Linear Algebra Computations for Parameterized Partial Differential Equations

Howard C. Elman
Department of Computer Science
University of Maryland at College Park

SIAM Applied Linear Algebra
October 2015

Collaborators:

Darran Furnival	UMD / Université de Bretagne
Catherine Powell	University of Manchester
Chris Miller	UMD / Mitre Corporation
Eric Phipps	Sandia National Laboratories
Ray Tuminaro	Sandia National Laboratories
Qifeng Liao	Shanghai Tech University
Virginia Forstall	UMD / Leidos

Work supported by: NSF, DOE

Parameter-Dependent Partial Differential Equations

Examples:

- Diffusion equation: $-\nabla \cdot (a(\mathbf{x},\xi)\nabla u) = f$
- Navier-Stokes equations: $-\nabla \cdot (a(\mathbf{x},\xi)\nabla \vec{u}) + (\vec{u}\cdot\nabla)\vec{u} + \nabla p = \vec{f}$
  $\nabla \cdot \vec{u} = 0$
- Posed on $\mathcal{D} \subset \mathbb{R}^d$ with suitable boundary conditions
- Sources: models of diffusion in media with uncertain permeabilities
  multiphase flows

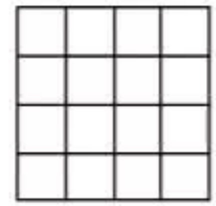
Want solution $\mathbf{u} = \mathbf{u}(\cdot,\xi)$ for many values of ξ. Why?

- Want to perform simulation for multiple design parameters
- Properties of a are not fully understood. Treat them as random
  $a = a(\mathbf{x},\xi)$ is a *random field*: for each fixed $x \in \mathcal{D}$, $a(\mathbf{x},\xi)$ is a random variable depending on m random parameters $\xi_1,\ldots,\xi_m$
- In this study: $a(\mathbf{x},\xi) = a_0(\mathbf{x}) + \sum_{r=1}^{m} a_r(\mathbf{x})\,\xi_r$

$a(\mathbf{x},\xi) = a_0(\mathbf{x}) + \sum_{r=1}^{m} a_r(\mathbf{x})\,\xi_r,$ mean $\bar{a}(\mathbf{x}) \equiv E(a(\mathbf{x},\cdot))$

Possible sources:

Karhunen-Loève expansion or Piecewise constant coefficients on $\mathcal{D}$



One approach for solution: Monte Carlo simulation

- Sample ξ
- Solve PDE $\mathcal{L}_\xi u = f$. (Sample the solution $u(\cdot,\xi)$)
- Repeat

Obtain statistical properties by averaging or counting

Issues: Convergence is slow, costs of sampling (of $u(\cdot,\xi)$) are high

Alternative approach: Use surrogate solutions

Goal: Generate solutions $u(\cdot,\xi)$ for many ξ

Alternative approach:

- Generate *surrogate solutions* $u^s(\cdot,\xi) \approx u(\cdot,\xi)$ that are
  - not too expensive to find, and
  - inexpensive to evaluate
- Use surrogates to perform simulation

Strategies:

- Stochastic Galerkin method
- Stochastic collocation method
- Reduced-order models
- Combinations of some of these

Many interesting linear algebra issues

The Stochastic Galerkin Method

Standard weak diffusion problem: find $u \in H^1_E(\mathcal{D})$ s.t.

$$a(u,v) = \int_{\mathcal{D}} a\nabla u \cdot \nabla v dx = \int_{\mathcal{D}} f\, v\, dx \quad \forall\, v \in H^1_0(\mathcal{D})$$

Extended (*stochastic*) weak formulation: find $u \in H^1_E(\mathcal{D}) \otimes L_2(\Omega)$ s.t.

$$\underbrace{\int_{\Omega}\int_{\mathcal{D}} a\nabla u \cdot \nabla v\, dx\, dP(\Omega)}_{\int_{\Gamma}\int_{\mathcal{D}} a(\mathbf{x},\xi)\,\nabla u\cdot\nabla v\, d\mathbf{x}\,\rho(\xi)\,d\xi} = \underbrace{\int_{\Omega}\int_{\mathcal{D}} f\, v\, dx\, dP(\Omega)}_{\int_{\Gamma}\int_{\mathcal{D}} f\, v\, dx\,\rho(\xi)\,d\xi} \quad \forall\, v \in H^1_0(\mathcal{D}) \otimes L_2(\Omega) \quad (\Gamma = \xi(\Omega))$$

- **Discretization** in physical space: $\mathcal{S}^{(h)}_E \subset H^1_E(\mathcal{D})$, basis $\{\phi_j\}_{j=1}^N$
  Example: piecewise linear "hat functions"
- **Discretization** in space of random variables: $\mathcal{T}^{(p)} \subset L^2(\Gamma)$, basis $\{\psi_\ell\}_{\ell=1}^M$
  Example: *m*-variate polynomials in ξ of total degree p

