# AA216/CME345: PROJECTION-BASED MODEL ORDER REDUCTION

Local Parametric Approaches

Charbel Farhat
Stanford University
*cfarhat@stanford.edu*

# Outline

- Note: The material covered in this chapter is based on the following published documents:
  - D. Amsallem, C. Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. AIAA Journal 2008; 46(7):1803-1813.
  - D. Amsallem, J. Cortial, C. Farhat. Towards real-time CFD-based aeroelastic computations using a database of reduced-order models. AIAA Journal 2010; 48(9):2029-2037.
  - D. Amsallem, C. Farhat. An online method for interpolating linear parametric reduced-order models. SIAM Journal on Scientific Computing 2011; 33(5): 2169-2198.
  - D. Amsallem, Interpolation on manifolds of CFD-based fluid and finite element-based structural reduced-order models for on-line aeroelastic predictions. Ph.D. Thesis, Stanford University, 2010.

- Parametric linear High-Dimensional (time-invariant) Model (HDM)

$$\begin{aligned} \frac{d\mathbf{w}}{dt}(t;\boldsymbol{\mu}) &= \mathbf{A}(\boldsymbol{\mu})\mathbf{w}(t;\boldsymbol{\mu}) + \mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t) \\ \mathbf{y}(t;\boldsymbol{\mu}) &= \mathbf{C}(\boldsymbol{\mu})\mathbf{w}(t;\boldsymbol{\mu}) + \mathbf{D}(\boldsymbol{\mu})\mathbf{u}(t) \\ \mathbf{w}(0;\boldsymbol{\mu}) &= \mathbf{w}_0(\boldsymbol{\mu}) \end{aligned}$$

- Parametric nonlinear HDM

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

- $\mathbf{w} \in \mathbb{R}^N$: Vector of state variables
- $\mathbf{u} \in \mathbb{R}^{\text{in}}$: Vector of input variables – typically in $\ll N$
- $\mathbf{y} \in \mathbb{R}^q$: Vector of output variables – typically $q \ll N$
- $\boldsymbol{\mu} \in \mathcal{D} \subset \mathbb{R}^p$: Vector of parameters – typically $p \ll N$

- Parametric linear (time-invariant) HDM
  - goal: Construct a corresponding parametric Projection-based Reduced-Order Model (PROM) using a **local** rather than global approach

$$\begin{aligned}\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) &= \mathbf{A}_r(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu}) + \mathbf{B}_r(\boldsymbol{\mu})\mathbf{u}(t)\\ \mathbf{y}(t;\boldsymbol{\mu}) &= \mathbf{C}_r(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu}) + \mathbf{D}_r(\boldsymbol{\mu})\mathbf{u}(t)\end{aligned}$$

    - based on **local** Reduced-Order Bases (ROBs) $\left(\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right),\ \mathbf{W}\left(\boldsymbol{\mu}^{(\ell)}\right)\right)$ and the approximation

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

    - $\boldsymbol{\mu}^{(\ell)} \in \mathcal{D}$; $\mathbf{q} \in \mathbb{R}^k$
    - all local ROBs have the same dimension $k \ll N$
  - local PROM operators resulting from Petrov-Galerkin projection

$$\begin{aligned}\mathbf{A}_r(\boldsymbol{\mu}) &= (\mathbf{W}(\boldsymbol{\mu})^T\mathbf{V}(\boldsymbol{\mu}))^{-1}\mathbf{W}(\boldsymbol{\mu})^T\mathbf{A}(\boldsymbol{\mu})\mathbf{V}(\boldsymbol{\mu}) \in \mathbb{R}^{k\times k}\\ \mathbf{B}_r(\boldsymbol{\mu}) &= (\mathbf{W}(\boldsymbol{\mu})^T\mathbf{V}(\boldsymbol{\mu}))^{-1}\mathbf{W}(\boldsymbol{\mu})^T\mathbf{B}(\boldsymbol{\mu}) \in \mathbb{R}^{k\times p}\\ \mathbf{C}_r(\boldsymbol{\mu}) &= \mathbf{C}(\boldsymbol{\mu})\mathbf{V}(\boldsymbol{\mu}) \in \mathbb{R}^{q\times k}; \quad \mathbf{D}_r(\boldsymbol{\mu}) = \mathbf{D}(\boldsymbol{\mu}) \in \mathbb{R}^{q\times p}\end{aligned}$$

- Parametric nonlinear HDM
  - goal: Construct a corresponding parametric PROM using a local rather than global approach

$$\begin{aligned}\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) &= \mathbf{f}_r(\mathbf{q}(t),t;\boldsymbol{\mu}) + \mathbf{B}_r(\boldsymbol{\mu})\mathbf{u}(t) \\ \mathbf{y}(t;\boldsymbol{\mu}) &= \mathbf{C}_r(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu}) + \mathbf{D}_r(\boldsymbol{\mu})\mathbf{u}(t)\end{aligned}$$

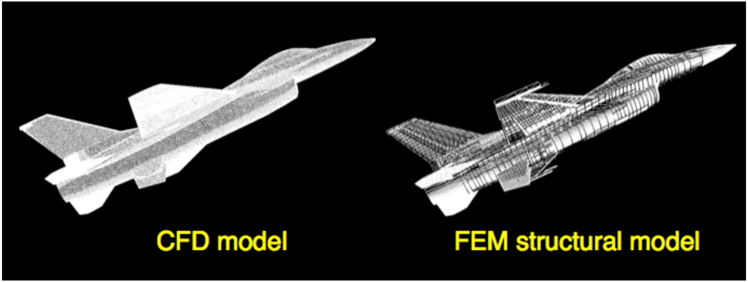
  - based on **local** ROBs $\left(\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{W}\left(\boldsymbol{\mu}^{(\ell)}\right)\right)$ and the approximation

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

  - $\boldsymbol{\mu}^{(\ell)} \in \mathcal{D}$; $\mathbf{q} \in \mathbb{R}^k$
  - all local ROBs have the same dimension $k \ll N$
  - local PROM resulting from Petrov-Galerkin projection

$$\mathbf{f}_r(\mathbf{q}(t),t;\boldsymbol{\mu}) = \left(\mathbf{W}\left(\boldsymbol{\mu}\right)^T\mathbf{V}\left(\boldsymbol{\mu}\right)\right)^{-1}\mathbf{W}(\boldsymbol{\mu})^T\mathbf{f}(\mathbf{V}(\boldsymbol{\mu})\mathbf{q}(t),t;\boldsymbol{\mu}) \in \mathbb{R}^k$$

- Parametric, linearized, aeroelastic identification of a F-16 Block 40 aircraft in clean wing configuration (e.g., for flutter analysis in the time domain)
- Linearized coupled fluid-structure system around an aeroelastic equilibrium position
- Hundreds of flight conditions $\boldsymbol{\mu} = (M_\infty, \alpha)$ for flutter clearance




- $N_{\text{fluid}} \approx 2 \times 10^6$, $N_{\text{structure}} \approx 1.6 \times 10^5$

- Consider the following procedure
  1. construct local ROBs $\left(\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right), \mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)\right)$ at the parametric flight condition $\boldsymbol{\mu}^{(1)}$

- Consider the following procedure
  1. construct local ROBs $\left(\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right), \mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)\right)$ at the parametric flight condition $\boldsymbol{\mu}^{(1)}$
  2. avoid reconstructing new local ROBs every time the flight condition is varied and thus use the local ROBs constructed at $\boldsymbol{\mu}^{(1)}$ to reduce the HDM at $\boldsymbol{\mu}^{(2)}$

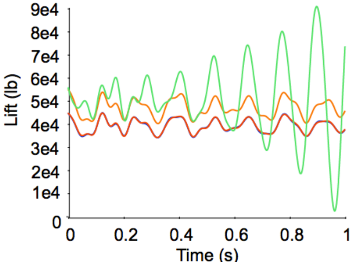
- Consider the following procedure
  1. construct local ROBs $\left(\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right), \mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)\right)$ at the parametric flight condition $\boldsymbol{\mu}^{(1)}$
  2. avoid reconstructing new local ROBs every time the flight condition is varied and thus use the local ROBs constructed at $\boldsymbol{\mu}^{(1)}$ to reduce the HDM at $\boldsymbol{\mu}^{(2)}$
  3. build the following local PROM

$$
\begin{aligned}
\frac{d\mathbf{q}}{dt}\left(t; \boldsymbol{\mu}^{(2)}\right) &= \mathbf{A}_r\left(\boldsymbol{\mu}^{(2)}\right)\mathbf{q}\left(t; \boldsymbol{\mu}^{(2)}\right) + \mathbf{B}_r\left(\boldsymbol{\mu}^{(2)}\right)\mathbf{u}(t) \\
\mathbf{y}\left(t; \boldsymbol{\mu}^{(2)}\right) &= \mathbf{C}_r\left(\boldsymbol{\mu}^{(2)}\right)\mathbf{q}\left(t; \boldsymbol{\mu}^{(2)}\right) + \mathbf{D}_r\left(\boldsymbol{\mu}^{(2)}\right)\mathbf{u}(t) \\
\mathbf{w}\left(t, \boldsymbol{\mu}^{(2)}\right) &\approx \mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right)\mathbf{q}\left(t; \boldsymbol{\mu}^{(2)}\right)
\end{aligned}
$$

where

$$
\begin{aligned}
\mathbf{A}_r\left(\boldsymbol{\mu}^{(2)}\right) &= \left(\mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)^T\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right)\right)^{-1}\mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)^T\mathbf{A}\left(\boldsymbol{\mu}^{(2)}\right)\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right) \in \mathbb{R}^{k\times k} \\
\mathbf{B}_r\left(\boldsymbol{\mu}^{(2)}\right) &= \left(\mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)^T\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right)\right)^{-1}\mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)^T\mathbf{B}\left(\boldsymbol{\mu}^{(2)}\right) \in \mathbb{R}^{k\times p} \\
\mathbf{C}_r\left(\boldsymbol{\mu}^{(2)}\right) &= \mathbf{C}\left(\boldsymbol{\mu}^{(2)}\right)\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right) \in \mathbb{R}^{q\times k}; \quad \mathbf{D}_r\left(\boldsymbol{\mu}^{(2)}\right) = \mathbf{D}\left(\boldsymbol{\mu}^{(2)}\right) \in \mathbb{R}^{q\times p}
\end{aligned}
$$

