AA216/CME345: MODEL REDUCTION

Local Parametric Approaches

Charbel Farhat
Stanford University
cfarhat@stanford.edu

Outline

1. Concept of a Database of Local PROMs
2. Interpolation of ROBs on Quotient Manifolds (Amsallem and Farhat, 2008)
3. Interpolation of PROMs on Embedded Manifolds (Amsallem and Farhat, 2011)

- Note: The material covered in this chapter is based on the following published papers:
  - 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.

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

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

- Goal: Construct a parametric Projection-based Reduced-Order Model (PROM)

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

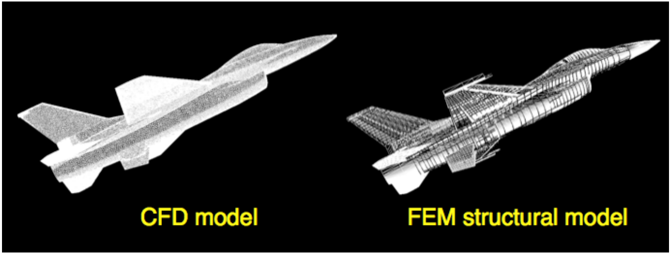
  - based on **local** Reduced-Order Bases (ROBs) $\left(\mathbf{V}(\boldsymbol{\mu}^{(l)}),\ \mathbf{W}(\boldsymbol{\mu}^{(l)})\right)$ and the approximation

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

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

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

- Hundreds of flight conditions $\boldsymbol{\mu} = (M_\infty, \alpha)$ for flutter clearance
- Multiple aircraft configurations



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

- 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 configuration/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 configuration/flight condition $\boldsymbol{\mu}^{(1)}$
  2. use these two bases 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 configuration/flight condition $\boldsymbol{\mu}^{(1)}$
  2. use these two bases to reduce the HDM at $\boldsymbol{\mu}^{(2)}$
  3. avoid reconstructing a ROB every time the configuration/flight condition is varied

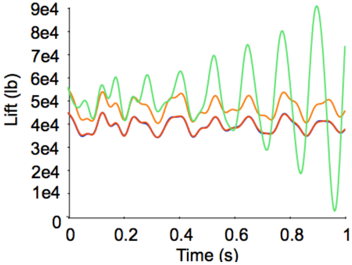
- Consider the following procedure
  1. construct local ROBs $\left(\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right), \mathbf{W}\left(\boldsymbol{\mu}^{(1)}\right)\right)$ at the parametric configuration/flight condition $\boldsymbol{\mu}^{(1)}$
  2. use these two bases to reduce the HDM at $\boldsymbol{\mu}^{(2)}$
  3. avoid reconstructing a ROB every time the configuration/flight condition is varied
  4. build the resulting 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
  - $\boldsymbol{\mu}^{(1)} = (M_{\infty,1}, \alpha^{(1)}) = (0.71, \alpha_{\text{trimmed}}(0.71))$
  - $\boldsymbol{\mu}^{(2)} = (M_{\infty,2}, \alpha^{(2)}) = (0.8, \alpha_{\text{trimmed}}(0.8))$



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

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

- The lack of robustness of the ROBs with respect to parameter changes implies that the ROBs should be **reconstructed** every time the parameters are varied
- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$

- The lack of robustness of the ROBs with respect to parameter changes implies that the ROBs should be **reconstructed** every time the parameters are varied
- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$

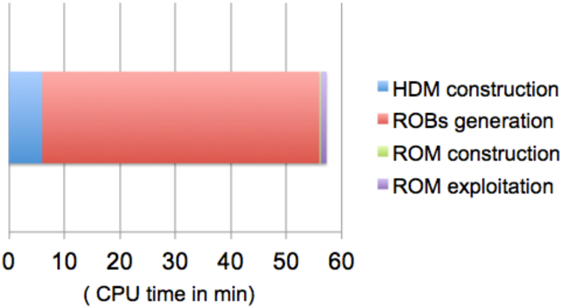
- The lack of robustness of the ROBs with respect to parameter changes implies that the ROBs should be **reconstructed** every time the parameters are varied
- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. generate the ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred projection-based model order reduction method

- The lack of robustness of the ROBs with respect to parameter changes implies that the ROBs should be **reconstructed** every time the parameters are varied
- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. generate the ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred projection-based model order reduction method
  4. construct the PROM operators $(\mathbf{A}_r(\boldsymbol{\mu}^\star), \mathbf{B}_r(\boldsymbol{\mu}^\star), \mathbf{C}_r(\boldsymbol{\mu}^\star), \mathbf{D}_r(\boldsymbol{\mu}^\star))$ using a Petrov-Galerkin projection