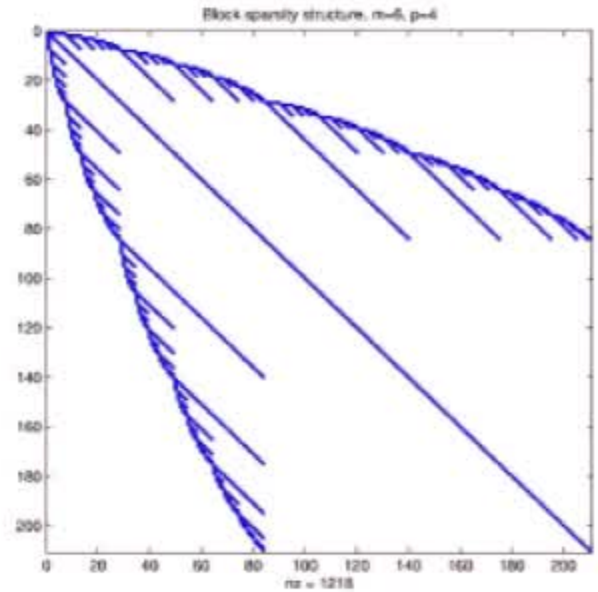
Discrete surrogate solution:

$u_{hp}(\mathbf{x},\xi) = \sum_{j=1}^{N} \sum_{\ell=1}^{M} u_{j\ell}\phi_j(\mathbf{x})\psi_\ell(\xi)$

Requires solution of large coupled system

Matrix (right): $G_0 \otimes A_0 + \sum_{r=1}^{m} G_r \otimes A_r$

"Stochastic dimension": $M = \binom{m+p}{p}$



$M = 210$

Ghanem, Spanos, Babuška, Deb, Oden, Matthies, Keese, Karniadakis, Xue, Schwab, Todor

Multigrid for Galerkin systems

I. Apply multigrid across spatial component (E. & Furnival)

Solving $A\mathbf{u} = \mathbf{f}$, $\quad A = G_0 \otimes A_0^{(h)} + \sum_{r=1}^{m} G_r \otimes A_r^{(h)}$

$[A_r]_{jk} = \int_{\mathcal{D}} a_r(x) \nabla\phi_k(x) \cdot \nabla\phi_j(x) dx$, $\ [G_r]_{lq} = \int_{\Gamma(\Omega)} \xi_r \psi_q(\xi) \psi_l(\xi) \rho(\xi) d\xi$

Fine grid operators: $A^{(h)}$, $A_r^{(h)}$ spatial discretization parameter h
Course grid operators: $A^{(2h)}$, $A_r^{(2h)}$ spatial discretization parameter $2h$

One multigrid (two-grid) step:

for $j = 1 : k$
  $u^{(h)} \leftarrow u^{(h)} + Q^{-1}(f^{(h)} - A^{(h)} u^{(h)})$ k smoothing steps
end
$r^{(2h)} = \mathcal{R}(f^{(h)} - A^{(h)} u^{(h)})$ Restriction
Solve $A^{(2h)} c^{(2h)} = r^{(2h)}$ Coarse grid correction $\mathcal{R} = I \otimes R$
$u^{(h)} \leftarrow u^{(h)} + \mathcal{P} c^{(2h)}$ Prolongation $\mathcal{P} = I \otimes P$

Sketch of convergence analysis: Use "standard" approach

$$e^{(i+1)} = [(A^{(h)})^{-1} - \mathcal{P}(A^{(2h)})^{-1}\mathcal{R}]\,[A^{(h)}(I - Q^{-1}A^{(h)})^k]\,e^{(i)}$$

Establish for all y

Approximation property $\left\|[(A^{(h)})^{-1} - \mathcal{P}(A^{(2h)})^{-1}\mathcal{R}]y\right\|_{A^{(h)}} \leq \|y\|_2$

Smoothing property $\left\|A^{(h)}(I - Q^{-1}A^{(h)})^k y\right\|_2 \leq \|y\|_{A^{(h)}}$

For approximation property: Introduce *semi-discrete space* $H_0^1(\mathcal{D}) \otimes \mathcal{T}^{(p)}$

$\mathcal{T}^{(p)}$ = discrete stochastic space

Weak formulation: $a(u^{(p)}, v^{(p)}) = (f, v^{(p)})$ for all $v^{(p)} \in H_0^1(\mathcal{D}) \otimes \mathcal{T}^{(p)}$

Then:

$$\begin{aligned}\left\|[(A^{(h)})^{-1} - \mathcal{P}(A^{(2h)})^{-1}\mathcal{R}]y\right\|_{A^{(h)}} &= \|u^{(hp)} - u^{(2h,p)}\|_a \\ &\leq \|u^{(hp)} - u^{(p)}\|_a + \|u^{(p)} - u^{(2h,p)}\|_a \\ &\leq c\|y\|_{A^{(h)}}\end{aligned}$$

Last step: from standard arguments based on approximability, regularity for *every realization* in the semi-discrete space

Mean-Based Multigrid

II. Apply multigrid to mean as preconditioner

Solving $A\mathbf{u} = \mathbf{f}$

Preconditioner for use with CG (Kruger, Pellisetti, Ghanem):

Mean $Q = G_0 \otimes A_0$

$A_0 \sim \int_{\mathcal{D}} \bar{a}(x,\cdot)\nabla\phi_k(x)\cdot\nabla\phi_j(x)dx, \quad G_0 = I$

Further refinement (Le Maître, et al.)