- Queried flight conditions
  - $\boldsymbol{\mu}^{(1)} = \left(M_\infty^{(1)}, \alpha^{(1)}\right) = (0.71, \alpha_{\text{trimmed}}(0.71))$
  - $\boldsymbol{\mu}^{(2)} = \left(M_\infty^{(2)}, \alpha^{(2)}\right) = (0.8, \alpha_{\text{trimmed}}(0.8))$




- *HDM* $(\boldsymbol{\mu}^{(1)})$
- *PROM* $(\boldsymbol{\mu}^{(1)})$
- *HDM* $(\boldsymbol{\mu}^{(2)})$
- *PROM* $(\boldsymbol{\mu}^{(2)})$

$\Rightarrow$ the local ROBs lack robustness with respect to parameter changes

- The lack of robustness of the local ROBs with respect to parameter changes implies that they should be **reconstructed** every time the parameters are varied

- Alternative procedure: Given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  1. construct the HDM operators $\mathbf{A}(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}(\mathbf{w}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}(\boldsymbol{\mu}^\star)$, $\mathbf{C}(\boldsymbol{\mu}^\star)$, and $\mathbf{D}(\boldsymbol{\mu}^\star)$

- The lack of robustness of the local ROBs with respect to parameter changes implies that they should be **reconstructed** every time the parameters are varied

- Alternative procedure: Given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  1. construct the HDM operators $\mathbf{A}(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}(\mathbf{w}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}(\boldsymbol{\mu}^\star)$, $\mathbf{C}(\boldsymbol{\mu}^\star)$, and $\mathbf{D}(\boldsymbol{\mu}^\star)$
  2. generate the local ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred approach

- The lack of robustness of the local ROBs with respect to parameter changes implies that they should be **reconstructed** every time the parameters are varied

- Alternative procedure: Given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  1. construct the HDM operators $\mathbf{A}(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}(\mathbf{w}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}(\boldsymbol{\mu}^\star)$, $\mathbf{C}(\boldsymbol{\mu}^\star)$, and $\mathbf{D}(\boldsymbol{\mu}^\star)$
  2. generate the local ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred approach
  3. construct the local PROM operators $\mathbf{A}_r(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}_r(\mathbf{q}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}_r(\boldsymbol{\mu}^\star)$, $\mathbf{C}_r(\boldsymbol{\mu}^\star)$, and $\mathbf{D}_r(\boldsymbol{\mu}^\star)$ using a preferred Petrov-Galerkin Projection-Based Model Order Reduction (PMOR) method

- The lack of robustness of the local ROBs with respect to parameter changes implies that they should be **reconstructed** every time the parameters are varied

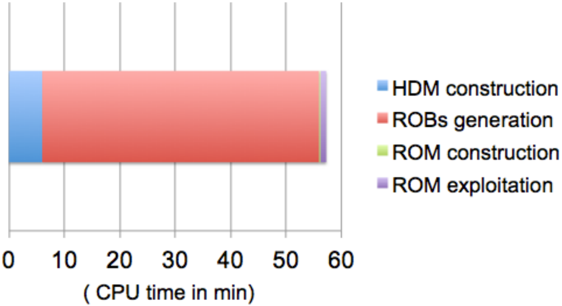
- Alternative procedure: Given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  1. construct the HDM operators $\mathbf{A}(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}(\mathbf{w}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}(\boldsymbol{\mu}^\star)$, $\mathbf{C}(\boldsymbol{\mu}^\star)$, and $\mathbf{D}(\boldsymbol{\mu}^\star)$
  2. generate the local ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred approach
  3. construct the local PROM operators $\mathbf{A}_r(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}_r(\mathbf{q}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}_r(\boldsymbol{\mu}^\star)$, $\mathbf{C}_r(\boldsymbol{\mu}^\star)$, and $\mathbf{D}_r(\boldsymbol{\mu}^\star)$ using a preferred Petrov-Galerkin Projection-Based Model Order Reduction (PMOR) method
  4. exploit the constructed local Petrov-Galerkin PROM

- The lack of robustness of the local ROBs with respect to parameter changes implies that they should be **reconstructed** every time the parameters are varied

- Alternative procedure: Given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  1. construct the HDM operators $\mathbf{A}(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}(\mathbf{w}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}(\boldsymbol{\mu}^\star)$, $\mathbf{C}(\boldsymbol{\mu}^\star)$, and $\mathbf{D}(\boldsymbol{\mu}^\star)$
  2. generate the local ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred approach
  3. construct the local PROM operators $\mathbf{A}_r(\boldsymbol{\mu}^\star)$ (linear setting) or $\mathbf{f}_r(\mathbf{q}(t), t; \boldsymbol{\mu}^\star)$ (nonlinear setting), $\mathbf{B}_r(\boldsymbol{\mu}^\star)$, $\mathbf{C}_r(\boldsymbol{\mu}^\star)$, and $\mathbf{D}_r(\boldsymbol{\mu}^\star)$ using a preferred Petrov-Galerkin Projection-Based Model Order Reduction (PMOR) method
  4. exploit the constructed local Petrov-Galerkin PROM

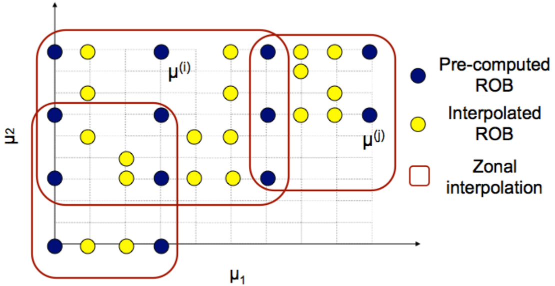
- Question: Is this procedure computationally **efficient**?

- Construction and exploitation in $t \in [0, 1]$ s of a local, linearized, aeroelastic F-16 PROM




- The direct generation of a pair of local ROBs accounts for 89% of the total CPU time
- The overall procedure takes 56 minutes, which renders this approach **non-amenable to real-time parametric applications**

- Idea
  - pre-compute local ROBs at a number of sampled parameter points $\left\{\boldsymbol{\mu}^{(\ell)} \in \mathcal{D}\right\}_{\ell=1}^{N_s}$
  - interpolate these ROBs to obtain a local ROB at a queried but unsampled parameter $\boldsymbol{\mu}^\star \notin \left\{\boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}$




  - **Question: How does one interpolate local ROBs?**
- For simplicity, assume an orthogonal Galerkin projection

- Tempting idea: Interpolate the matrices $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right) \in \mathbb{R}^{N \times k}$ entry-by-entry (linear interpolation on the manifold $\mathbb{R}^{Nk}$)

- Tempting idea: Interpolate the matrices $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right) \in \mathbb{R}^{N \times k}$ entry-by-entry (linear interpolation on the manifold $\mathbb{R}^{Nk}$)
- Input
  - queried parameter $\boldsymbol{\mu}^{\star}$
  - pre-computed ROBs $\left\{\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\right\}_{\ell=1}^{N_s}$
  - multi-variate interpolator $\mathcal{I}$ in $\mathbb{R}^p$

$$a(\boldsymbol{\mu}) = \mathcal{I}\left(\boldsymbol{\mu}; \left\{a\left(\boldsymbol{\mu}^{(\ell)}\right), \boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}\right)$$

- Tempting idea: Interpolate the matrices $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right) \in \mathbb{R}^{N \times k}$ entry-by-entry (linear interpolation on the manifold $\mathbb{R}^{Nk}$)
- Input
  - queried parameter $\boldsymbol{\mu}^\star$
  - pre-computed ROBs $\left\{\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\right\}_{\ell=1}^{N_s}$
  - multi-variate interpolator $\mathcal{I}$ in $\mathbb{R}^p$

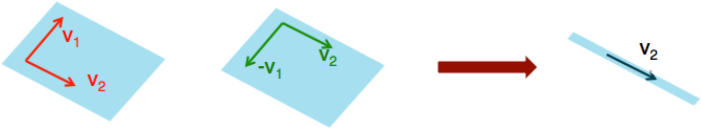
$$a(\boldsymbol{\mu}) = \mathcal{I}\left(\boldsymbol{\mu}; \left\{a\left(\boldsymbol{\mu}^{(\ell)}\right), \boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}\right)$$

- Algorithm

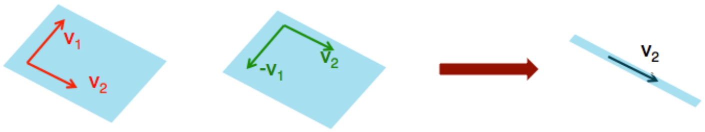
1: **for** $i = 1 : N$ **do**
2:     **for** $j = 1 : k$ **do**
3:         compute $v_{ij}(\boldsymbol{\mu}^\star) = \mathcal{I}\left(\boldsymbol{\mu}^\star; \left\{v_{ij}\left(\boldsymbol{\mu}^{(\ell)}\right), \boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}\right)$
4:     **end for**
5: **end for**
6: form $\mathbf{V}(\boldsymbol{\mu}^\star) = [v_{ij}(\boldsymbol{\mu}^\star)]$