- The lack of robustness of the ROBs with respect to parameter changes implies that the ROBs should be **reconstructed** every time the parameters are varied
- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. generate the ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred projection-based model order reduction method
  4. construct the PROM operators $(\mathbf{A}_r(\boldsymbol{\mu}^\star), \mathbf{B}_r(\boldsymbol{\mu}^\star), \mathbf{C}_r(\boldsymbol{\mu}^\star), \mathbf{D}_r(\boldsymbol{\mu}^\star))$ using a Petrov-Galerkin projection
  5. exploit the constructed PROM

- The lack of robustness of the ROBs with respect to parameter changes implies that the ROBs should be **reconstructed** every time the parameters are varied
- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. generate the ROBs $(\mathbf{V}(\boldsymbol{\mu}^\star), \mathbf{W}(\boldsymbol{\mu}^\star))$ using a preferred projection-based model order reduction method
  4. construct the PROM operators $(\mathbf{A}_r(\boldsymbol{\mu}^\star), \mathbf{B}_r(\boldsymbol{\mu}^\star), \mathbf{C}_r(\boldsymbol{\mu}^\star), \mathbf{D}_r(\boldsymbol{\mu}^\star))$ using a Petrov-Galerkin projection
  5. exploit the constructed PROM
- Question: Is this procedure computationally **efficient**?

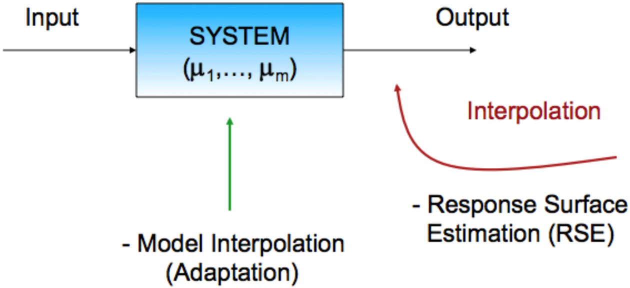
- Construction and exploitation in $t \in [0, 1]s$ of a linearized aeroelastic F-16 PROM



- The direct generation of the ROB 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**

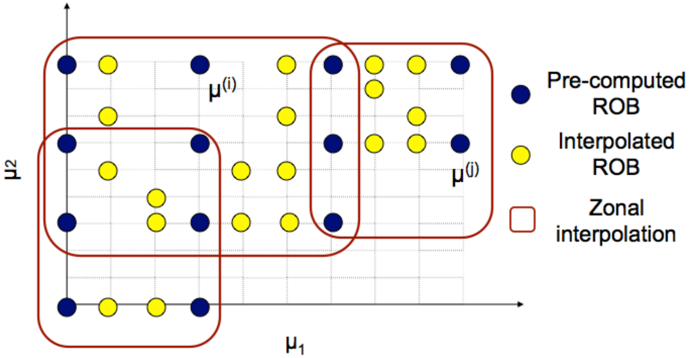
- Idea: Pre-compute some quantities offline
- Interpolate these quantities online

- Idea: Pre-compute some quantities offline
- Interpolate these quantities online
- What entities should be interpolated and how?



$\Rightarrow$ interpolate the ROBs given the above CPU time analysis

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



- For simplicity, assume an orthogonal Galerkin projection

$$\mathbf{V}(\mu) = \mathbf{W}(\mu) \quad \text{and} \quad \mathbf{V}(\mu)^T \mathbf{V}(\mu) = \mathbf{I}_k$$

- Procedure

- For simplicity, assume an orthogonal Galerkin projection

$$\mathbf{V}(\boldsymbol{\mu}) = \mathbf{W}(\boldsymbol{\mu}) \quad \text{and} \quad \mathbf{V}(\boldsymbol{\mu})^T \mathbf{V}(\boldsymbol{\mu}) = \mathbf{I}_k$$

- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$

- For simplicity, assume an orthogonal Galerkin projection

$$\mathbf{V}(\boldsymbol{\mu}) = \mathbf{W}(\boldsymbol{\mu}) \quad \text{and} \quad \mathbf{V}(\boldsymbol{\mu})^T \mathbf{V}(\boldsymbol{\mu}) = \mathbf{I}_k$$

- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$

- For simplicity, assume an orthogonal Galerkin projection

$$\mathbf{V}(\boldsymbol{\mu}) = \mathbf{W}(\boldsymbol{\mu}) \quad \text{and} \quad \mathbf{V}(\boldsymbol{\mu})^T \mathbf{V}(\boldsymbol{\mu}) = \mathbf{I}_k$$

- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. compute the ROB $\mathbf{V}(\boldsymbol{\mu}^\star)$ by interpolating the pre-computed ROBs

- For simplicity, assume an orthogonal Galerkin projection

$$\mathbf{V}(\boldsymbol{\mu}) = \mathbf{W}(\boldsymbol{\mu}) \quad \text{and} \quad \mathbf{V}(\boldsymbol{\mu})^T\mathbf{V}(\boldsymbol{\mu}) = \mathbf{I}_k$$

- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. compute the ROB $\mathbf{V}(\boldsymbol{\mu}^\star)$ by interpolating the pre-computed ROBs
  4. construct the PROM operators $(\mathbf{A}_r(\boldsymbol{\mu}^\star), \mathbf{B}_r(\boldsymbol{\mu}^\star), \mathbf{C}_r(\boldsymbol{\mu}^\star), \mathbf{D}_r(\boldsymbol{\mu}^\star))$ by Galerkin projection

- For simplicity, assume an orthogonal Galerkin projection

$$\mathbf{V}(\boldsymbol{\mu}) = \mathbf{W}(\boldsymbol{\mu}) \quad \text{and} \quad \mathbf{V}(\boldsymbol{\mu})^T\mathbf{V}(\boldsymbol{\mu}) = \mathbf{I}_k$$

- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. compute the ROB $\mathbf{V}(\boldsymbol{\mu}^\star)$ by interpolating the pre-computed ROBs
  4. construct the PROM operators $(\mathbf{A}_r(\boldsymbol{\mu}^\star), \mathbf{B}_r(\boldsymbol{\mu}^\star), \mathbf{C}_r(\boldsymbol{\mu}^\star), \mathbf{D}_r(\boldsymbol{\mu}^\star))$ by Galerkin projection
  5. exploit the constructed PROM

- For simplicity, assume an orthogonal Galerkin projection

$$\mathbf{V}(\boldsymbol{\mu}) = \mathbf{W}(\boldsymbol{\mu}) \quad \text{and} \quad \mathbf{V}(\boldsymbol{\mu})^T\mathbf{V}(\boldsymbol{\mu}) = \mathbf{I}_k$$

- Procedure
  1. given a queried but unsampled parameter point $\boldsymbol{\mu}^\star \in \mathcal{D}$
  2. construct the HDM operators $(\mathbf{A}(\boldsymbol{\mu}^\star), \mathbf{B}(\boldsymbol{\mu}^\star), \mathbf{C}(\boldsymbol{\mu}^\star), \mathbf{D}(\boldsymbol{\mu}^\star))$
  3. compute the ROB $\mathbf{V}(\boldsymbol{\mu}^\star)$ by interpolating the pre-computed ROBs
  4. construct the PROM operators $(\mathbf{A}_r(\boldsymbol{\mu}^\star), \mathbf{B}_r(\boldsymbol{\mu}^\star), \mathbf{C}_r(\boldsymbol{\mu}^\star), \mathbf{D}_r(\boldsymbol{\mu}^\star))$ by Galerkin projection
  5. exploit the constructed PROM
- Question: How does one interpolate pre-computed ROBs?

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

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

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

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

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

- Algorithm

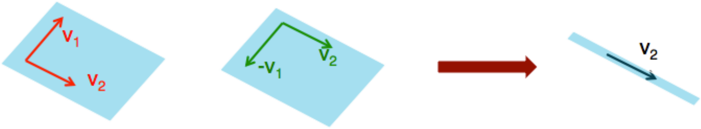
```
for i = 1 : N do
    for j = 1 : k do
        compute v_ij(μ*) = I(μ*; {v_ij(μ^(l)), μ^(l)}_{l=1}^{N_s})
    end for
end for
form V(μ*) = [v_ij(μ*)]
```