Use multigrid to approximate action of Q^{-1}:

$$Q_{MG}^{-1} \equiv I \otimes A_{0,MG}^{-1}$$

Convergence analysis (E. & Powell):

Coefficient:	$a(\mathbf{x},\xi) = a_0 + \sigma \sum_{r=1}^{m} \sqrt{\lambda_r}\, a_r(\mathbf{x})\xi_r$
Coefficient matrix:	$A = G_0 \otimes A_0 + \sum_{r=1}^{m} G_r \otimes A_r$
Mean-based preconditioner:	$Q = G_0 \otimes A_0$
Multigrid preconditioner:	$Q_{MG} = G_0 \otimes A_{0,MG}$

Theorem: For $a_0 = \mu$ constant,

$$1 - \tau \le \frac{(w,Aw)}{(w,Qw)} \le 1 + \tau$$

where
$$\tau = (\sigma/\mu)\, c(p) \sum_{r=1}^{m} \sqrt{\lambda_r}\, \|a_r\|_\infty .$$

If in addition the MG approximation satisfies $\beta_1 \le \frac{(w,Qw)}{(w,Q_{MG}w)} \le \beta_2$, then

$$\frac{(w,Aw)}{(w,Q_{MG}w)} = \frac{(w,Aw)}{(w,Qw)}\, \frac{(w,Qw)}{(w,Q_{MG}w)} \le \left(\frac{1+\tau}{1-\tau}\right)\left(\frac{\beta_2}{\beta_1}\right)$$

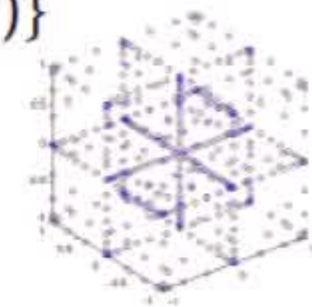
The Stochastic Collocation Method

Monte-Carlo (sampling) method: find $u \in H^1_E(\mathcal{D})$ s.t.

$$\int_{\mathcal{D}} a(\mathbf{x}, \xi^{(k)}) \nabla u \cdot \nabla v dx \quad \text{for all } v \in H^1_{E_0}(\mathcal{D})$$

for a collection of samples $\{\xi^{(k)}\} \in L^2(\Gamma)$

Collocation (Xiu, Hesthaven, Babuška, Nobile, Tempone, Webster)
Choose $\{\xi^{(k)}\}$ in a special way (sparse grids), then construct
construct discrete solution $u_{hp}(\mathbf{x}, \xi)$ to interpolate $\{u_h(\mathbf{x}, \xi^{(k)})\}$



Surrogate (collocation) solution:

$$u_{hp}(\mathbf{x}, \xi) := \sum_{\xi^{(k)} \in \Theta_p} u_c(\mathbf{x}, \xi^{(k)}) L_{\xi^{(k)}}(\xi)$$

Features:
- Decouples algebraic system (like MC)
- Applies in a straightforward way to nonlinear random terms
- Coefficients $\{u_c(\mathbf{x}, \xi^{(k)})\}$ obtained from *large-scale* PDE solve
- *Expensive* when number of points $|\Theta_p|$ is large

Properties of These Methods

For both Galerkin and collocation

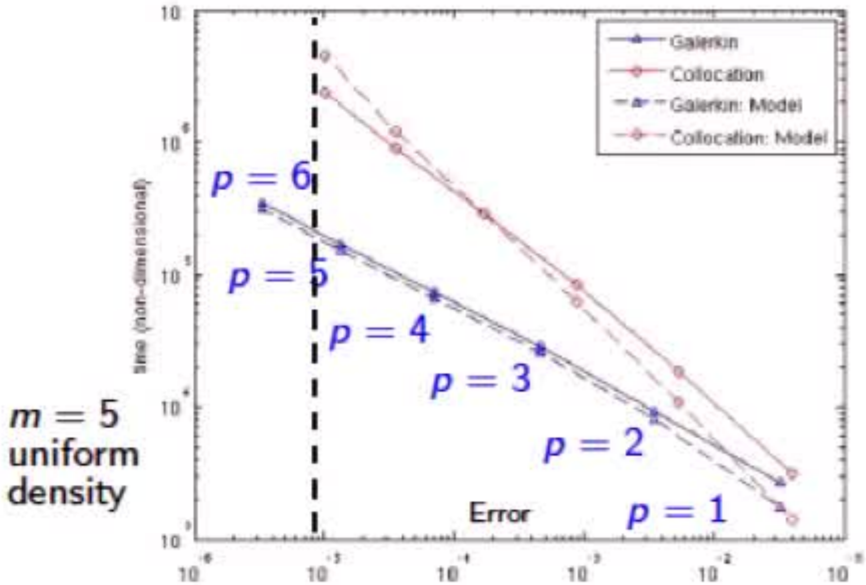
- Each computes a discrete function u_{hp}
- Moments of u estimated using moments of u_{hp} (cheap)
- Convergence: $\|E(u) - E(u_{hp})\|_{H_1(\mathcal{D})} \le c_1 h + c_2 r^p$, $r < 1$
  Exponential in polynomial degree
- Contrast with Monte Carlo:
  Perform N_{MC} (discrete) PDE solves to obtain samples $\{u_h^{(s)}\}_{s=1}^{N_{MC}}$
  Moments from averaging, e.g., $\hat{E}(u_h) = \frac{1}{N_{MC}} \sum_{s=1}^{N_{MC}} u_h^{(s)}$
  Error $\sim 1/\sqrt{N_{MC}}$

One other thing: "p" has different meaning for Galerkin and collocation

- **Disadvantage of collocation:** For comparable accuracy
  $\#$ stochastic dof (collocation) $\approx 2^p$ ($\#$ stochastic dof (Galerkin))

Representative Comparison for Diffusion Equation

Representative comparative performance (E., Miller, Phipps, Tuminaro)



Using mean-based preconditioner for Galerkin system
Kruger, Pellisetti, Ghanem
Le Maître, et al., E. & Powell

Question: Can costs of collocation be reduced?

