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)
  \[ \mathcal{L}(W, x, t; \mu) = 0 \]
- Associated boundary conditions
  \[ \mathcal{B}(W, x_{BC}, t; \mu) = 0 \]
- Initial condition
  \[ W_0(x) = W\text{IC}(x, \mu) \]

- \( W = W(x, t) \in \mathbb{R}^q \): state variable
- \( x \in \Omega \subset \mathbb{R}^d, d \leq 3 \): space variable
- \( t \geq 0 \): time variable
- \( \mu \in \mathcal{D} \subset \mathbb{R}^p \): 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 $N_w = q \times N_{\text{space}}$ ordinary differential equations (ODEs)

\[
\frac{dw}{dt} = f(w, t; \mu)
\]

in terms of the discretized state variable

$w = w(t; \mu) \in \mathbb{R}^{N_w}$

with initial condition $w(0; \mu) = w(\mu)$

- 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 \( \mathbf{V} \)

\[
\mathbf{w}(t) \approx \mathbf{V}\mathbf{q}(t)
\]

- \( q(t) \): reduced (generalized) coordinates

- Inserting in the HDM equation

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

- \( N_w \) 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

\[
\frac{dq}{dt}(t) = V^T f(Vq(t), t)
\]

- \(k\) equations in terms of \(k\) unknowns

- To evaluate \(f_k(Vq(t), t)\):
  1. Compute \(w(t) = Vq(t)\)
  2. Evaluate \(f(Vq(t), t)\)
  3. Left-multiply by \(V^T: V^T f(Vq(t), t)\)

- The computational cost associated with these three steps scales linearly with the dimension \(N_w\) 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 $V$ for the database
  3. For a new face $f$, record a few pixels $f_1, \ldots, f_n$
  4. Using the POD basis $V$, approximately reconstruct the new face $f$
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 $f$ in

$$\frac{dq}{dt} = V^T f(Vq(t), t)$$

- The evaluation of all the entries in the vector $f(\cdot, t)$ is expensive (scales with $N_w$)
- 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 $V_f$
- 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 $f$

- Construction of a POD basis $V_f$ of dimension $k_f$
  1. Collection of snapshots for the nonlinear function from a transient simulation

$$F = \begin{bmatrix} f(w(t_1), t_1), & \cdots, & f(w(t_{m_f}), t_{m_f}) \end{bmatrix} \in \mathbb{R}^{N_w \times m_f}$$

  2. Singular value decomposition

$$F = U_f \Sigma_f Z_f^T$$

  3. Basis truncation ($k_f \ll m_f$)

$$V_f = [u_{f,1}, \cdots, u_{f,k_f}]$$
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 $N_w$-by-$k_i$ matrix
  \[ P = \begin{bmatrix} e_{i_1}, \cdots, e_{i_{k_i}} \end{bmatrix} \]
- At each time $t$, for a given value of the state $w(t) = Vq(t)$, only the entries in the function $f$ corresponding to those indices will be evaluated
  \[ P^T f(w(t), t) = \begin{bmatrix} f_{i_1}(w(t), t) \\ \vdots \\ f_{i_{k_i}}(w(t), t) \end{bmatrix} \]
- This is cheap if $k_i \ll N_w$
- Usually only a subset of the entries in $w(t)$ will be required to construct that vector (case of sparse Jacobian)
Discrete Empirical Interpolation Method

- Case where \( k_i = k_f \): interpolation
  - Idea: \( \hat{f}_{i,j}(w, t) = f_{i,j}(w, t) \), \( \forall w \in \mathbb{R}^{N_w} \), \( \forall j = 1, \ldots, k_i \)
  - This means that
    \[
    P^T \hat{f}(w(t), t) = P^T f(w(t), t)
    \]
  - Remember that \( \hat{f}(\cdot, t) \) belongs to the span of the vectors in \( V_f \), that is
    \[
    \hat{f}(Vq(t), t) = V_f f_r(q(t), t)
    \]
  - Then
    \[
    P^T V_f f_r(q(t), t) = P^T f(Vq(t), t)
    \]
  - Assuming \( P^T V_f \) is nonsingular
    \[
    f_r(q(t), t) = (P^T V_f)^{-1} P^T f(Vq(t), t)
    \]
  - In terms of \( \hat{f}(\cdot, t) \):
    \[
    \hat{f}(\cdot, t) = V_f (P^T V_f)^{-1} P^T f(\cdot, t) = \Pi_{V_f, P} f(\cdot, t)
    \]
  - This results in an oblique projection of the full nonlinear vector

