Gappy POD and GNAT

David Amsallem

Stanford University

February 05, 2014

Outline

- Nonlinear partial differential equations
- An issue with the model reduction of nonlinear equations
- The gappy proper orthogonal decomposition
- The discrete empirical interpolation method (DEIM)
- The Gauss-Newton with approximated tensors method (GNAT)
- Two applications

Nonlinear PDE

- Parametrized partial differential equation (PDE)

$$\mathcal{L}(\mathcal{W}, \mathbf{x}, t; \boldsymbol{\mu}) = 0$$

- Associated boundary conditions

$$\mathcal{B}(\mathcal{W}, \mathbf{x}_{\text{BC}}, t; \boldsymbol{\mu}) = 0$$

- Initial condition

$$\mathcal{W}_0(\mathbf{x}) = \mathcal{W}_{\text{IC}}(\mathbf{x}, \boldsymbol{\mu})$$

  - $\mathcal{W} = \mathcal{W}(\mathbf{x}, t) \in \mathbb{R}^q$: state variable
  - $\mathbf{x} \in \Omega \subset \mathbb{R}^d$, $d \leq 3$: space variable
  - $t \geq 0$: time variable
  - $\boldsymbol{\mu} \in \mathcal{D} \subset \mathbb{R}^p$: parameter vector

Discretization of nonlinear PDE

- The PDE is then discretized in space by one of the following methods
  - Finite Differences approximation
  - Finite Element method
  - Finite Volumes method
  - Discontinuous Galerkin method
  - Spectral method....
- This leads to a system of $N_{\mathbf{w}} = q \times N_{\text{space}}$ ordinary differential equations (ODEs)

$$\boxed{\frac{d\mathbf{w}}{dt} = \mathbf{f}(\mathbf{w}, t; \boldsymbol{\mu})}$$

  in terms of the discretized state variable

$$\mathbf{w} = \mathbf{w}(t; \boldsymbol{\mu}) \in \mathbb{R}^{N_{\mathbf{w}}}$$

  with initial condition $\mathbf{w}(0; \boldsymbol{\mu}) = \mathbf{w}(\boldsymbol{\mu})$
- This is the high-dimensional model (HDM)

Model reduction of nonlinear equations

- High-dimensional model (HDM)

$$\frac{d\mathbf{w}}{dt}(t) = \mathbf{f}(\mathbf{w}(t), t)$$

- Reduced-order modeling assumption using a reduced basis $\mathbf{V}$

$$\mathbf{w}(t) \approx \mathbf{V}\mathbf{q}(t)$$

  - $\mathbf{q}(t)$: reduced (generalized) coordinates
- Inserting in the HDM equation

$$\mathbf{V}\frac{d\mathbf{q}}{dt}(t) \approx \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

- $N_{\mathbf{w}}$ equations in terms of k unknowns $\mathbf{q}$
- Galerkin projection

$$\frac{d\mathbf{q}}{dt}(t) = \mathbf{V}^T\mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

Issue with the model reduction of nonlinear equations

- Galerkin projection

$$\frac{d\mathbf{q}}{dt}(t) = \mathbf{V}^T \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

- k equations in terms of k unknowns
- To evaluate $\mathbf{f}_k(\mathbf{V}\mathbf{q}(t), t)$:
  1. Compute $\mathbf{w}(t) = \mathbf{V}\mathbf{q}(t)$
  2. Evaluate $\mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$
  3. Left-multiply by $\mathbf{V}^T$: $\mathbf{V}^T \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$
- The computational cost associated with these three steps scales linearly with the dimension $N_{\mathbf{w}}$ of the HDM
- Hence no significant speedup can be expected when solving the projection-based ROM

The Gappy POD

- First applied to face recognition (Emerson and Sirovich, "Karhunen-Loeve Procedure for Gappy Data" 1996)
- Procedure
  1. Build a database of m faces (snapshots)
  2. Construct a POD basis $\mathbf{V}$ for the database
  3. For a new face $\mathbf{f}$, record a few pixels $f_1, \cdots, f_n$
  4. Using the POD basis $\mathbf{V}$, approximately reconstruct the new face $\mathbf{f}$