- Example
  - $N = 3,\ k = 2,\ p = 1$
  - for $\boldsymbol{\mu}^{(1)} = 0$: $\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right) = \mathbf{V}(0) = (\mathbf{v}_1\ \mathbf{v}_2)$
  - for $\boldsymbol{\mu}^{(2)} = 1$: $\mathbf{V}\left(\boldsymbol{\mu}^{(2)}\right) = \mathbf{V}(1) = (-\mathbf{v}_1\ \mathbf{v}_2)$
  - queried but unsampled parameter $\boldsymbol{\mu} = 0.5$
  - linear interpolation

- Example
  - $N = 3,\ k = 2,\ p = 1$
  - for $\boldsymbol{\mu}^{(1)} = 0$: $\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right) = \mathbf{V}(0) = (\mathbf{v}_1\ \mathbf{v}_2)$
  - for $\boldsymbol{\mu}^{(2)} = 1$: $\mathbf{V}\left(\boldsymbol{\mu}^{(2)}\right) = \mathbf{V}(1) = (-\mathbf{v}_1\ \mathbf{v}_2)$
  - queried but unsampled parameter $\boldsymbol{\mu} = 0.5$
  - linear interpolation
- Interpolatory result
  $\mathbf{V}(0.5) = 0.5\,(\mathbf{V}(0) + \mathbf{V}(1)) = (0.5(\mathbf{v}_1 - \mathbf{v}_1)\ 0.5(\mathbf{v}_2 + \mathbf{v}_2)) = (\mathbf{0}\ \mathbf{v}_2)$



- Example
  - $N = 3,\ k = 2,\ p = 1$
  - for $\boldsymbol{\mu}^{(1)} = 0$: $\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right) = \mathbf{V}(0) = (\mathbf{v}_1\ \mathbf{v}_2)$
  - for $\boldsymbol{\mu}^{(2)} = 1$: $\mathbf{V}\left(\boldsymbol{\mu}^{(2)}\right) = \mathbf{V}(1) = (-\mathbf{v}_1\ \mathbf{v}_2)$
  - queried but unsampled parameter $\boldsymbol{\mu} = 0.5$
  - linear interpolation
- Interpolatory result
  $\mathbf{V}(0.5) = 0.5\,(\mathbf{V}(0) + \mathbf{V}(1)) = (0.5(\mathbf{v}_1 - \mathbf{v}_1)\ 0.5(\mathbf{v}_2 + \mathbf{v}_2)) = (\mathbf{0}\ \mathbf{v}_2)$




- What went wrong?
  - a **relevant constraint** was neither identified nor preserved
  - the **wrong entity** was interpolated

- Reduced-order equation
  - linear (time-invariant) system

$$\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{A}(\boldsymbol{\mu})\mathbf{V}(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu}) + \mathbf{V}(\boldsymbol{\mu})^T\mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t)$$

  - nonlinear system

$$\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{f}(\tilde{\mathbf{w}}(t), t;\boldsymbol{\mu})$$

- Reduced-order equation
  - linear (time-invariant) system

$$\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{A}(\boldsymbol{\mu})\mathbf{V}(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu}) + \mathbf{V}(\boldsymbol{\mu})^T\mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t)$$

  - nonlinear system

$$\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{f}(\tilde{\mathbf{w}}(t), t;\boldsymbol{\mu})$$

- Equivalent high-dimensional equations for $\tilde{\mathbf{w}}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu})$

$$\frac{d\tilde{\mathbf{w}}}{dt}(t;\boldsymbol{\mu}) = \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{A}(\boldsymbol{\mu})\tilde{\mathbf{w}}(t;\boldsymbol{\mu}) + \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t) \quad \text{(linear)}$$

$$\frac{d\tilde{\mathbf{w}}}{dt}(t;\boldsymbol{\mu}) = \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{f}(\tilde{\mathbf{w}}(t), t;\boldsymbol{\mu}) + \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t) \quad \text{(nonlinear)}$$

- Reduced-order equation
  - linear (time-invariant) system

$$\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{A}(\boldsymbol{\mu})\mathbf{V}(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu}) + \mathbf{V}(\boldsymbol{\mu})^T\mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t)$$

  - nonlinear system

$$\frac{d\mathbf{q}}{dt}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{f}(\tilde{\mathbf{w}}(t), t;\boldsymbol{\mu})$$

- Equivalent high-dimensional equations for $\tilde{\mathbf{w}}(t;\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})\mathbf{q}(t;\boldsymbol{\mu})$

$$\frac{d\tilde{\mathbf{w}}}{dt}(t;\boldsymbol{\mu}) = \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{A}(\boldsymbol{\mu})\tilde{\mathbf{w}}(t;\boldsymbol{\mu}) + \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t) \quad \text{(linear)}$$

$$\frac{d\tilde{\mathbf{w}}}{dt}(t;\boldsymbol{\mu}) = \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{f}(\tilde{\mathbf{w}}(t), t;\boldsymbol{\mu}) + \boldsymbol{\Pi}_{\mathbf{V}(\boldsymbol{\mu}),\mathbf{V}(\boldsymbol{\mu})}\mathbf{B}(\boldsymbol{\mu})\mathbf{u}(t) \quad \text{(nonlinear)}$$

- In both cases, the PROM solution is independent of the choice of ROB associated with the projection subspace

$\Longrightarrow$ the **correct entity to interpolate** is $\mathcal{S}(\boldsymbol{\mu}) = \text{range}(\mathbf{V}(\boldsymbol{\mu}))$

- A subspace $\mathcal{S}$ is typically represented by a ROB

- A subspace $\mathcal{S}$ is typically represented by a ROB
- The appropriate choice of a ROB is not unique

$$\mathcal{S} = \text{range}(\mathbf{V}) = \text{range}(\mathbf{VQ}), \ \forall \, \mathbf{Q} \in \text{GL}(k)$$

- A subspace $\mathcal{S}$ is typically represented by a ROB
- The appropriate choice of a ROB is not unique

$$\mathcal{S} = \text{range}(\mathbf{V}) = \text{range}(\mathbf{VQ}), \ \forall\, \mathbf{Q} \in \text{GL}(k)$$

- A subspace is a linear special case of a manifold; manifolds locally resemble vector spaces, with tangent spaces that are linear subspaces

- A subspace $\mathcal{S}$ is typically represented by a ROB
- The appropriate choice of a ROB is not unique

$$\mathcal{S} = \text{range}(\mathbf{V}) = \text{range}(\mathbf{VQ}), \ \forall\, \mathbf{Q} \in \text{GL}(k)$$

- A subspace is a linear special case of a manifold; manifolds locally resemble vector spaces, with tangent spaces that are linear subspaces
- Manifolds of interest
  - $\mathcal{G}(k, N)$ (Grassmann manifold): Set of subspaces in $\mathbb{R}^N$ of dimension $k$
  - $\mathcal{ST}(k, N)$ (orthogonal Stiefel manifold): Set of orthogonal ROB matrices in $\mathbb{R}^{N\times k}$
  - $\text{GL}(k)$ (general linear group): Set of nonsingular square matrices of size $k$
  - $\mathcal{O}(k)$: Set of orthogonal square matrices of size $k$

- A subspace $\mathcal{S}$ is typically represented by a ROB
- The appropriate choice of a ROB is not unique

$$\mathcal{S} = \text{range}(\mathbf{V}) = \text{range}(\mathbf{VQ}), \ \forall \, \mathbf{Q} \in \text{GL}(k)$$

- A subspace is a linear special case of a manifold; manifolds locally resemble vector spaces, with tangent spaces that are linear subspaces
- Manifolds of interest
  - $\mathcal{G}(k, N)$ (Grassmann manifold): Set of subspaces in $\mathbb{R}^N$ of dimension $k$
  - $\mathcal{ST}(k, N)$ (orthogonal Stiefel manifold): Set of orthogonal ROB matrices in $\mathbb{R}^{N \times k}$
  - $\text{GL}(k)$ (general linear group): Set of nonsingular square matrices of size $k$
  - $\mathcal{O}(k)$: Set of orthogonal square matrices of size $k$
- Properties
  - $\mathcal{O}(k) \subset \text{GL}(k)$
  - $\mathcal{ST}(N, N) = \mathcal{O}(N)$

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$
- Equivalence class

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$
- Equivalence class
  - $\mathcal{S}(\boldsymbol{\mu}) = \text{range}(\mathbf{V}(\boldsymbol{\mu})) = \text{range}(\mathbf{V}(\boldsymbol{\mu})\mathbf{Q}), \ \forall\, \mathbf{Q} \in \mathcal{O}(k)$

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$
- Equivalence class
  - $\mathcal{S}(\boldsymbol{\mu}) = \text{range}(\mathbf{V}(\boldsymbol{\mu})) = \text{range}(\mathbf{V}(\boldsymbol{\mu})\mathbf{Q}), \ \forall\, \mathbf{Q} \in \mathcal{O}(k)$
  - an element of the Grassmann manifold defines an entire **class of equivalence on the Stiefel manifold**

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$
- Equivalence class
  - $\mathcal{S}(\boldsymbol{\mu}) = \text{range}(\mathbf{V}(\boldsymbol{\mu})) = \text{range}(\mathbf{V}(\boldsymbol{\mu})\mathbf{Q}), \ \forall\, \mathbf{Q} \in \mathcal{O}(k)$
  - an element of the Grassmann manifold defines an entire **class of equivalence on the Stiefel manifold**
  - this class of equivalence is defined by the range operation

$$\begin{aligned} \forall\, \mathbf{V}_1, \mathbf{V}_2 \in \mathcal{ST}(k, N), \ \mathbf{V}_1 \sim \mathbf{V}_2 \quad &\Leftrightarrow \quad \text{range}(\mathbf{V}_1) = \text{range}(\mathbf{V}_2) \\ &\Leftrightarrow \quad \exists\, \mathbf{Q} \in \mathcal{O}(k) \text{ s.t } \mathbf{V}_1 = \mathbf{V}_2\mathbf{Q} \end{aligned}$$