Reduced Basis Methods

Starting point: Parameter-dependent PDE $\mathcal{L}_{\boldsymbol{\xi}} u = f$

In examples given: $\mathcal{L}_{\boldsymbol{\xi}} = -\nabla \cdot (a_0 + \sigma \sum_{r=1}^{m} \sqrt{\lambda_r} a_r(\mathbf{x}) \xi_r) \nabla$

Discretize: Discrete system $\mathcal{L}_{h,\boldsymbol{\xi}}(u_h) = f$

Algebraic system $\mathcal{F}_{\boldsymbol{\xi}}(\mathbf{u}_h) = 0$ $(A_{\boldsymbol{\xi}} \mathbf{u}_h = \mathbf{f})$ of order N

Complication:

Expensive if many realizations (samples of ξ) are required

Idea (Patera, Boyaval, Bris, Lelièvre, Maday, Nguyen, ...):

Solve the problem on a *reduced space*

That is: by some means, choose $\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(n)}$, $n \ll N$

Solve $\mathcal{F}_{\boldsymbol{\xi}^{(i)}}(u_h^{(i)}) = 0$, $u_h^{(i)} = u_h(\cdot, \xi^{(i)})$, $i = 1, \ldots, n$

For other ξ, approximate $u_h(\cdot, \xi)$ by $\tilde{u}_h(\cdot, \xi) \in span\{u_h^{(1)}, \ldots, u_h^{(n)}\}$

Terminology: $\{u_h^{(1)}, \ldots, u_h^{(n)}\}$ called **snapshots**

Offline Computations

Strategy for generating a basis / choosing snapshots (Patera, et al.):

For $\tilde{u}_h(\cdot,\xi) \approx u_h(\cdot,\xi)$ (equivalently, $\tilde{\mathbf{u}}_\xi \approx \mathbf{u}_\xi$), use an **error indicator** $\eta(\tilde{u}_h) \approx \|e_h\|$, $e_h = u_h - \tilde{u}_h$

Given: a set of candidate parameters $\mathcal{X} = \{\xi\}$,
an initial choice $\xi^{(1)} \in \mathcal{X}$, and $u^{(1)} = u(\cdot,\xi^{(1)})$

Set $Q = \mathbf{u}^{(1)}$

```
while max_{ξ∈X} (η(ũ_h(·,ξ))) > τ
    compute ũ_h(·,ξ), η(ũ_h(·,ξ)), ∀ξ ∈ X    % use current reduced
    let ξ* = argmax_{ξ∈X} (η(ũ_h(·,ξ))       % basis
    if η(ũ_h(·,ξ*)) > τ then
        augment basis with u_h(·,ξ*), update Q with u_{ξ*}
    endif
end
```

Potentially expensive, but viewed as "offline" *preprocessing*
"*Online*" simulation done using reduced basis

Reduced Problem

For linear problems, matrix form:

Coefficient matrix A_{ξ}, nodal coefficients $\mathbf{u}_h$, $\tilde{\mathbf{u}}_h$, $\mathbf{u}^{(1)}, \ldots \mathbf{u}^{(n)}$

$Q =$ orthogonal matrix whose columns span space spanned by $\{\mathbf{u}^{(i)}\}$

Galerkin condition: make residual orthogonal to spanning space

$r = f - A_{\xi}\tilde{\mathbf{u}}_{\xi} = f - A_{\xi} Q\mathbf{y}_{\xi}$ orthogonal to Q

Result is **reduced problem**: Galerkin system of order $n \ll N$:

$$[Q^T A Q]\mathbf{y}_{\xi} = Q^T f, \quad \tilde{\mathbf{u}}_{\xi} = Q\mathbf{y}_{\xi}$$

Goals: Reduced solution should

- be available at significantly lower cost
- capture features of the model

How are costs reduced?

- Matrix A of order N
- Reduced matrix $Q^T A Q$ of order $n \ll N$
- Solving reduced problem is cheap for small n
- Note: making assumption that $\mathcal{L}_{\boldsymbol{\xi}}$ is affinely dependent on ξ

$$\mathcal{L}_{\boldsymbol{\xi}} = \sum_{i=1}^{k} \phi_i(\xi) \mathcal{L}_i$$
$$\Rightarrow A_{\boldsymbol{\xi}} = \sum_{i=1}^{k} \phi_i(\xi) A_i$$
$$\Rightarrow Q^T A_{\boldsymbol{\xi}} Q = \sum_{i=1}^{k} \phi_i(\xi) \underbrace{[Q^T A_i Q]}$$