The Gappy POD

- First applied to face recognition (Emerson and Sirovich, ”Karhunen-Loeve Procedure for Gappy Data” 1996)

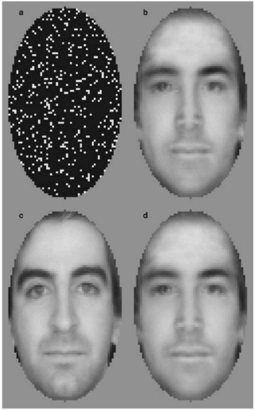


Fig. 1. Reconstruction of a face, not in the original ensemble, from a 10% mask. The reconstructed face, b, was determined with 50 empirical eigenfunctions and only the white pixels shown in a. The original face is shown in c, and a projection (with all the pixels) of the face onto 50 empirical eigenfunctions is shown in d.

The Gappy POD

- Other applications
  - Flow sensing and estimation (Willcox, 2004)
  - Flow reconstruction
  - Nonlinear model reduction

Nonlinear function approximation by gappy POD

- Approximation of the nonlinear function $\mathbf{f}$ in

$$\frac{d\mathbf{q}}{dt} = \mathbf{V}^T \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

- The evaluation of all the entries in the vector $\mathbf{f}(\cdot, t)$ is expensive (scales with $N_{\mathbf{w}}$)
- Only a small subset of these entries will be evaluated (gappy approach)
- The other entries will be reconstructed either by interpolation or a least-squares strategy using a pre-computed specific reduced-order basis $\mathbf{V}_{\mathbf{f}}$
- The solution space is still reduced by any preferred model reduction method (by POD for instance)

Nonlinear function approximation by gappy POD

A complete model reduction method should then provide algorithms for

- Selecting the evaluation indices $\mathcal{I} = \{i_1, \cdots, i_{N_i}\}$
- Selecting a reduced-order bases $\mathbf{V_f}$ for the nonlinear function
- Reconstructing the complete approximated nonlinear function vector $\hat{\mathbf{f}}(\cdot, t)$

Construction of a POD basis for $\mathbf{f}$

- Construction of a POD basis $\mathbf{V_f}$ of dimension $k_\mathbf{f}$
  1. Collection of snapshots for the nonlinear function from a transient simulation
  $$\mathbf{F} = [\mathbf{f}(\mathbf{w}(t_1), t_1), \cdots, \mathbf{f}(\mathbf{w}(t_{m_\mathbf{f}}), t_{m_\mathbf{f}})] \in \mathbb{R}^{N_\mathbf{w} \times m_\mathbf{f}}$$
  2. Singular value decomposition
  $$\mathbf{F} = \mathbf{U_f} \mathbf{\Sigma_f} \mathbf{Z_f}^T$$
  3. Basis truncation ($k_\mathbf{f} \ll m_\mathbf{f}$)
  $$\mathbf{V_f} = [\mathbf{u}_{\mathbf{f},1}, \cdots, \mathbf{u}_{\mathbf{f},k_\mathbf{f}}]$$

Reconstruction of an approximated nonlinear function

- Assume k_i indices have been chosen

$$\mathcal{I} = \{i_1, \cdots, i_{k_i}\}$$

- The choice of indices will be specified later
- Consider the $N_{\mathbf{w}}$-by-k_i matrix

$$\mathbf{P} = \left[\mathbf{e}_{i_1}, \cdots, \mathbf{e}_{i_{k_i}}\right]$$

- At each time t, for a given value of the state $\mathbf{w}(t) = \mathbf{V}\mathbf{q}(t)$, only the entries in the function $\mathbf{f}$ corresponding to those indices will be evaluated

$$\mathbf{P}^T\mathbf{f}(\mathbf{w}(t), t) = \begin{bmatrix} f_{i_1}(\mathbf{w}(t), t) \\ \vdots \\ f_{i_{k_i}}(\mathbf{w}(t), t) \end{bmatrix}$$

- This is cheap if $k_i \ll N_{\mathbf{w}}$
- Usually only a subset of the entries in $\mathbf{w}(t)$ will be required to construct that vector (case of sparse Jacobian)