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$
- Equivalence class
  - $\mathcal{S}(\boldsymbol{\mu}) = \text{range}(\mathbf{V}(\boldsymbol{\mu})) = \text{range}(\mathbf{V}(\boldsymbol{\mu})\mathbf{Q}), \ \forall\, \mathbf{Q} \in \mathcal{O}(k)$
  - an element of the Grassmann manifold defines an entire **class of equivalence on the Stiefel manifold**
  - this class of equivalence is defined by the range operation

$$\begin{aligned} \forall\, \mathbf{V}_1, \mathbf{V}_2 \in \mathcal{ST}(k, N), \ \mathbf{V}_1 \sim \mathbf{V}_2 \quad &\Leftrightarrow \quad \text{range}(\mathbf{V}_1) = \text{range}(\mathbf{V}_2) \\ &\Leftrightarrow \quad \exists\, \mathbf{Q} \in \mathcal{O}(k) \text{ s.t } \mathbf{V}_1 = \mathbf{V}_2\mathbf{Q} \end{aligned}$$

  - therefore, the Grassmann manifold is a **quotient manifold** denoted as

$$\mathcal{G}(k, N) = \mathcal{ST}(k, N)/\mathcal{O}(k)$$

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$
- Equivalence class
  - $\mathcal{S}(\boldsymbol{\mu}) = \text{range}(\mathbf{V}(\boldsymbol{\mu})) = \text{range}(\mathbf{V}(\boldsymbol{\mu})\mathbf{Q}), \ \forall \, \mathbf{Q} \in \mathcal{O}(k)$
  - an element of the Grassmann manifold defines an entire **class of equivalence on the Stiefel manifold**
  - this class of equivalence is defined by the range operation
$$\begin{aligned} \forall \, \mathbf{V}_1, \mathbf{V}_2 \in \mathcal{ST}(k, N), \ \mathbf{V}_1 \sim \mathbf{V}_2 \quad &\Leftrightarrow \quad \text{range}(\mathbf{V}_1) = \text{range}(\mathbf{V}_2) \\ &\Leftrightarrow \quad \exists \, \mathbf{Q} \in \mathcal{O}(k) \text{ s.t } \mathbf{V}_1 = \mathbf{V}_2\mathbf{Q} \end{aligned}$$
  - therefore, the Grassmann manifold is a **quotient manifold** denoted as
$$\mathcal{G}(k, N) = \mathcal{ST}(k, N)/\mathcal{O}(k)$$
- Hence, one should interpolate subspaces, but has access in practice to (orthogonal) ROBs

- Case of PMOR with orthogonal ROBs
  - $\mathbf{V}(\boldsymbol{\mu}) \in \mathcal{ST}(k, N)$
  - $\text{range}(\mathbf{V}(\boldsymbol{\mu})) \in \mathcal{G}(k, N)$
- Equivalence class
  - $\mathcal{S}(\boldsymbol{\mu}) = \text{range}(\mathbf{V}(\boldsymbol{\mu})) = \text{range}(\mathbf{V}(\boldsymbol{\mu})\mathbf{Q}), \ \forall\, \mathbf{Q} \in \mathcal{O}(k)$
  - an element of the Grassmann manifold defines an entire **class of equivalence on the Stiefel manifold**
  - this class of equivalence is defined by the range operation

$$\begin{aligned} \forall\, \mathbf{V}_1, \mathbf{V}_2 \in \mathcal{ST}(k, N), \ \mathbf{V}_1 \sim \mathbf{V}_2 \quad &\Leftrightarrow \quad \text{range}(\mathbf{V}_1) = \text{range}(\mathbf{V}_2) \\ &\Leftrightarrow \quad \exists\, \mathbf{Q} \in \mathcal{O}(k) \text{ s.t } \mathbf{V}_1 = \mathbf{V}_2\mathbf{Q} \end{aligned}$$

  - therefore, the Grassmann manifold is a **quotient manifold** denoted as

$$\mathcal{G}(k, N) = \mathcal{ST}(k, N)/\mathcal{O}(k)$$

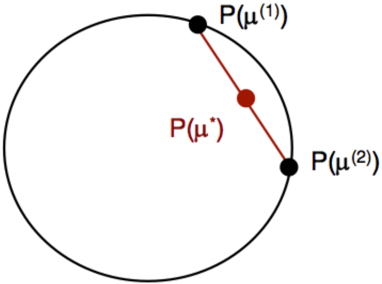
- Hence, one should interpolate subspaces, but has access in practice to (orthogonal) ROBs
- Solution: Perform interpolation on the Grasmann manifold using entities belonging to the (orthogonal) Stiefel manifold

- Quotient matrix manifold
  - the Grassmann manifold

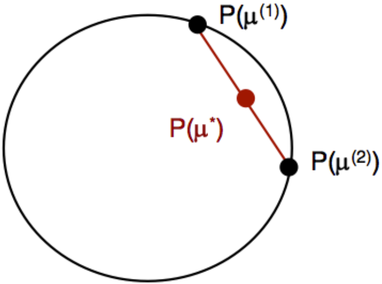
- Embedded matrix manifolds[1]
  - the sphere
  $$\mathbb{S}(N) = \left\{\mathbf{w} \in \mathbb{R}^N \text{ s.t. } \|\mathbf{w}\|_2 = 1\right\} \subset \mathbb{R}^N$$
  - the manifold of orthogonal matrices
  $$\mathcal{O}(N) = \left\{\mathbf{M} \in \mathbb{R}^{N\times N} \text{ s.t. } \mathbf{M}^T\mathbf{M} = \mathbf{I}_N\right\} \subset \mathbb{R}^{N\times N}$$
  - the general linear group
  $$\mathrm{GL}(N) = \left\{\mathbf{M} \in \mathbb{R}^{N\times N} \text{ s.t. } \det(\mathbf{M}) \neq 0\right\} \subset \mathbb{R}^{N\times N}$$
  - the manifold of symmetric positive definite matrices
  $$\mathrm{SPD}(N) = \left\{\mathbf{M} \in \mathbb{R}^{N\times N} \text{ s.t. } \mathbf{M} = \mathbf{M}^T \& \ \mathbf{w}^T\mathbf{M}\mathbf{w} > 0 \ \forall \mathbf{w} \neq \mathbf{0}\right\} \subset \mathbb{R}^{N\times N}$$
  - the orthogonal Stiefel manifold
  $$\mathcal{ST}(k, N) = \left\{\mathbf{M} \in \mathbb{R}^{N\times k} \text{ s.t. } \mathbf{M}^T\mathbf{M} = \mathbf{I}_k\right\} \subset \mathbb{R}^{N\times k}$$

[1] In differential geometry, a manifold is said to be embedded if it can be placed in a higher-dimensional space such that the topology and smooth structure of the manifold are preserved within that space

- Example: The circle (sphere $\mathbb{S}(N)$ for $N = 2$)

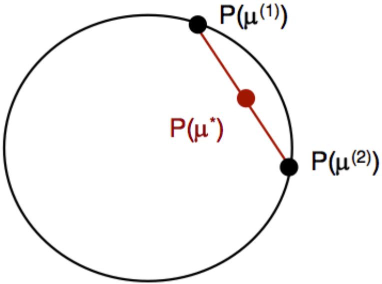


- Example: The circle (sphere $\mathbb{S}(N)$ for $N = 2$)




- Standard interpolation fails to preserve a nonlinear manifold (essentially because standard interpolation applies only in vector spaces)

- Example: The circle (sphere $\mathbb{S}(N)$ for $N = 2$)




- Standard interpolation fails to preserve a nonlinear manifold (essentially because standard interpolation applies only in vector spaces)
- Idea: perform interpolation in a linear space $\Rightarrow$ on a **tangent space of the manifold**

- Input
  - pre-computed matrices $\left\{\mathbf{A}\left(\boldsymbol{\mu}^{(\ell)}\right) \in \mathbb{R}^{N\times M}\right\}_{\ell=1}^{N_s}$
  - map $m_{\mathbf{A}}$ from the manifold $\mathcal{M}$ to the tangent space of $\mathcal{M}$ at the point $\mathbf{A}$
  - multi-variate interpolator $\mathcal{I}$ in $\mathbb{R}^p$
    $a(\boldsymbol{\mu}) = \mathcal{I}\left(\boldsymbol{\mu}; \left\{a\left(\boldsymbol{\mu}^{(\ell)}\right), \boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}\right)$
  - inverse map $m_{\mathbf{A}}^{-1}$ from the tangent space of $\mathcal{M}$ at the point $\mathbf{A}$ to the manifold $\mathcal{M}$

- Input
  - pre-computed matrices $\left\{\mathbf{A}\left(\boldsymbol{\mu}^{(\ell)}\right) \in \mathbb{R}^{N\times M}\right\}_{\ell=1}^{N_s}$
  - map $m_{\mathbf{A}}$ from the manifold $\mathcal{M}$ to the tangent space of $\mathcal{M}$ at the point $\mathbf{A}$
  - multi-variate interpolator $\mathcal{I}$ in $\mathbb{R}^p$
    $a(\boldsymbol{\mu}) = \mathcal{I}\left(\boldsymbol{\mu}; \left\{a\left(\boldsymbol{\mu}^{(\ell)}\right), \boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}\right)$
  - inverse map $m_{\mathbf{A}}^{-1}$ from the tangent space of $\mathcal{M}$ at the point $\mathbf{A}$ to the manifold $\mathcal{M}$
- Requirement: The interpolation operator $\mathcal{I}$ must preserve the tangent space $\Rightarrow$ linear operator – for example,

$$a(\boldsymbol{\mu}^\star) = \mathcal{I}\left(\boldsymbol{\mu}^\star; \left\{a\left(\boldsymbol{\mu}^{(\ell)}\right), \boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}\right) = \sum_{\ell=1}^{N_s} \theta_\ell(\boldsymbol{\mu}^\star) a\left(\boldsymbol{\mu}^{(\ell)}\right)$$