part of offline computation

True for example seen so far, KL-expansion

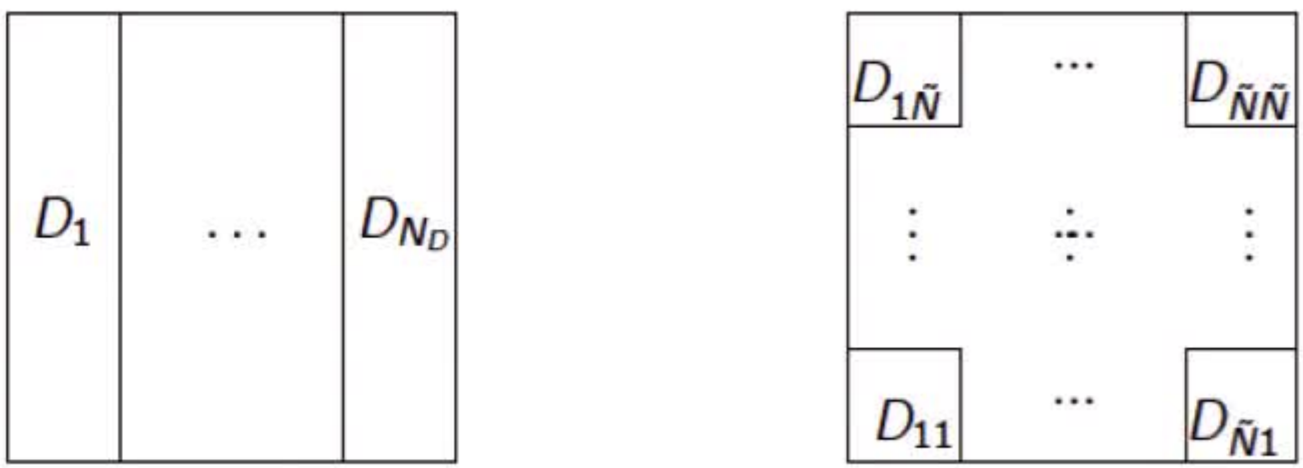
- Consequence: constructing reduced matrix for new ξ is cheap
- Analogue for nonlinear problems is more complex

Reduced Problem: Capturing Features of Model

Consider benchmark problems:

Diffusion equation $-\nabla \cdot (a(\mathbf{x}, \xi)\nabla u) = f$ in $\mathbb{R}^2$

Piecewise constant diffusion coefficient parameterized as a random variable $\xi = [\xi_1, \cdots, \xi_{N_D}]^T$ independently and uniformly distributed in $\Gamma = [0.01, 1]^{N_D}$



(a) Case 1: N_D subdomains (b) Case 2: $N_D = \tilde{N} \times \tilde{N}$ subdomains

Does reduced basis capture features of model?

To assess this: consider

Full snapshot set, set of snapshots for all possible parameter values:

$$\mathcal{S}_\Gamma := \{u_h(\cdot,\xi),\ \xi \in \Gamma\}$$

Finite snapshot set, for finite $\Theta \subset \Gamma$:

$$\mathcal{S}_\Theta := \{u_h(\cdot,\xi),\ \xi \in \Theta\}$$

Question:

How many samples $\{\xi\}$ / $\{u_h(\cdot,\xi)\}$ are needed to accurately represent the features of $\mathcal{S}_\Gamma$?

Experiment: to gain insight into this, estimate "rank" of $\mathcal{S}_\Gamma$

Generate a large set Θ of samples of ξ

Generate the finite snapshot set S_Θ associated with Θ

Construct the matrix S_Θ of coefficient vectors $\mathbf{u}_\xi$ from $\mathcal{S}_\Theta$

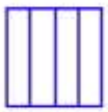
Compute the rank of S_Θ

Results follow. Used 3000 samples

Experiment was repeated ten times with similar results

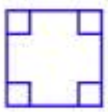
Estimated ranks of $\mathcal{S}_\Gamma$ for two classes of benchmark problems

Case 1



Grid \ N_D	2	3	4	5	6	7	8	9	10
$33^2 = 1089$	3	12	18	30	40	53	55	76	84
$65^2 = 4225$	3	12	18	30	40	48	55	70	87
$129^2 = 16641$	3	12	18	28	39	48	55	72	81

Case 2



Grid \ N_D	4	9	16	25	36	49	64
$33^2 = 1089$	27	121	193	257	321	385	449
$65^2 = 4225$	28	148	290	465	621	769	897
$129^2 = 16641$	28	153	311	497	746	1016	1298

Trends:

- Rank is dramatically smaller than problem dimension N
- Rank is independent of problem dimension ($\sim$ (mesh size)$^{-2}$)
- In most cases, cost of treating reduced problem of given rank is low

1. Introduction
2. Surrogate solution methods
3. Combined approaches and low-rank methods
    - Combined collocation and reduced-basis
    - Low-rank methods
4. Reduced-order methods for nonlinear problems
5. Concluding remarks

Reduced Basis + Sparse Grid Collocation

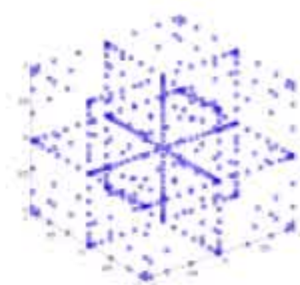
Adapt to sparse grid collocation: Recall collocation solution

$$u_q^{(hp)}(x, \xi^{(k)}) = \sum_{\xi^{(k)} \in \Theta_q} u_c(x, \xi^{(k)}) L_{\xi^{(k)}}(\xi) \tag{1}$$

Main ideas:

1. Use sparse grid collocation points as candidate set $\mathcal{X}$,
2. Use reduced solution as coefficient $u_c(\cdot, \xi^{(k)})$ whenever possible

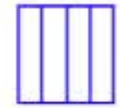
Algorithm



for each sparse grid level p
    for each point $\xi^{(k)}$ at level p
        compute reduced solution $u_R(\cdot, \xi^{(k)})$
        if $\eta(u_R(\cdot, \xi^{(k)})) \leq \tau$, **then**
            use $u_R(\cdot, \xi^{(k)})$ as coefficient $u_c(\cdot, \xi^{(k)})$ in (1)
        else
            compute snapshot $u_h(\cdot, \xi^{(k)})$, use it as $u_c(\cdot, \xi^{(k)})$ in (1)
            augment reduced basis with $u_h(\cdot, \xi^{(k)})$, update Q with $\mathbf{u}_{\xi^{(k)}}$
        endif
    end
end