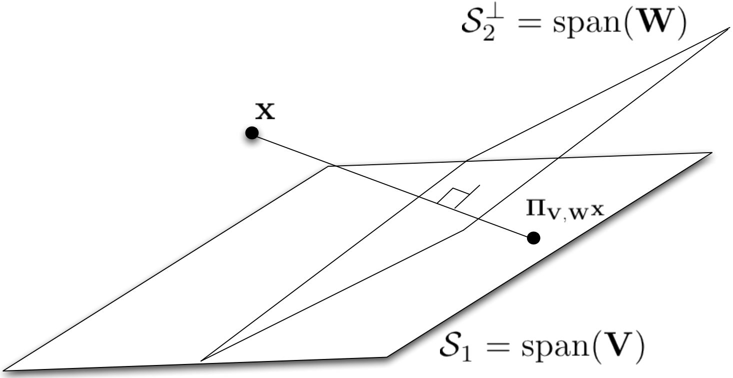
Discrete Empirical Interpolation Method

- Case where $k_i = k_{\mathbf{f}}$: interpolation
  - Idea: $\hat{f}_{i_j}(\mathbf{w}, t) = f_{i_j}(\mathbf{w}, t), \ \forall \mathbf{w} \in \mathbb{R}^{N_\mathbf{w}}, \ \forall j = 1, \cdots, k_i$
  - This means that
$$\mathbf{P}^T \hat{\mathbf{f}}(\mathbf{w}(t), t) = \mathbf{P}^T \mathbf{f}(\mathbf{w}(t), t)$$
  - Remember that $\hat{\mathbf{f}}(\cdot, t)$ belongs to the span of the vectors in $\mathbf{V}_\mathbf{f}$, that is
$$\hat{\mathbf{f}}(\mathbf{V}\mathbf{q}(t), t) = \mathbf{V}_\mathbf{f} \mathbf{f}_r(\mathbf{q}(t), t)$$
  - Then
$$\mathbf{P}^T \mathbf{V}_\mathbf{f} \mathbf{f}_r(\mathbf{q}(t), t) = \mathbf{P}^T \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$
  - Assuming $\mathbf{P}^T \mathbf{V}_\mathbf{f}$ is nonsingular
$$\mathbf{f}_r(\mathbf{q}(t), t) = (\mathbf{P}^T \mathbf{V}_\mathbf{f})^{-1} \mathbf{P}^T \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$
  - In terms of $\hat{\mathbf{f}}(\cdot, t)$:
$$\hat{\mathbf{f}}(\cdot, t) = \mathbf{V}_\mathbf{f} (\mathbf{P}^T \mathbf{V}_\mathbf{f})^{-1} \mathbf{P}^T \mathbf{f}(\cdot, t) = \mathbf{\Pi}_{\mathbf{V}_\mathbf{f}, \mathbf{P}} \mathbf{f}(\cdot, t)$$
  - This results in an oblique projection of the full nonlinear vector

Oblique projection of the full nonlinear vector

$$\hat{\mathbf{f}}(\cdot, t) = \mathbf{V_f}(\mathbf{P}^T\mathbf{V_f})^{-1}\mathbf{P}^T\mathbf{f}(\cdot, t) = \mathbf{\Pi}_{\mathbf{V_f},\mathbf{P}}\mathbf{f}(\cdot, t)$$

- $\mathbf{\Pi}_{\mathbf{V},\mathbf{W}} = \mathbf{V}(\mathbf{W}^T\mathbf{V})^{-1}\mathbf{W}^T$: oblique projector onto $\mathbf{V}$ orthogonally to $\mathbf{W}$



Reduced-order dynamical system

- Case where $k_i > k_{\mathbf{f}}$: least-squares reconstruction
  - Idea: $\hat{f}_{i_j}(\mathbf{w}, t) \approx f_{i_j}(\mathbf{w}, t), \ \forall \mathbf{w} \in \mathbb{R}^{N_\mathbf{w}}, \ \forall j = 1, \cdots, N_i$ in the least squares sense
  - Idea: minimize

$$\mathbf{f}_r(\mathbf{q}(t)) = \underset{\mathbf{y}_r}{\operatorname{argmin}} \|\mathbf{P}^T \mathbf{V}_\mathbf{f} \mathbf{y}_r - \mathbf{P}^T \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)\|_2$$