- Algorithm

**for** $\ell = 1 : N_s$ **do**
    compute $\mathbf{\Gamma}\left(\boldsymbol{\mu}^{(\ell)}\right) = m_{\mathbf{A}}\left(\mathbf{A}\left(\boldsymbol{\mu}^{(\ell)}\right)\right)$
**end for**
**for** $i = 1 : N$ **do**
    **for** $j = 1 : M$ **do**
        compute $\Gamma_{ij}(\boldsymbol{\mu}^\star) = \mathcal{I}\left(\boldsymbol{\mu}^\star; \left\{\Gamma_{ij}\left(\boldsymbol{\mu}^{(\ell)}\right), \boldsymbol{\mu}^{(\ell)}\right\}_{\ell=1}^{N_s}\right)$
    **end for**
**end for**
form $\mathbf{\Gamma}(\boldsymbol{\mu}^\star) = [\Gamma_{ij}(\boldsymbol{\mu}^\star)]$ and compute $\mathbf{A}(\boldsymbol{\mu}^\star) = m_{\mathbf{A}}^{-1}(\mathbf{\Gamma}(\boldsymbol{\mu}^\star))$

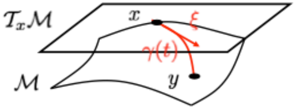
- How does one find $m_{\mathbf{A}}$ and its inverse $m_{\mathbf{A}}^{-1}$?

- How does one find $m_{\mathbf{A}}$ and its inverse $m_{\mathbf{A}}^{-1}$?
- Idea: Use concepts from differential geometry

- How does one find $m_{\mathbf{A}}$ and its inverse $m_{\mathbf{A}}^{-1}$?
- Idea: Use concepts from differential geometry
- Geodesic
  - is a generalization of a "straight line" to "curved spaces" (manifolds)
  - is uniquely defined given a point $x$ on the manifold and a tangent vector $\xi$ at this point

$$\gamma(t; x, \xi) : [0, 1] \to \mathcal{M}$$

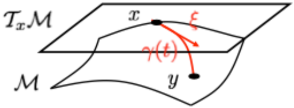
$$\gamma(0; x, \xi) = x, \quad \dot{\gamma}(0, x, \xi) = \xi$$



- How does one find $m_{\mathbf{A}}$ and its inverse $m_{\mathbf{A}}^{-1}$?
- Idea: Use concepts from differential geometry
- Geodesic
  - is a generalization of a "straight line" to "curved spaces" (manifolds)
  - is uniquely defined given a point $x$ on the manifold and a tangent vector $\xi$ at this point

$$\gamma(t; x, \xi) : [0, 1] \to \mathcal{M}$$

$$\gamma(0; x, \xi) = x, \quad \dot{\gamma}(0, x, \xi) = \xi$$



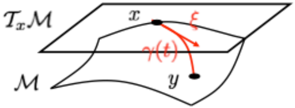

- Exponential map

$$\text{Exp}_x : \mathcal{T}_x\mathcal{M} \to \mathcal{M} \quad \xi \longmapsto \gamma(1; x, \xi)$$

- How does one find $m_{\mathbf{A}}$ and its inverse $m_{\mathbf{A}}^{-1}$?
- Idea: Use concepts from differential geometry
- Geodesic
  - is a generalization of a "straight line" to "curved spaces" (manifolds)
  - is uniquely defined given a point $x$ on the manifold and a tangent vector $\xi$ at this point

$$\gamma(t; x, \xi) : [0, 1] \to \mathcal{M}$$

$$\gamma(0; x, \xi) = x, \quad \dot{\gamma}(0, x, \xi) = \xi$$



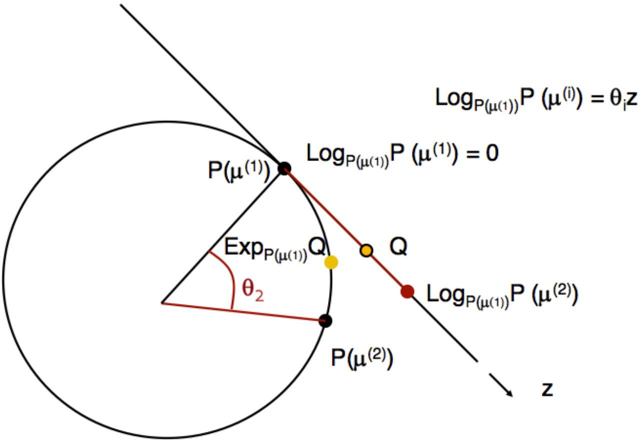

- Exponential map

$$\text{Exp}_x : \mathcal{T}_x\mathcal{M} \to \mathcal{M} \quad \xi \longmapsto \gamma(1; x, \xi)$$

- Logarithm map at base point $x$ (defined in neighborhood $\mathcal{U}_x$ of $x$)

$$\text{Log}_x : \mathcal{U}_x \subset \mathcal{M} \to \mathcal{T}_x\mathcal{M} \quad y \longmapsto \text{Exp}_x^{-1}(y) = \text{Log}_x(y) = \dot{\gamma}(0, x, \xi) = \xi$$

- Application to the interpolation of two points on a circle



- Logarithm map
  1. compute a thin SVD

$$(\mathbf{I} - \mathbf{V}_0\mathbf{V}_0^T)\mathbf{V}_i(\mathbf{V}_0^T\mathbf{V}_i)^{-1} = \mathbf{U}\mathbf{\Sigma}\mathbf{Z}^T$$

  2. compute

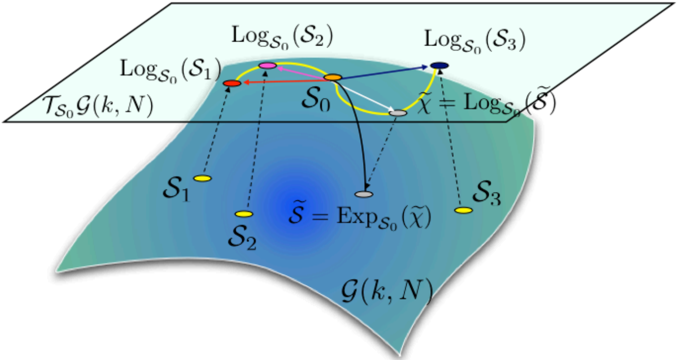
$$\mathbf{\Gamma} = \mathbf{U}\tan^{-1}(\mathbf{\Sigma})\mathbf{Z}^T \in \mathbb{R}^{N\times k}$$

  3. $\mathbf{\Gamma} \leftrightarrow \text{Log}_{\mathcal{S}_0}(\mathcal{S}_i) \in \mathcal{T}_{\mathcal{S}_0}\mathcal{G}(k, N)$

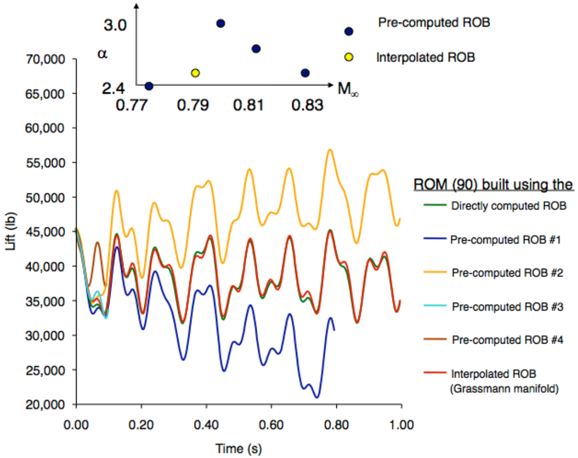
- Logarithm map
  1. compute a thin SVD
  $$(\mathbf{I} - \mathbf{V}_0\mathbf{V}_0^T)\mathbf{V}_i(\mathbf{V}_0^T\mathbf{V}_i)^{-1} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Z}^T$$
  2. compute
  $$\boldsymbol{\Gamma} = \mathbf{U}\tan^{-1}(\boldsymbol{\Sigma})\mathbf{Z}^T \in \mathbb{R}^{N\times k}$$
  3. $\boldsymbol{\Gamma} \leftrightarrow \text{Log}_{\mathcal{S}_0}(\mathcal{S}_i) \in \mathcal{T}_{\mathcal{S}_0}\mathcal{G}(k, N)$
- Exponential map of $\tilde{\xi} \in \mathcal{T}_{\mathcal{S}_0}\mathcal{G}(k, N) \leftrightarrow \widetilde{\boldsymbol{\Gamma}}$
  1. compute a thin SVD
  $$\widetilde{\boldsymbol{\Gamma}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Z}^T$$
  2. compute
  $$\mathbf{V} = (\mathbf{V}_0\mathbf{Z}\cos\boldsymbol{\Sigma} + \mathbf{U}\sin\boldsymbol{\Sigma}) \in \mathcal{ST}(k, N)$$
  3. $\text{range}(\mathbf{V}) = \text{Exp}_{\mathcal{S}_0}(\tilde{\xi}) \in \mathcal{G}(k, N)$