Number of full system solves

Case 1



Case 1, 5×1 subdomains, 65×65 grid, rank=30

p	1	2	3	4	5	6	7	8	11
tol \ $\lvert\Theta_q\rvert$	11	61	241	801	2433	7K	19K	52K	870K
10^{-3}	10	9	0	0	0	0	0	0	0
10^{-4}	10	11	1	0	0	0	0	0	0
10^{-5}	10	13	0	0	0	0	0	0	0

Case 1, 9×1 subdomains, 65×65 grid, rank=70, $tol = 10^{-4}$

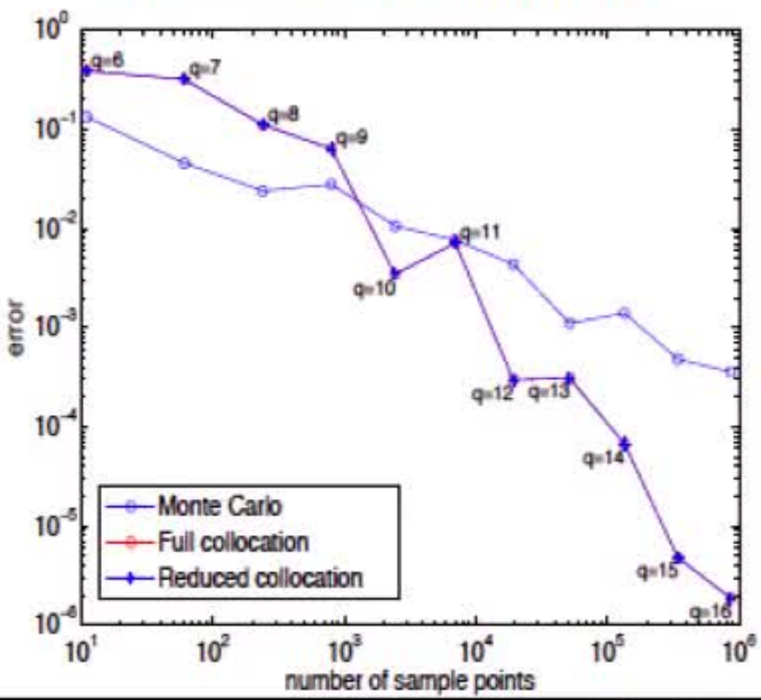
p	1	2	3	4	5	6	7	8
$\lvert\Theta_q\rvert$	19	181	1177	6001	26017	100897	361249	1218049
$N_{full\ solve}$	18	34	2	1	1	0	0	0

To assess accuracy: Examine error (vs. reference solution) in expected values of full or reduced collocation solution:

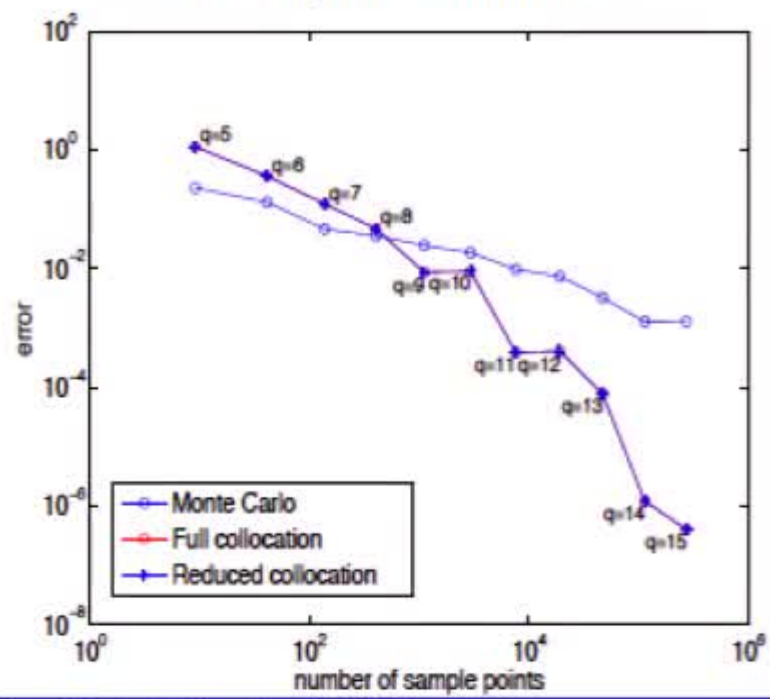
Full collocation $\epsilon_h := \left\| \tilde{\mathbb{E}}\left(u_q^{hsc}\right) - \tilde{\mathbb{E}}\left(u_r^{hsc}\right) \right\|_0 \Big/ \left\| \tilde{\mathbb{E}}\left(u_r^{hsc}\right) \right\|_0$

Reduced collocation $\epsilon_R := \left\| \tilde{\mathbb{E}}\left(u_q^{rsc}\right) - \tilde{\mathbb{E}}\left(u_r^{hsc}\right) \right\|_0 \Big/ \left\| \tilde{\mathbb{E}}\left(u_r^{hsc}\right) \right\|_0$

Case 1: vertical subdomains



Case 2: square subdomains



Interpretation of these results

Collocation points $\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(n_\xi)}$