  - Note that $\mathbf{M} = \mathbf{P}^T \mathbf{V}_\mathbf{f} \in \mathbb{R}^{k_i \times k_\mathbf{f}}$ is a skinny matrix
  - One can compute its singular value decomposition

$$\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Z}^T$$

  - The left inverse of $\mathbf{M}$ is then defined as

$$\mathbf{M}^\dagger = \mathbf{Z}\boldsymbol{\Sigma}^\dagger\mathbf{U}^T$$

    where $\boldsymbol{\Sigma}^\dagger = \text{diag}(\frac{1}{\sigma_1}, \cdots, \frac{1}{\sigma_r}, 0, \cdots, 0)$ if $\boldsymbol{\Sigma} = \text{diag}(\sigma_1, \cdots, \sigma_r, 0, \cdots, 0)$ with $\sigma_1 \geq \cdots \sigma_r > 0$
  - Then

$$\hat{\mathbf{f}}(\mathbf{q}(t)) = \mathbf{V}_\mathbf{f}(\mathbf{P}^T\mathbf{V}_\mathbf{f})^\dagger \mathbf{P}^T \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

Greedy function sampling

- This selection takes place after the vectors $[\mathbf{v}_{f,1}, \cdots, \mathbf{v}_{f,k_\mathbf{f}}]$ have already been computed by POD
- Greedy algorithm (Chaturantabut et al. 2010):

1: $[s, i_1] = \max\{|\mathbf{v}_{f,1}|\}$
2: $\mathbf{V_f} = [\mathbf{v}_{f,1}]$, $\mathbf{P} = [\mathbf{e}_{i_1}]$
3: **for** $l = 2 : k_\mathbf{f}$ **do**
4: Solve $\mathbf{P}^T\mathbf{V_f}\mathbf{c} = \mathbf{P}^T\mathbf{v}_{f,l}$ for $\mathbf{c}$
5: $\mathbf{r} = \mathbf{v}_{f,l} - \mathbf{V_f}\mathbf{c}$
6: $[s, i_l] = \max\{|\mathbf{r}|\}$
7: $\mathbf{V}_f = [\mathbf{V_f}, \mathbf{v}_{f,l}]$, $\mathbf{P} = [\mathbf{P}, \ \mathbf{e}_{i_l}]$
8: **end for**

Model reduction at the fully discrete level

- Semi-discrete level: $\frac{d\mathbf{w}}{dt}(t) = \mathbf{f}(\mathbf{w}(t), t)$
- Subspace assumption $\mathbf{w}(t) \approx \mathbf{V}\mathbf{q}(t)$

$$\mathbf{V}\frac{d\mathbf{q}}{dt}(t) \approx \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

- Fully discrete level (implicit, backward Euler scheme)

$$\mathbf{V}\frac{\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}}{\Delta t^{(n)}} \approx \mathbf{f}\left(\mathbf{V}\mathbf{q}^{(n+1)}, t^{(n+1)}\right)$$

- Fully discrete residual

$$\mathbf{r}_D^{(n+1)}(\mathbf{q}^{(n+1)}) = \mathbf{V}\frac{\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}}{\Delta t^{(n)}} - \mathbf{f}\left(\mathbf{V}\mathbf{q}^{(n+1)}, t^{(n+1)}\right)$$

- Model reduction by least-squares (Petrov-Galerkin)

$$\mathbf{q}^{(n+1)} = \underset{\mathbf{y}}{\operatorname{argmin}} \|\mathbf{r}_D^{(n+1)}(\mathbf{y})\|_2$$

- $\mathbf{r}_D(\mathbf{q}^{(n+1)})$ is nonlinear $\Rightarrow$ use the gappy POD idea