David Amsallem

Gappy POD and GNAT
Oblique projection of the full nonlinear vector

\[ \hat{f}(\cdot,t) = V_f (P^T V_f)^{-1} P^T f(\cdot,t) = \Pi_{V_f,P} f(\cdot,t) \]

- \( \Pi_{V,W} = V (W^T V)^{-1} W^T \): oblique projector onto \( V \) orthogonally to \( W \)

\[ S_2^\perp = \text{span}(W) \]

\[ S_1 = \text{span}(V) \]
Reduced-order dynamical system

- Case where \( k_i > k_f \): least-squares reconstruction
  - Idea: \( \hat{f}_{ij}(w, t) \approx f_{ij}(w, t), \forall w \in \mathbb{R}^{Nw}, \forall j = 1, \cdots, N_i \) in the least squares sense
  - Idea: minimize
    \[
    f_r(q(t)) = \arg\min_{y_r} \| P^T V_f y_r - P^T f(Vq(t), t) \|_2
    \]

- Note that \( M = P^T V_f \in \mathbb{R}^{k_i \times k_f} \) is a skinny matrix
- One can compute its singular value decomposition
  \[
  M = U \Sigma Z^T
  \]

- The left inverse of \( M \) is then defined as
  \[
  M^\dagger = Z \Sigma^\dagger U^T
  \]
  where \( \Sigma^\dagger = \text{diag}(\frac{1}{\sigma_1}, \cdots, \frac{1}{\sigma_r}, 0, \cdots, 0) \) if \( \Sigma = \text{diag}(\sigma_1, \cdots, \sigma_r, 0, \cdots, 0) \) with \( \sigma_1 \geq \cdots \sigma_r > 0 \)

- Then
  \[
  \hat{f}(q(t)) = V_f (P^T V_f)^\dagger P^T f(Vq(t), t)
  \]
Greedy function sampling

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

1: \( [s, i_1] = \max\{|v_{f,1}|\} \)
2: \( V_f = [v_{f,1}], P = [e_{i_1}] \)
3: \textbf{for} \( l = 2 : k_f \) \textbf{do} 
4: \quad \text{Solve} \quad P^T V_f c = P^T v_{f,l} \quad \text{for} \ c 
5: \quad r = v_{f,l} - V_f c 
6: \quad [s, i_l] = \max\{|r|\} 
7: \quad V_f = [V_f, v_{f,l}], P = [P, e_{i_l}] 
8: \textbf{end for}
Model reduction at the fully discrete level

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

\[
V \frac{dq}{dt}(t) \approx f(V\mathbf{q}(t), t)
\]

- **Fully discrete level (implicit, backward Euler scheme)**

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

- **Fully discrete residual**

\[
r_D^{(n+1)}(q^{(n+1)}) = V \frac{q^{(n+1)} - q^{(n)}}{\Delta t^{(n)}} - f \left( Vq^{(n+1)}, t^{(n+1)} \right)
\]

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

\[
q^{(n+1)} = \arg\min_{\mathbf{y}} \| r_D^{(n+1)}(\mathbf{y}) \|_2
\]

- **\( r_D(q^{(n+1)}) \) is nonlinear ⇒ use the gappy POD idea**
Gappy POD at the fully discrete level

- Gappy POD procedure for the fully discrete residual $r_D$
- Algorithm
  1. Build a reduced basis $V_r \in \mathbb{R}^{N_w \times kr}$ for $r_D$
  2. Construct a sample mesh $\mathcal{I}$ (indices $i_1, \cdots, i_{kr}$) by a greedy procedure
  3. Consider the gappy approximation
     \[ r_D^{(n+1)}(q^{(n+1)}) \approx V_r r_{kr}(q^{(n+1)}) \approx V_r \left( P^T V_r \right)^\dagger P^T r^{(n+1)}(Vq^{(n+1)}) \]
  4. Solve
     \[ q^{(n+1)} = \arg\min_y \| V_r r_{kr}(y) \|_2 \]
     \[ = \arg\min_y \| r_{kr}(y) \|_2 \]
     \[ = \arg\min_y \left\| \left( P^T V_r \right)^\dagger P^T r^{(n+1)}(Vq) \right\|_2 \] (1)

David Amsallem  Gappy POD and GNAT
Gauss-Newton for nonlinear least squares problem