Solutions $\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \ldots, \mathbf{u}^{(n_\xi)}$, arrange into matrix U

Results show: U is of *low rank* n_r, spanned by reduced basis

$$U = \underbrace{\begin{bmatrix} \mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \cdots, \mathbf{u}^{(n_\xi)} \end{bmatrix}}_{n_\xi} \updownarrow n_x = \underbrace{\begin{bmatrix} Q \end{bmatrix}}_{n_r} \begin{bmatrix} \mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \cdots, \mathbf{y}^{(n_\xi)} \end{bmatrix} \updownarrow n_r$$

Can write collection of collocation equations as $\mathcal{A}(U) = F$

Reduced basis method $\sim$ finding low-rank solution

Idea applies to Galerkin formulation

Galerkin system

$$\left(\sum_{\ell=0}^{m} G_\ell \otimes A_\ell\right) \mathbf{u}_{hp} = \mathbf{f}$$

Equivalently:

$$\sum_{\ell=0}^{m} A_\ell U G_\ell^T = F, \quad \mathbf{u}_{hp} = vec(U),\ \mathbf{f} = vec(F)$$

Kressner & Tobler, Ballani & Grasedyck, Matthies & Zander, Oseledets & Tyrtyshnikov, Schwab & Gittelson, Khoromskij & Schwab, Benner, Onwunta & Stoll, Powell, Silvester & Simoncini

New approach: tensor methods

Recapitulating: For linear/affine models

Three + techniques for construction of surrogates:

1. **Stochastic Galerkin**
   Offline: solve coupled Galerkin system
   Online simulation: evaluate Galerkin solution
2. **Stochastic collocation**
   Offline: solve n_ξ deterministic systems
   Online simulation: evaluate interpolant
3. **Reduced-order model**
   Offline: compute n_r snapshots, use error indicator
   Online simulation: solve reduced-order model

\+ **Combined approaches**
   Offline: use reduced-order philosophy in combination with collocation / Galerkin
   Online simulation: evaluate solution

Reduced-order models for nonlinear systems

Nonlinear discrete system $F_\xi(\mathbf{u}_\xi) = \mathbf{0}$

Preliminary:

Recall linear form $F_\xi(\mathbf{u}_\xi) = A_\xi \mathbf{u}_\xi - \mathbf{f}$, $\quad A_\xi \equiv \sum_{\ell=1}^m A_\ell \phi_\ell(\xi)$

Reduced basis in columns of Q, span $\{\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \ldots, \mathbf{u}^{(n_r)}\}$, $n_r \ll N$

Reduced (surrogate) solution $\tilde{\mathbf{u}}_\xi = Q\mathbf{y}_\xi \approx \mathbf{u}_\xi$ from Galerkin system

$$\underbrace{\left[\sum_{\ell=1}^{m} \underbrace{(Q^T A_\ell Q)}_{\text{Precompute}} \phi_\ell(\xi)\right]}_{\text{Matrix of order } n_r} \mathbf{y}_\xi = Q^T \mathbf{f} \qquad (1)$$

Simulation: New $\xi \longrightarrow$ new system (1)

Construct, solve at cost depending on $n_r \ll N$

Return to nonlinear system $F_\xi(\mathbf{u}_\xi) = \mathbf{0}$

Reduced basis in Q

Reduced operator $Q^T \underbrace{F_\xi(Q\tilde{\mathbf{y}}_\xi)}_{N \text{ (scalar) nonlinear function evaluations}}$

Jacobian $J_{F_\xi}(Q\mathbf{y})$, cost of evaluation also depends on N

Advantages of reduced basis are gone

Example: Navier-Stokes equations

$$-\nabla \cdot (a(\mathbf{x}, \xi) \nabla \vec{u}) + (\vec{u} \cdot \nabla)\vec{u} + \nabla p = \vec{f}, \quad \nabla \cdot \vec{u} = 0$$

Algebraic system has form $F_\xi(\mathbf{u}) = A_\xi \mathbf{u} + C(\mathbf{u}) - \mathbf{b}$

A_ξ = discrete parameter-dependent diffusion operator

$C(\mathbf{u}) = N(\mathbf{u})\mathbf{u}$ = discrete version of $-(u \cdot \nabla)u$

Discrete empirical interpolation

DEIM (Barrault, Maday, Nguyen, & Patera, Chaturantabut & Sorensen)

For $F_{\boldsymbol{\xi}}(\mathbf{u}) = A_{\boldsymbol{\xi}}\mathbf{u} + C(\mathbf{u}) - \mathbf{b}$, reduced model has form

$$F_{\boldsymbol{\xi}}^{(r)}(\hat{\mathbf{u}}) = Q^T A_{\boldsymbol{\xi}} Q\,\mathbf{y} + Q^T C(Q\mathbf{y}) - Q^T\mathbf{b}$$

Strategy for approximating nonlinear term:

- Generate matrix of snapshots $S \equiv \left[C(\mathbf{u}^{(1)}),\ C(\mathbf{u}^{(2)}), \ldots, C(\mathbf{u}^{(M)})\right]$
- Generate low-rank Φ for which $range(S) \approx range(\Phi)$ (via SVD)
  $n_s \equiv rank(\Phi)$, analogous to n_r
- Identify "index choosing" matrix $P = [e_{i_1}, e_{i_2}, \ldots, e_{i_{n_s}}]$
- Replace $C(Q\mathbf{y})$ with approximation $\widehat{C}(Q\mathbf{y}) \equiv \Phi(P^T\Phi)^{-1}P^T C(Q\mathbf{y})$
  $\longrightarrow$ approximation $\widehat{F}_{\boldsymbol{\xi}}(Q\mathbf{y}) = A_{\boldsymbol{\xi}}Q\mathbf{y} + \widehat{C}(Q\mathbf{y}) - \mathbf{b}$
- Galerkin condition: $Q^T\widehat{F}_{\boldsymbol{\xi}}(Q\mathbf{y}) = 0$