Gappy POD at the fully discrete level

- Gappy POD procedure for the fully discrete residual $\mathbf{r}_D$
- Algorithm
  1. Build a reduced basis $\mathbf{V}_\mathbf{r} \in \mathbb{R}^{N_\mathbf{w} \times k_\mathbf{r}}$ for $\mathbf{r}_D$
  2. Construct a sample mesh $\mathcal{I}$ (indices $i_1, \cdots, i_{k_i}$) by a greedy procedure
  3. Consider the gappy approximation

$$\mathbf{r}_D^{(n+1)}(\mathbf{q}^{(n+1)}) \approx \mathbf{V}_\mathbf{r}\mathbf{r}_{k_\mathbf{r}}(\mathbf{q}^{(n+1)}) \approx \mathbf{V}_\mathbf{r}\left(\mathbf{P}^T\mathbf{V}_\mathbf{r}\right)^\dagger \mathbf{P}^T\mathbf{r}^{(n+1)}(\mathbf{V}\mathbf{q}^{(n+1)})$$

  4. Solve

$$\begin{aligned}\mathbf{q}^{(n+1)} &= \underset{\mathbf{y}}{\operatorname{argmin}} \|\mathbf{V}_\mathbf{r}\mathbf{r}_{k_\mathbf{r}}(\mathbf{y})\|_2 \\ &= \underset{\mathbf{y}}{\operatorname{argmin}} \|\mathbf{r}_{k_\mathbf{r}}(\mathbf{y})\|_2 \\ &= \underset{\mathbf{y}}{\operatorname{argmin}} \left\|\left(\mathbf{P}^T\mathbf{V}_\mathbf{r}\right)^\dagger \mathbf{P}^T\mathbf{r}^{(n+1)}(\mathbf{V}\mathbf{y})\right\|_2\end{aligned} \tag{1}$$

Gauss-Newton for nonlinear least squares problem

- Nonlinear least squares problem $\min_{\mathbf{y}} \|\mathbf{r}(\mathbf{y})\|_2$
- Equivalent function to be minimized

$$f(\mathbf{y}) = 0.5\|\mathbf{r}(\mathbf{y})\|_2^2 = \mathbf{r}(\mathbf{y})^T\mathbf{r}(\mathbf{y})$$

- Gradient

$$\nabla f(\mathbf{y}) = \mathbf{J}(\mathbf{y})^T\mathbf{r}(\mathbf{y})$$

  where $\mathbf{J}(\mathbf{y}) = \frac{\partial \mathbf{r}}{\partial \mathbf{y}}(\mathbf{y})$
- Iterative solution using Newton's method $\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \Delta\mathbf{y}^{(k+1)}$ with

$$\nabla^2 f(\mathbf{y}^{(k)})\Delta\mathbf{y}^{(k+1)} = -\nabla f(\mathbf{y}^{(k)})$$

- What is $\nabla^2 f(\mathbf{y})$?

$$\nabla^2 f(\mathbf{y}) = \mathbf{J}(\mathbf{y})^T\mathbf{J}(\mathbf{y}) + \sum_{i=1}^{N} \frac{\partial^2 r_i}{\partial \mathbf{y}^2}(\mathbf{y}) r_i(\mathbf{y})$$

- Gauss-Newton method

$$\nabla^2 f(\mathbf{y}) \approx \mathbf{J}(\mathbf{y})^T\mathbf{J}(\mathbf{y})$$

Gauss-Newton for nonlinear least squares problem

- Gauss-Newton method $\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \Delta\mathbf{y}^{(k+1)}$ with

$$\mathbf{J}(\mathbf{y}^{(k)})^T\mathbf{J}(\mathbf{y}^{(k)})\Delta\mathbf{y}^{(k+1)} = -\mathbf{J}(\mathbf{y}^{(k)})^T\mathbf{r}(\mathbf{y}^{(k)})$$

- This is the normal equation for

$$\Delta\mathbf{y}^{(k+1)} = \underset{\mathbf{z}}{\operatorname{argmin}} \left\|\mathbf{J}(\mathbf{y}^{(k)})\mathbf{z} + \mathbf{r}(\mathbf{y}^{(k)})\right\|_2$$