- Nonlinear least squares problem $\min_y \|r(y)\|_2$
- Equivalent function to be minimized
  \[ f(y) = 0.5\|r(y)\|_2^2 = r(y)^T r(y) \]
- Gradient
  \[ \nabla f(y) = J(y)^T r(y) \]
  where $J(y) = \frac{\partial r}{\partial y}(y)$
- Iterative solution using Newton’s method $y^{(k+1)} = y^{(k)} + \Delta y^{(k+1)}$ with
  \[ \nabla^2 f(y^{(k)}) \Delta y^{(k+1)} = -\nabla f(y^{(k)}) \]
- What is $\nabla^2 f(y)$?
  \[ \nabla^2 f(y) = J(y)^T J(y) + \sum_{i=1}^{N} \frac{\partial^2 r_i}{\partial y^2}(y) r_i(y) \]
- Gauss-Newton method
  \[ \nabla^2 f(y) \approx J(y)^T J(y) \]
Gauss-Newton for nonlinear least squares problem

- Gauss-Newton method $y^{(k+1)} = y^{(k)} + \Delta y^{(k+1)}$ with
  \[
  J(y^{(k)})^T J(y^{(k)}) \Delta y^{(k+1)} = -J(y^{(k)})^T r(y^{(k)})
  \]

- This is the normal equation for
  \[
  \Delta y^{(k+1)} = \underset{z}{\text{argmin}} \left\| J(y^{(k)})z + r(y^{(k)}) \right\|_2
  \]

- QR decomposition of the Jacobian
  \[
  J(y^{(k)}) = Q^{(k)} R^{(k)}
  \]

- Equivalent solution using the QR decomposition (assume $R^{(k)}$ is full rank)
  \[
  \Delta y^{(k+1)} = -J(y^{(k)})^\dagger r(y^{(k)}) = -\left( R^{(k)} \right)^{-1} \left( Q^{(k)} \right)^T r(y^{(k)})
  \]
Gauss-Newton with Approximated Tensors

- GNAT = Gauss-Newton + Gappy POD
- Minimization problem
  \[
  \min_y \left\| \left( P^T V_r \right)^\dagger P^T r^{(n+1)}(Vy) \right\|_2
  \]
- Jacobian
  \[
  J_D(y) = \left( P^T V_r \right)^\dagger P^T J^{(n+1)}(Vy)
  \]
- Define a small dimensional operator (constructed offline)
  \[
  A = \left( P^T V_r \right)^\dagger
  \]
- Least-squares problem at iteration \( k \)
  \[
  \Delta y^{(k)} = \arg\min_z \left\| AP^T J^{(n+1)}(Vy^{(k)}) Vz + AP^T r^{(n+1)}(Vy^{(k)}) \right\|_2
  \]
- GNAT solution using QR decomposition
  \[
  Q^{(k)} R^{(k)} = AP^T J^{(n+1)}(Vy^{(k)}) V
  \]
  \[
  \Delta y^{(k)} = - \left( R^{(k)} \right)^{-1} \left( Q^{(k)} \right)^T AP^T r^{(n+1)}(Vy^{(k)})
  \]
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
- \( N_w = 1.73 \times 10^7 \)
- \( Re = 4.48 \times 10^6, M_{\infty} = 0.175 \) (216km/h)
- More details in Carlberg et al., JCP 2013
Application 1: compressible Navier-Stokes equations

- Model reduction (POD+GNAT): $k = 283$, $k_f = 1,514$, $k_i = 2,268$

<table>
<thead>
<tr>
<th>Method</th>
<th>CPU Time</th>
<th>Number of CPUs</th>
<th>Relative Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-Order Model</td>
<td>13.28 h</td>
<td>512</td>
<td>–</td>
</tr>
<tr>
<td>ROM (GNAT)</td>
<td>3.88 h</td>
<td>4</td>
<td>0.68%</td>
</tr>
</tbody>
</table>

David Amsallem

Gappy POD and GNAT
Application 2: design-optimization of a nozzle

- Full model: \( N_w = 2,048, p = 5 \) shape parameters
- Model reduction (POD+DEIM): \( k = 8, k_f = 20, k_i = 20 \)

\[
\min_{\mu \in \mathbb{R}^5} \| M(w(\mu)) - M_{\text{target}} \|_2 \\
\text{s.t. } f(w(\mu), \mu) = 0
\]
Application 2: design-optimization of a nozzle

<table>
<thead>
<tr>
<th>Method</th>
<th>Offline CPU Time</th>
<th>Online CPU Time</th>
<th>Total CPU Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-Order Model</td>
<td>–</td>
<td>78.8 s</td>
<td>78.8 s</td>
</tr>
<tr>
<td>ROM (GNAT)</td>
<td>5.08 s</td>
<td>4.87 s</td>
<td>9.96 s</td>
</tr>
</tbody>
</table>

![Graph 1](image1.png)  
![Graph 2](image2.png)
References

References (continued)