- Logarithm map
  1. compute a thin SVD
  $$(\mathbf{I} - \mathbf{V}_0\mathbf{V}_0^T)\mathbf{V}_i(\mathbf{V}_0^T\mathbf{V}_i)^{-1} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Z}^T$$
  2. compute
  $$\boldsymbol{\Gamma} = \mathbf{U}\tan^{-1}(\boldsymbol{\Sigma})\mathbf{Z}^T \in \mathbb{R}^{N\times k}$$
  3. $\boldsymbol{\Gamma} \leftrightarrow \text{Log}_{\mathcal{S}_0}(\mathcal{S}_i) \in \mathcal{T}_{\mathcal{S}_0}\mathcal{G}(k,N)$
- Exponential map of $\tilde{\xi} \in \mathcal{T}_{\mathcal{S}_0}\mathcal{G}(k,N) \leftrightarrow \widetilde{\boldsymbol{\Gamma}}$
  1. compute a thin SVD
  $$\widetilde{\boldsymbol{\Gamma}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Z}^T$$
  2. compute
  $$\mathbf{V} = (\mathbf{V}_0\mathbf{Z}\cos\boldsymbol{\Sigma} + \mathbf{U}\sin\boldsymbol{\Sigma}) \in \mathcal{ST}(k,N)$$
  3. $\text{range}(\mathbf{V}) = \text{Exp}_{\mathcal{S}_0}(\tilde{\xi}) \in \mathcal{G}(k,N)$
- Note: The trigonometric operators apply only to the diagonal entries of the relevant matrices

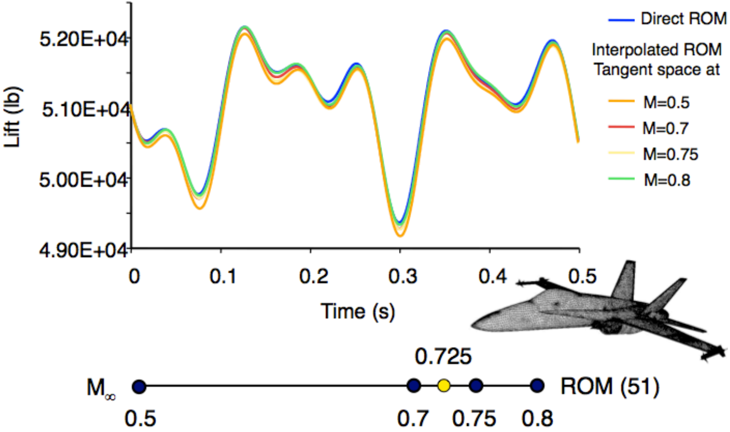
- Interpolation on the tangent space of $\mathcal{G}(k, N)$



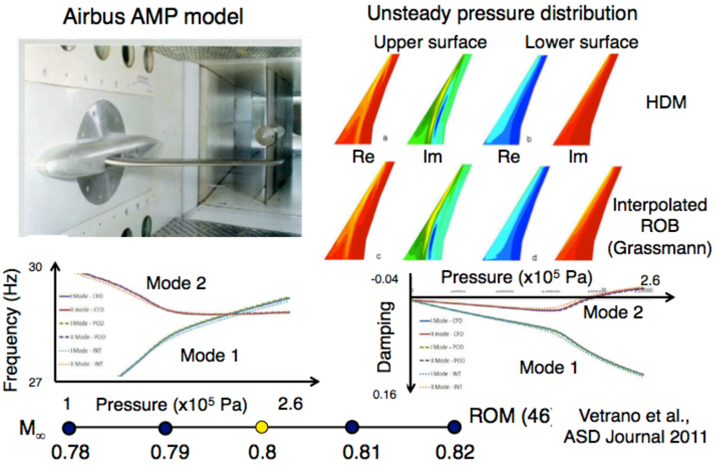
- Parametric, linearized, aeroelastic identification of a F-16 Block 40 aircraft in clean wing configuration (e.g., for flutter analysis in the time domain)



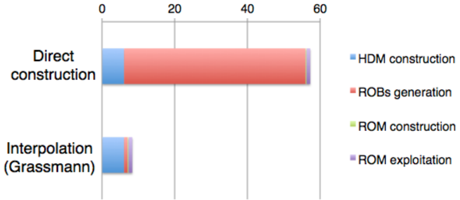
- Parametric, linearized, aeroelastic identification of a F/A-18 aircraft in clean wing configuration (e.g., for flutter analysis in the time domain): Effect of the choice of the tangent plane



- Prediction of the linearized, aeroelastic behavior of the wing of a commercial aircraft (Airbus)

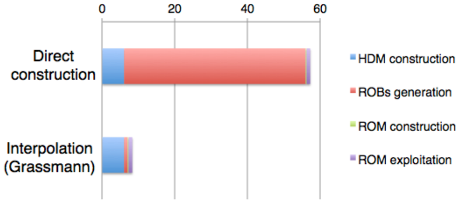


- Parametric, linearized, aeroelastic identification of a F-16 Block 40 aircraft in clean wing configuration (e.g., for flutter analysis in the time domain): Construction and exploitation in $t \in [0, 1]$ $s$ of a linearized, aeroelastic PROM




- Overall CPU time is decreased from 55 minutes to 8 minutes

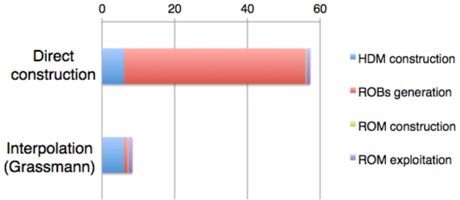
- Parametric, linearized, aeroelastic identification of a F-16 Block 40 aircraft in clean wing configuration (e.g., for flutter analysis in the time domain): Construction and exploitation in $t \in [0, 1]$ $s$ of a linearized, aeroelastic PROM




- Overall CPU time is decreased from 55 minutes to 8 minutes
- New dominant cost: Construction of the HDM operators

$$(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$$

- Parametric, linearized, aeroelastic identification of a F-16 Block 40 aircraft in clean wing configuration (e.g., for flutter analysis in the time domain): Construction and exploitation in $t \in [0, 1]$ s of a linearized, aeroelastic PROM

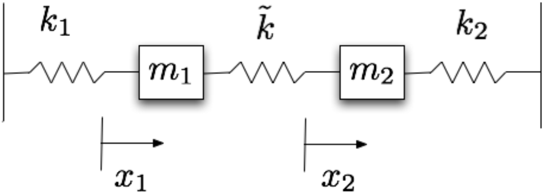



- Overall CPU time is decreased from 55 minutes to 8 minutes
- New dominant cost: Construction of the HDM operators
$$(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$$
- This suggests the following alternative approach: Interpolate the reduced-order operators $\left(\mathbf{A}_r\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{B}_r\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{C}_r\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{D}_r\left(\boldsymbol{\mu}^{(\ell)}\right)\right)$ since they are linear in this application $\Rightarrow$ concept of a database of local *linear* PROMs

- Applicable only to linear systems characterized by operators such as $(\mathbf{A}_r(\boldsymbol{\mu}), \mathbf{B}_r(\boldsymbol{\mu}), \mathbf{C}_r(\boldsymbol{\mu}), \mathbf{D}_r(\boldsymbol{\mu}))$ that are pre-computed and stored in a database of local PROMs
  - for each individual set of local operators – e.g., $\left\{\mathbf{A}_r\left(\boldsymbol{\mu}^\ell\right)\right\}_{\ell=1}^{N_s}$ – identify the appropriate matrix manifold $\mathcal{M}$ and interpolate the aforementioned set of local operators on $\mathcal{M}$

- Simple example: Mass-spring system with two degrees of freedom

$$\mathbf{M}\frac{d^2\mathbf{w}}{dt^2}(t) + \mathbf{K}(\mu)\mathbf{w}(t) = \mathbf{B}\mathbf{u}(t), \quad \boxed{\mu = k_1 - 0.1}$$




$$\mathbf{w}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

- the values of $m_1$, $m_2$, $\tilde{k}$, and $k_2$ are fixed to some constants
- the value of $k_1$ is set to $k_1 = 0.1 + \mu$, and $\mu$ is treated as a parameter

- PMOR by modal truncation: $\mathbf{V}(\boldsymbol{\mu})$ is the matrix of the two eigenmodes of the structural system

$$\mathbf{K}(\boldsymbol{\mu})\mathbf{v}_j(\boldsymbol{\mu}) = \lambda_j(\boldsymbol{\mu})\mathbf{M}\mathbf{v}_j(\boldsymbol{\mu})$$

- PMOR by modal truncation: $\mathbf{V}(\boldsymbol{\mu})$ is the matrix of the two eigenmodes of the structural system

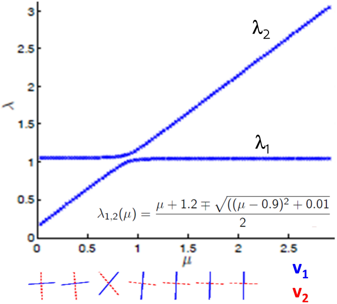
$$\mathbf{K}(\boldsymbol{\mu})\mathbf{v}_j(\boldsymbol{\mu}) = \lambda_j(\boldsymbol{\mu})\mathbf{M}\mathbf{v}_j(\boldsymbol{\mu})$$

- Matrix of eigenvalues: $\mathbf{K}_r(\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{K}(\boldsymbol{\mu})\mathbf{V}(\boldsymbol{\mu}) = \boldsymbol{\Lambda}(\boldsymbol{\mu})$

- PMOR by modal truncation: $\mathbf{V}(\boldsymbol{\mu})$ is the matrix of the two eigenmodes of the structural system

$$\mathbf{K}(\boldsymbol{\mu})\mathbf{v}_j(\boldsymbol{\mu}) = \lambda_j(\boldsymbol{\mu})\mathbf{M}\mathbf{v}_j(\boldsymbol{\mu})$$

