- A global measure of importance using marginal distribution:

$$I(\alpha_i) = \frac{\mathrm{var}[E(J_i)]}{E(J)}$$

$$E[J_i] = \frac{1}{|\Omega(\alpha_i)|} \int_{\Omega(\alpha_i)} J(S)\, d\alpha_1 d\alpha_2 \ldots d\alpha_{i-1} d\alpha_{i+1} \ldots d\alpha_n$$

Global sensitivity:

- 12 “heat transport” uncertainty parameters, 24 “neutronics” uncertainty parameters

- Complete derivative information can be obtained at approximately 200% additional computational cost, faster with adjoint differentiation.
- There is a significant improvement in bias over the linear model, at a fraction of computational cost of random sampling.
- Sampling of the polynomial approximation shows that it preserves the dominant sensitivites of the full model. Note: sensitivities on 9000 runs of the full model runs were reproduced by 15000 runs of BRD.

	Model	*Approximaton*
K_{fuel}	0.3876	0.4090
	0.0005	0.0017
	0.6097	0.5893

- Some of the uncertainty parameters (“K-coolant”) are of relatively much larger importance for the output of the model (depends on the output).
- Note that if you change the objective function, the conclusion of importance does change.

Global Sensitivity Results "Bang for the buck"

	$Model:$	$SFEM:$
$c_{p,coolant}$	$0.2187 \cdot 10^{-3}$	$0.0001 \cdot 10^{-3}$
	$0.1239 \cdot 10^{-5}$	$0.0017 \cdot 10^{-5}$
	$0.1203 \cdot 10^{-3}$	$0.0001 \cdot 10^{-3}$
K_{fuel}	0.3876	0.4090
	0.0005	0.0017
	0.6097	0.5893
$K_{coolant}$	$0.4963 \cdot 10^{-4}$	$0.1232 \cdot 10^{-4}$
	$0.8723 \cdot 10^{-6}$	$0.0207 \cdot 10^{-6}$
	$0.1416 \cdot 10^{-4}$	$0.1349 \cdot 10^{-4}$
h	$0.3635 \cdot 10^{-4}$	$0.5180 \cdot 10^{-9}$
	$0.2890 \cdot 10^{-5}$	$0.2593 \cdot 10^{-12}$
	$0.2941 \cdot 10^{-6}$	$0.6660 \cdot 10^{-9}$
$Neutronics:$	$0.2892 \cdot 10^{-5} : 0.2651 \cdot 10^{-3}$	$0.0001 \cdot 10^{-5} : 0.0001 \cdot 10^{-3}$

- Correct large values and orders of magnitude.
- To get comparable accuracy 10,000 samples for MC versus 40 samples for derivative SFEM !!

Conclusion

- Polynomial regression with derivatives – a novel way to efficiently incorporate computer simulation information in an uncertainty model.
- High accuracy even for nonlinear phenomena, *without* needing higher order sensitivity
- Can be used to accurately predict relative importance *at a fraction of the cost of sampling the full model*– essential for accuracy/effort compromise.
- The uncertainty information in the thermorhydraulics parameters may not be available to as high accuracy as one might think, this research show initial steps on how to reduce its dimension without sacrificing accuracy.

Future work- bigger questions

- Derivatives:
  - How do I obtain derivatives, particularly if some legacy code is involved? Automatic differentiation? – Apply to simplified version of the SAS codes and/or depletion. It seems reasonable to expect that new codes will provide some form of adjoint sensitivity, as SAS/UNIC/ Nek get updated.
  - Nevertheless, it is likely that not all modules will provide derivative information (Nek?). Can one use derivative information with noise from simplified models? (e.g. use function from Nek, sensitivity from RANS). Note that since we use a regression approach, we can tolerate some error in derivative (also physical/mathematical).
- Machine learning connection
  - Include tools of Information Science and Optimization to achieve optimal multifidelity (1D, diffusion, transport) simulation management for a given set of endgoals.
  - But note that it looks we also may open a new chapter in ML.
- Current uncertainty models for advanced reactor calculations, not suitable for high fidelity simulation– a raw data + transparent analysis approach.