- QR decomposition of the Jacobian

$$\mathbf{J}(\mathbf{y}^{(k)}) = \mathbf{Q}^{(k)}\mathbf{R}^{(k)}$$

- Equivalent solution using the QR decomposition (assume $\mathbf{R}^{(k)}$ is full rank)

$$\Delta\mathbf{y}^{(k+1)} = -\mathbf{J}(\mathbf{y}^{(k)})^{\dagger}\mathbf{r}(\mathbf{y}^{(k)}) = -\left(\mathbf{R}^{(k)}\right)^{-1}\left(\mathbf{Q}^{(k)}\right)^T\mathbf{r}(\mathbf{y}^{(k)})$$

Gauss-Newton with Approximated Tensors

- GNAT = Gauss-Newton + Gappy POD
- Minimization problem

$$\min_{\mathbf{y}} \left\| \left(\mathbf{P}^T \mathbf{V}_{\mathbf{r}}\right)^{\dagger} \mathbf{P}^T \mathbf{r}^{(n+1)}(\mathbf{V}\mathbf{y}) \right\|_2$$

- Jacobian

$$\mathbf{J}_D(\mathbf{y}) = \left(\mathbf{P}^T \mathbf{V}_{\mathbf{r}}\right)^{\dagger} \mathbf{P}^T \mathbf{J}^{(n+1)}(\mathbf{V}\mathbf{y})$$

- Define a small dimensional operator (constructed offline)

$$\mathbf{A} = \left(\mathbf{P}^T \mathbf{V}_{\mathbf{r}}\right)^{\dagger}$$

- Least-squares problem at iteration k

$$\Delta \mathbf{y}^{(k)} = \underset{\mathbf{z}}{\operatorname{argmin}} \left\| \mathbf{A}\mathbf{P}^T \mathbf{J}^{(n+1)}(\mathbf{V}\mathbf{y}^{(k)})\mathbf{V}\mathbf{z} + \mathbf{A}\mathbf{P}^T \mathbf{r}^{(n+1)}(\mathbf{V}\mathbf{y}^{(k)}) \right\|_2$$

- GNAT solution using QR decomposition $\mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{A}\mathbf{P}^T \mathbf{J}^{(n+1)}(\mathbf{V}\mathbf{y}^{(k)})\mathbf{V}$

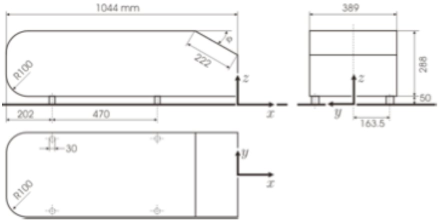
$$\Delta \mathbf{y}^{(k)} = -\left(\mathbf{R}^{(k)}\right)^{-1} \left(\mathbf{Q}^{(k)}\right)^{T} \mathbf{A}\mathbf{P}^T \mathbf{r}^{(n+1)}(\mathbf{V}\mathbf{y}^{(k)})$$

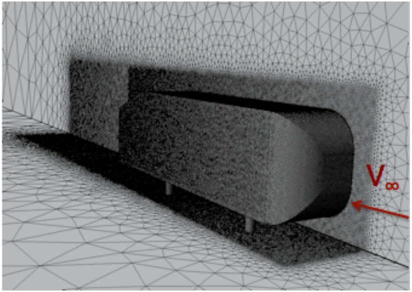
Gauss-Newton with Approximated Tensors

- Further developments
  - Concept of reduced mesh
  - Concept of output mesh
  - Error bounds
  - GNAT using Local reduced bases
- More details in Carlberg et al., JCP 2013

Application 1: compressible Navier-Stokes equations

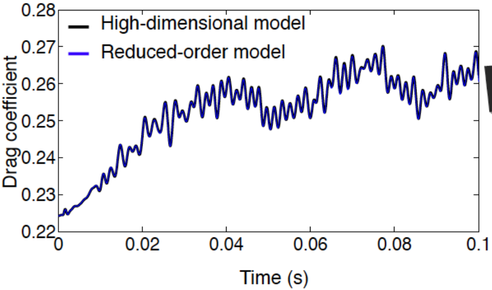
- Flow past the Ahmed body (automotive industry benchmark)
- 3D compressible Navier-Stokes equations
- $N_{\mathbf{w}} = 1.73 \times 10^7$
- $Re = 4.48 \times 10^6$, $M_\infty = 0.175$ (216km/h)
- More details in Carlberg et al., JCP 2013