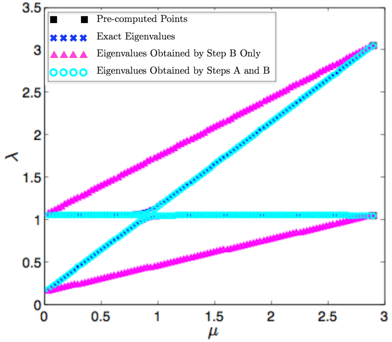
- Matrix of eigenvalues: $\mathbf{K}_r(\boldsymbol{\mu}) = \mathbf{V}(\boldsymbol{\mu})^T\mathbf{K}(\boldsymbol{\mu})\mathbf{V}(\boldsymbol{\mu}) = \boldsymbol{\Lambda}(\boldsymbol{\mu})$
- Variations of the eigenvalues and eigenmodes with the parameter $\boldsymbol{\mu}$ (first eigenmode is shown in blue color, second is shown in red color)



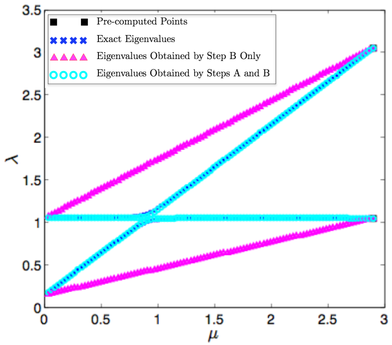
- Note that $\mathbf{\Lambda}(\boldsymbol{\mu})$ belongs to the manifold of (diagonal) symmetric positive definite matrices, SPD(2)

- Note that $\mathbf{\Lambda}(\mu)$ belongs to the manifold of (diagonal) symmetric positive definite matrices, SPD(2)
- Perform interpolation of $\mathbf{\Lambda}(\mu)$ on this manifold using $(\mathbf{\Lambda}(0), \mathbf{\Lambda}(2.9))$

- Note that $\mathbf{\Lambda}(\mu)$ belongs to the manifold of (diagonal) symmetric positive definite matrices, SPD(2)
- Perform interpolation of $\mathbf{\Lambda}(\mu)$ on this manifold using $(\mathbf{\Lambda}(0), \mathbf{\Lambda}(2.9))$
- Result is shown in magenta color

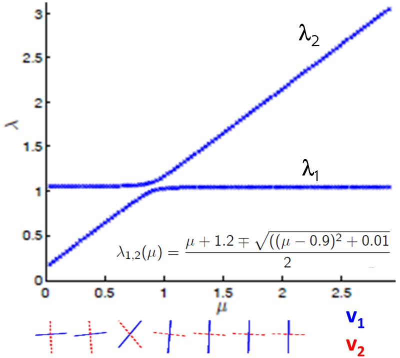


- Note that $\mathbf{\Lambda}(\mu)$ belongs to the manifold of (diagonal) symmetric positive definite matrices, SPD(2)
- Perform interpolation of $\mathbf{\Lambda}(\mu)$ on this manifold using $(\mathbf{\Lambda}(0), \mathbf{\Lambda}(2.9))$
- Result is shown in magenta color

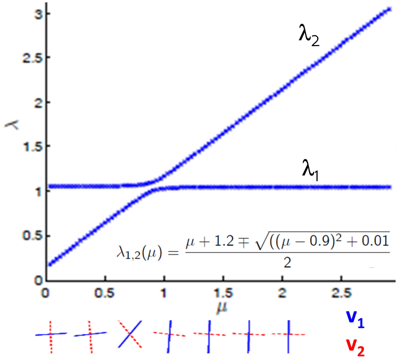



- Observe that the result is wrong, even for such a simple system

- The issue is the lack of consistency between the coordinates of the reduced-order matrices, triggered in this case by **mode veering**



- The issue is the lack of consistency between the coordinates of the reduced-order matrices, triggered in this case by **mode veering**




- Mode crossing would trigger a similar issue (the eigenfrequencies are ordered increasingly in $\mathbf{\Lambda}(\mu)$)

## Two-step solution

- step A: Pre-process the reduced-order matrices
  - enforce consistency by solving the following $N_s$ **orthogonal Procrustes problems**

$$\min_{\mathbf{Q}_\ell / \mathbf{Q}_\ell^T \mathbf{Q}_\ell = \mathbf{I}_k} \left\| \mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right) \mathbf{Q}_\ell - \mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right) \right\|_F, \; \forall \, \ell = 1, \cdots, N_s$$

## Two-step solution

- step A: Pre-process the reduced-order matrices
  - enforce consistency by solving the following $N_s$ **orthogonal Procrustes problems**

$$\min_{\mathbf{Q}_\ell / \mathbf{Q}_\ell^T \mathbf{Q}_\ell = \mathbf{I}_k} \left\| \mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right) \mathbf{Q}_\ell - \mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right) \right\|_F, \quad \forall \ell = 1, \cdots, N_s$$

  - compute analytical solutions of above problems as follows
    1. compute $\mathbf{P}_{\ell,\ell_0} = \mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)^T \mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$
    2. compute the SVD $\mathbf{P}_{\ell,\ell_0} = \mathbf{U}_{\ell,\ell_0} \boldsymbol{\Sigma}_{\ell,\ell_0} \mathbf{Z}_{l,l_0}^T$
    3. compute $\mathbf{Q}_\ell = \mathbf{U}_{\ell,\ell_0} \mathbf{Z}_{\ell,\ell_0}^T$

## Two-step solution

- step A: Pre-process the reduced-order matrices
  - enforce consistency by solving the following $N_s$ **orthogonal Procrustes problems**

$$\min_{\mathbf{Q}_\ell / \mathbf{Q}_\ell^T \mathbf{Q}_\ell = \mathbf{I}_k} \left\| \mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right) \mathbf{Q}_\ell - \mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right) \right\|_F, \quad \forall \ell = 1, \cdots, N_s$$

  - compute analytical solutions of above problems as follows
    1. compute $\mathbf{P}_{\ell,\ell_0} = \mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)^T \mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$
    2. compute the SVD $\mathbf{P}_{\ell,\ell_0} = \mathbf{U}_{\ell,\ell_0} \boldsymbol{\Sigma}_{\ell,\ell_0} \mathbf{Z}_{l,l_0}^T$
    3. compute $\mathbf{Q}_\ell = \mathbf{U}_{\ell,\ell_0} \mathbf{Z}_{\ell,\ell_0}^T$
  - the associated computational cost scales with $k$

$\Longrightarrow$ step A can be performed either online or offline

## Two-step solution (continue)

- step B: Note that (assuming a Galerkin PROM and orthogonal local ROBs)

$$\left(\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell\right)^T\mathbf{A}\left(\boldsymbol{\mu}^{(\ell)}\right)\left(\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell\right) = \mathbf{Q}_\ell^T\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)^T\mathbf{A}\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell = \mathbf{Q}_\ell^T\mathbf{A}_r\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell$$

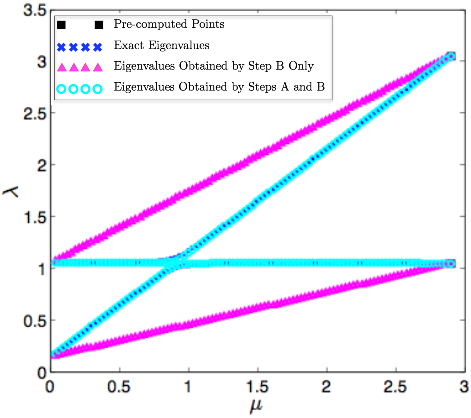
$$\left(\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell\right)^T\mathbf{B}\left(\boldsymbol{\mu}^{(\ell)}\right) = \mathbf{Q}_\ell^T\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)^T\mathbf{B}\left(\boldsymbol{\mu}^{(\ell)}\right) = \mathbf{Q}_\ell^T\mathbf{B}_r\left(\boldsymbol{\mu}^{(\ell)}\right)$$

$$\mathbf{C}\left(\boldsymbol{\mu}^{(\ell)}\right)\left(\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell\right) = \left(\mathbf{C}\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)\right)\mathbf{Q}_\ell = \mathbf{C}_r\mathbf{Q}_\ell$$

and therefore

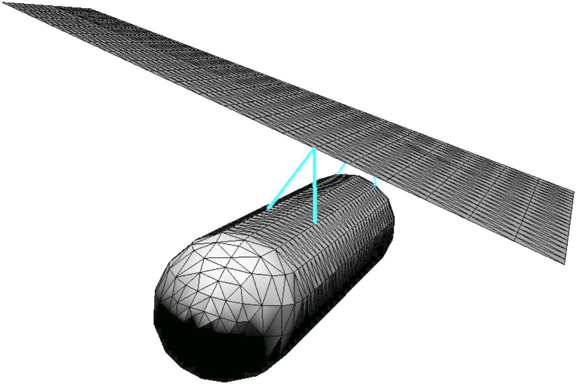
- first, transform *directly* each PROM $\left(\mathbf{A}_r\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{B}_r\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{C}_r\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{D}_r\left(\boldsymbol{\mu}^{(\ell)}\right)\right)$ to $\left(\mathbf{Q}_\ell^T\mathbf{A}_r\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell, \mathbf{Q}_\ell^T\mathbf{B}_r\left(\boldsymbol{\mu}^{(\ell)}\right), \mathbf{C}_r\left(\boldsymbol{\mu}^{(\ell)}\right)\mathbf{Q}_\ell, \mathbf{D}_r\left(\boldsymbol{\mu}^{(\ell)}\right)\right)$
- then, identify for each element of the transformed tuple an appropriate matrix manifold and perform the interpolation on this matrix manifold