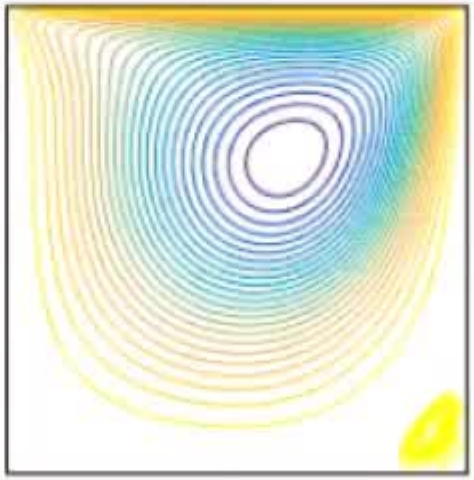
$$Q^T A_{\boldsymbol{\xi}} Q\mathbf{y} + Q^T\Phi(P^T\Phi)^{-1}P^T C(Q\mathbf{y}) - Q^T\mathbf{b} = 0$$

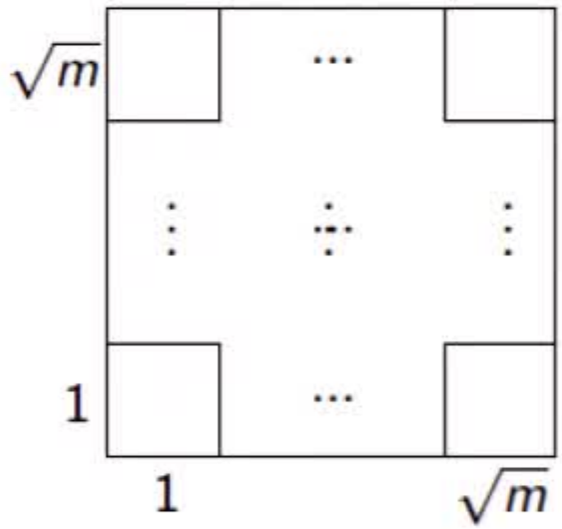
Comments:

- Approximation interpolates desired quantity at indices of P:
  $P^T \Phi (P^T \Phi)^{-1} P^T C(Q\mathbf{y}) = P^T C(Q\mathbf{y})$
- N.B. Need C to be "sparse", OK for grid-based discrete PDE
- Makes evaluation of reduced Jacobian cheap also

Benchmark problem:

Driven cavity flow, piecewise constant viscosity on $\sqrt{m} \times \sqrt{m}$ subdomains





Piecewise constant viscosity

$$\nu(\mathbf{x}, \xi) = \sum_{r=1}^{m} a_r(\mathbf{x})\, \xi_r, \quad a_r = \chi_{D_r}$$

parameterized by random variables $\xi = [\xi_1, \cdots, \xi_m]^T$ independently and uniformly distributed in $\Gamma = [0.01, 1]^m$

Experiment: Solve three versions of the discrete NS equations using Picard iteration:

1. the discrete full system, on 128×128 grid
2. the discrete reduced system w/o special treatment of nonlinear term
3. the discrete reduced system obtained from DEIM

Report: Average CPU times over 10 simulations
Relative residual norms $\eta \equiv \|F_{\boldsymbol{\xi}}\|_2 / \|\mathbf{b}\|_2$
N.B. this error measure is not available at low cost

m	4		16		36		49	
k n_{deim}	237 4		1383 14		3039 23		4083 30	
	time	η	time	η	time	η	time	η
Full	135	1.E-8	147	1.E-8	132	1.E-8	148	1.E-8
Reduced	1.62	1.13E-5	23.8	2.85E-5	98.1	5.14E-5	191	7.16E-5
DEIM	**0.09**	8.27E-5	**1.12**	1.02E-4	**7.11**	1.55E-4	**15.7**	1.56E-4

Preconditioning

During nonlinear iteration, have sequence of systems of order n_r

$$\left(Q^T \begin{bmatrix} A(\xi) & B^T \\ B & 0 \end{bmatrix} Q + \begin{bmatrix} Q_u^T \widehat{C}(\mathbf{u}_j^R) Q_u & 0 \\ 0 & 0 \end{bmatrix} \right) \begin{bmatrix} \delta \mathbf{u}_j \\ \delta \mathbf{p}_j \end{bmatrix} = -r_j^{deim}$$

Would like preconditioners whose construction depends on $n_r \ll N$

Changes the game. Choices:

- Stokes ("beginning") preconditioner: $M = Q^T \begin{bmatrix} A(\xi_0) & B^T \\ B & 0 \end{bmatrix} Q$
- "End" preconditioner:

$$M = Q^T \begin{bmatrix} A(\xi_0) & B^T \\ B & 0 \end{bmatrix} Q + \begin{bmatrix} Q_u^T \widehat{C}(\mathbf{u}_j^R(\xi_0)) Q_u & 0 \\ 0 & 0 \end{bmatrix}$$

Use entails computing and factoring preconditioners in "offline" stage