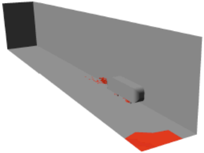




Application 1: compressible Navier-Stokes equations

- Model reduction (POD+GNAT): $k = 283$, $k_{\mathbf{f}} = 1,514$, $k_i = 2,268$

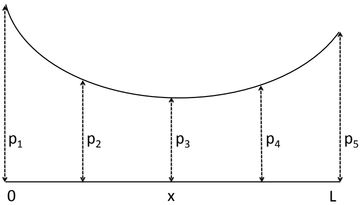




Method	CPU Time	Number of CPUs	Relative Error
Full-Order Model	13.28 h	512	–
ROM (GNAT)	3.88 h	4	0.68%

Application 2: design-optimization of a nozzle

- Full model: $N_{\mathbf{w}} = 2,048$, $p = 5$ shape parameters
- Model reduction (POD+DEIM): $k = 8$, $k_{\mathbf{f}} = 20$, $k_i = 20$

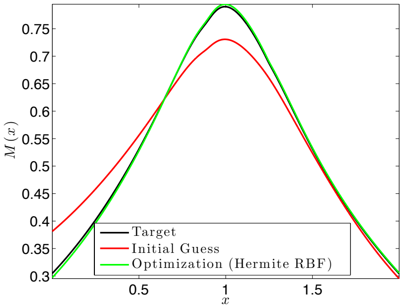


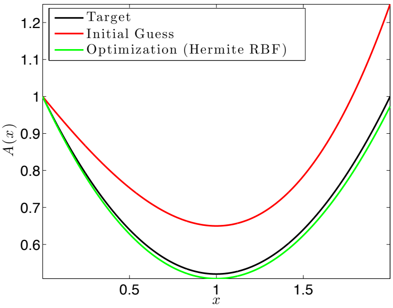
$$\min_{\boldsymbol{\mu} \in \mathbb{R}^5} \|M(\mathbf{w}(\boldsymbol{\mu})) - M_{\text{target}}\|_2$$

$$\text{s.t.} \quad \mathbf{f}(\mathbf{w}(\boldsymbol{\mu})), \boldsymbol{\mu}) = \mathbf{0}$$

Application 2: design-optimization of a nozzle

Method	Offline CPU Time	Online CPU Time	Total CPU Time
Full-Order Model	–	78.8 s	78.8 s
ROM (GNAT)	5.08 s	4.87 s	9.96 s





References

- R. Everson, L. Sirovich. Karhunen–Loeve procedure for gappy data. Journal of the Optical Society of America A 1995; 12(8):1657–1664.
- M. Barrault, Y. Maday, N.C. Nguyen, A.T. Patera. An empirical interpolation method: application to efficient reduced basis discretization of partial differential equations. Comptes Rendus de lAcademie des Sciences Paris 2004; 339:667–672.
- K. Willcox. Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition. Computers and Fluids 2006; 35:208–226.
- S. Chaturantabut, D.C. Sorensen. Nonlinear Model Reduction via Discrete Empirical Interpolation. SIAM Journal on Scientific Computing 2010; 32:2737–2764.

References (continued)

- K. Carlberg, C. Bou-Mosleh, C. Farhat. Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations. International Journal for Numerical Methods in Engineering 2011; 86(2):155–181.
- K. Carlberg, C. Farhat, J. Cortial, D. Amsallem. The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows. Journal of Computational Physics 2013; 242(C): 623–647.
- D. Amsallem, M. Zahr, Y. Choi, C. Farhat. Design Optimization Using Hyper-Reduced-Order Models, submitted for publication – preprint http://www.stanford.edu/~amsallem/HROM_Opt.pdf