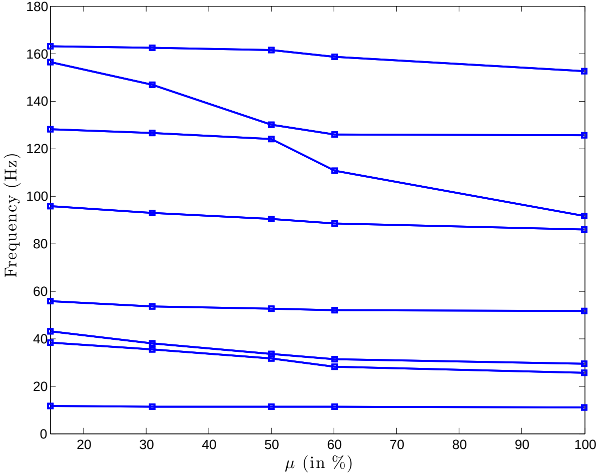
- Two-step result is shown in cyan color

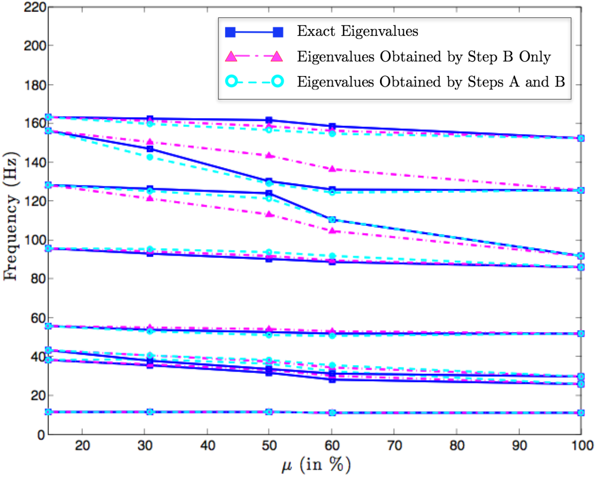



- Observe that the result is very accurate

- More challenging example: Wing with tank and sloshing effects
- The hydro-elastic effects affect the eigenfrequencies and eigenmodes of the structure
- The parameter $\mu$ defines the fuel fill level in the tank $0 \leq \mu \leq 100\%$




Frequency (Hz)
$\mu$ (in %)
180
160
140
120
100
80
60
40
20
0
20
30
40
50
60
70
80
90
100


Exact Eigenvalues
Eigenvalues Obtained by Step B Only
Eigenvalues Obtained by Steps A and B
Frequency (Hz)
μ (in %)
0
20
40
60
80
100
120
140
160
180
200
220
20
30
40
50
60
70
80
90
100

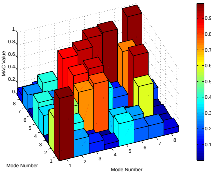
- Modal Assurance Criterion (MAC) between two modes $\phi$ and $\psi$

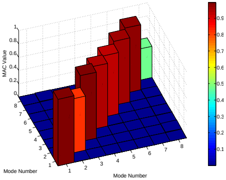
$$\text{MAC}(\phi, \psi) = \frac{|\phi^T \psi|^2}{(\phi^T \phi)(\psi^T \psi)}$$

- Modal Assurance Criterion (MAC) between two modes $\phi$ and $\psi$

$$\text{MAC}(\phi, \psi) = \frac{|\phi^T \psi|^2}{(\phi^T \phi)(\psi^T \psi)}$$

- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$ before and after Step A?

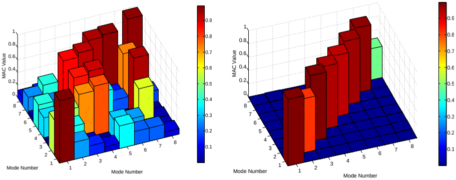


- Modal Assurance Criterion (MAC) between two modes $\phi$ and $\psi$

$$\text{MAC}(\phi, \psi) = \frac{|\phi^T \psi|^2}{(\phi^T \phi)(\psi^T \psi)}$$

- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$ before and after Step A?

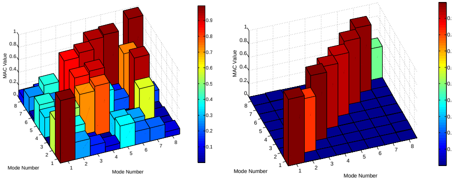



- When $\phi$ and $\psi$ are normalized, $\text{MAC}(\phi, \psi) = |\phi^T \psi|^2$

- Modal Assurance Criterion (MAC) between two modes $\phi$ and $\psi$

$$\text{MAC}(\phi, \psi) = \frac{|\phi^T \psi|^2}{(\phi^T \phi)(\psi^T \psi)}$$

- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$ before and after Step A?

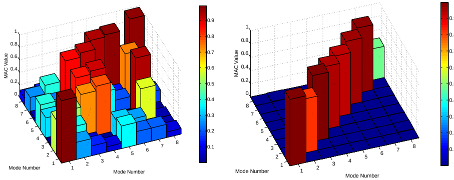



- When $\phi$ and $\psi$ are normalized, $\text{MAC}(\phi, \psi) = |\phi^T \psi|^2$
- $\mathbf{P}_{\ell,\ell_0} = \mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)^T \mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$ is the matrix of square roots of the MACs between the modes contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)$ and those contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$

- Modal Assurance Criterion (MAC) between two modes $\phi$ and $\psi$

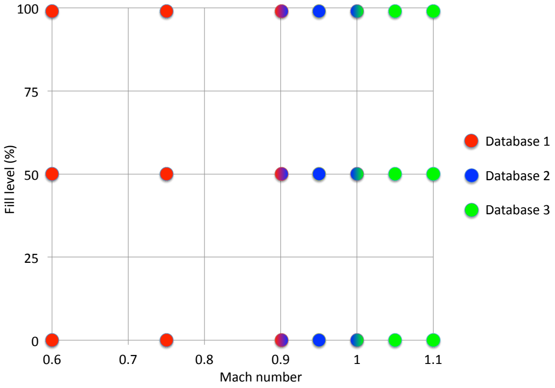
$$\text{MAC}(\phi, \psi) = \frac{|\phi^T \psi|^2}{(\phi^T \phi)(\psi^T \psi)}$$

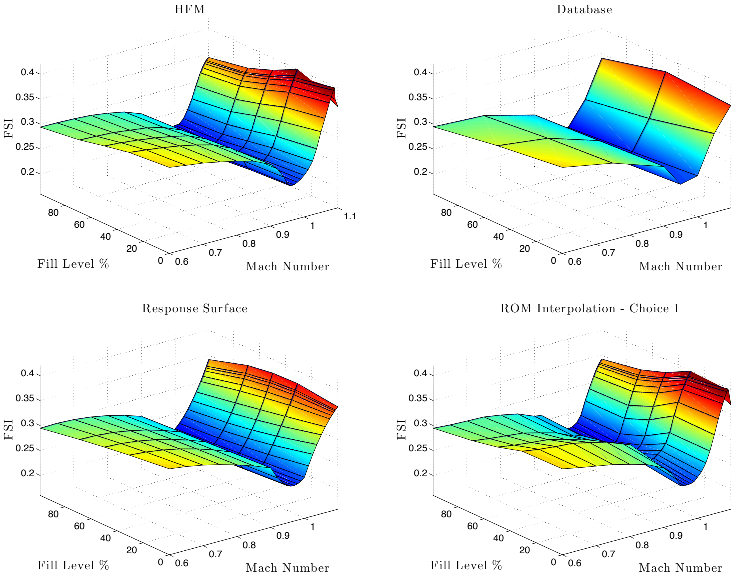
- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$ before and after Step A?



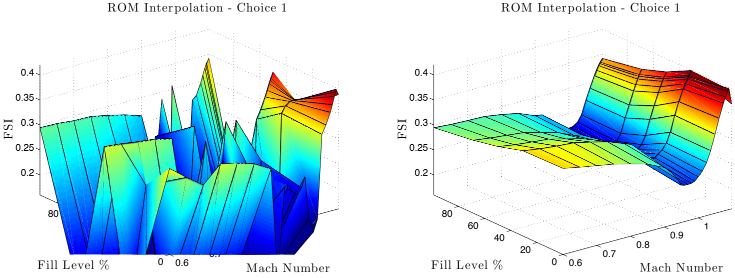

- When $\phi$ and $\psi$ are normalized, $\text{MAC}(\phi, \psi) = |\phi^T \psi|^2$
- $\mathbf{P}_{\ell,\ell_0} = \mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)^T \mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$ is the matrix of square roots of the MACs between the modes contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell)}\right)$ and those contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(\ell_0)}\right)$
- This is the Modal Assurance Criterion Square Root (MACSR)

- Aeroelastic study of a wing-tank system
- 2 parameters, namely, the fuel fill level and the free-stream Mach number $M_\infty$
- Database approach




HFM
Database
Response Surface
ROM Interpolation - Choice 1
FSI
0.4
0.35
0.3
0.25
0.2
Fill Level %
80
60
40
20
0
Mach Number
0.6
0.7
0.8
0.9
1
1.1

- Effect of Step A




- Skipping Step A leads to inaccurate interpolation results (left figure)
- Performing Step A ensures a consistent interpolation (right figure)

- CPU performance

| Approach | Offline phase CPU time (# procs) | Online phase CPU time (# procs) |
|---|---|---|
| HDM | - (-) | 9 152 000 s ≈ 106 days (32) |
| Response surface | 28 000 s ≈ 7 h (32) | 2 s (1) |
| PROM interpolation | 28 000 s ≈ 7 h (32) | 30 s (1) |

- Online speedup factor = 305 000
- Offline + Online speedup factor = 327

- Mobile computing using a database of PROMs