PRD: Polynomial regression with derivatives

- How do I choose better basis functions generators (resulting possibly in fewer samples)? Idea: use orthogonality with respect to the Sobolev product.
- How do we choose good sample points at which to evaluate function/ derivative information – Optimal Design with Derivative Information in arbitrary domains. Idea: minimize a discrepancy function.
- How do I optimally prune the basis functions. Adaptive basis selection using machine learning techniques.
- Time-dependendent bases (Wavelets?).
- Use it to represent model reduction with uncertainty.

A case for Automatic Differentiation

- Derivatives can be obtained with minimal code modifications
  - Even from legacy Fortran codes like MATWS.
- Faster to implement than hand-coded derivatives
  - More accurate and cheaper than finite differences
  - Derivatives computed with AD are fully accurate (up to roundoff)
    - *for smooth functions and implementations*

ADIFOR

- ADIFOR 2.0 (Argonne and Rice University) can handle all standard FORTRAN 77 constructs (the usual concern of NE colleagues)
  - COMMON blocks
    - *ADIFOR automatically creates common blocks for the derivatives*
  - EQUIVALENCE statements
    - *if several model variables share the same memory, so will their derivatives*
  - IMPLICIT variables:
    - *derivatives will explicitly be defined to have the same type as the original variables.*
  - GO TO constructs:
    - *Forward mode differentiation preserves the same control flow*
    - *adjoining requires forward model code modifications*
  - ENTRY statements
  - Labeled CONTINUE or BREAK statements

Differentiating MATWS

- We differentiated MATWS using ADIFOR 2.0
  - 10K lines of legacy F77 code
  - part of the Argonne SAS suite
- Just a few modifications needed before differentiating:
  - removed or re-coded CPU or OS dependent code sections
    - *variable address computations, system calls, etc.*
    - *do not play a role when computing derivatives of output quantities*
  - subroutines with variable number of arguments are not supported
    - *define one subroutine for each call as needed*
  - renamed common blocks with inconsistent sizes between subroutines
  - enforced correct argument types in subroutine calls by explicit casts
- Currently validating the tangent derivatives against finite differences

Automatic Differentiation

- Possible issues:
  - Differentiation of adaptive time stepping routines
    - *MATWS uses an adaptive method for ODE integration*
    - *The AD engine can generate spurious time/time step derivatives*
      - There are known fixes in the AD literature
    - *Mathematically non-differentiable implementations*
      - Fix by re-coding the function/subroutine before running AD
    - *ADIFOR helps by raising runtime exceptions*
      - Enabling exception reporting is always a good idea
    - *Cannot get around mathematical non-differentiability*
      - But neither can hand-coded derivatives or finite differences
      - We can still get perfectly valid derivatives near a discontinuity
      - Maybe use this for a smoothing approach?

Example of AD problems

```
if (x .lt. 0) then
    y = -x
else
    y = x
endif
```

→ 1

```
if (x .le. 0) then
    y = -x
else
    y = x
endif
```

→ -1

```
if (x .eq. 1) then
    y = 1
else
    y = x*x
endif
```

→ 0

Disclaimers

- Argonne National Laboratory is operated by The University of Chicago for the U.S. Department of Energy under contract DE-AC02-06CH11357. As such the following rules apply:

- DISCLAIMER OF LIABILITY: Documents available from this server were prepared as accounts of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor the University of Chicago, nor any of their employees or officers, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights.

- DISCLAIMER OF ENDORSEMENT: Reference to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of document authors do not necessarily state or reflect those of the U.S. Government or any agency thereof.

- COPYRIGHT STATUS: Documents authored by Argonne National Laboratory employees are the result of work under U.S. Government contract DE-AC02-06CH11357 and are therefore subject to the following license: The Government is granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable worldwide license in these documents to reproduce, prepare derivative works, and perform publicly and display publicly by or on behalf of the Government.