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

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

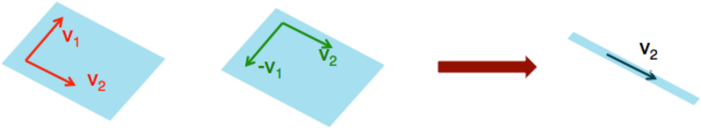
- Example
  - $N = 3,\ k = 2,\ m = 1$
  - for $\boldsymbol{\mu}^{(1)} = 0$: $\mathbf{V}\left(\boldsymbol{\mu}^{(1)}\right) = \mathbf{V}(0) = (\mathbf{v}_1\ \mathbf{v}_2)^T$
  - for $\boldsymbol{\mu}^{(2)} = 1$: $\mathbf{V}\left(\boldsymbol{\mu}^{(2)}\right) = \mathbf{V}(1) = (-\mathbf{v}_1\ \mathbf{v}_2)^T$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

$$\mathbf{V}(0.5) = 0.5\,(\mathbf{V}(0) + \mathbf{V}(1)) = \left(0.5(\mathbf{v}_1 - \mathbf{v}_1)\ 0.5(\mathbf{v}_2 + \mathbf{v}_2)\right)^T = \left(0\ \mathbf{v}_2\right)^T$$



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

$$\mathbf{V}(0.5) = 0.5\,(\mathbf{V}(0) + \mathbf{V}(1)) = \left(0.5(\mathbf{v}_1 - \mathbf{v}_1)\ 0.5(\mathbf{v}_2 + \mathbf{v}_2)\right)^T = \left(0\ \mathbf{v}_2\right)^T$$



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

- Reduced-order equation

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

- Reduced-order equation

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

- Equivalent high-dimensional equation for $\widetilde{\mathbf{w}}(t;\mu) = \mathbf{V}(\mu)\mathbf{q}(t;\mu)$

$$\frac{d\widetilde{\mathbf{w}}}{dt}(t;\mu) = \mathbf{\Pi}_{\mathbf{V}(\mu),\mathbf{V}(\mu)}\mathbf{A}(\mu)\widetilde{\mathbf{w}}(t;\mu) + \mathbf{\Pi}_{\mathbf{V}(\mu),\mathbf{V}(\mu)}\mathbf{B}(\mu)\mathbf{u}(t)$$

- Reduced-order equation

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

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

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

- 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

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

- 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 non-singular 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

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

- 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 non-singular 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 projection-based model order reduction 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 projection-based model order reduction 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 projection-based model order reduction 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 projection-based model order reduction 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 projection-based model order reduction 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 projection-based model order reduction 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 projection-based model order reduction 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 projection-based model order reduction 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

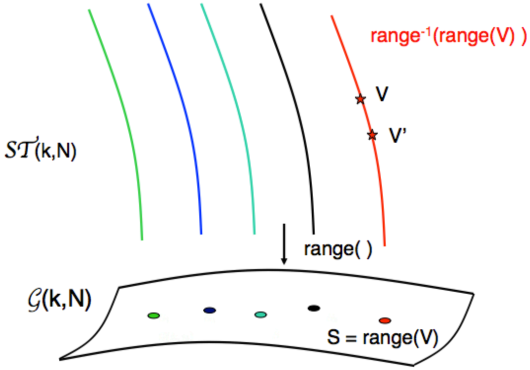
$$\begin{aligned} \forall\ \mathbf{V}_1, \mathbf{V}_2 \in \mathcal{ST}(k, N),\ \mathbf{V}_1 \sim \mathbf{V}_2 \quad &\Leftrightarrow \quad \text{range}(\mathbf{V}_1) = \text{range}(\mathbf{V}_2) \\ &\Leftrightarrow \quad \exists \mathbf{Q} \in \mathcal{O}(k) \text{ s.t } \mathbf{V}_1 = \mathbf{V}_2\mathbf{Q}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

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

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

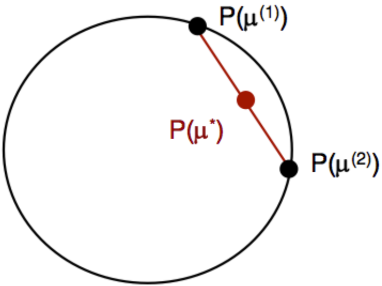
- Hence, one should interpolate subspaces, but has access in practice to (orthogonal) ROBs
- Solution: Perform interpolation on the Grasmann manifold using entities belonging to the (orthogonal) Stiefel manifold

- Matrix 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 $\mathbb{R}^{N \times k}$

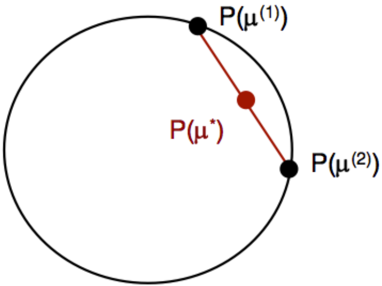


- Embedded matrix manifolds
  - the sphere
$$\mathbb{S}(N) = \left\{\mathbf{w} \in \mathbb{R}^N \text{ s.t. } \|\mathbf{w}\|_2 = 1\right\}$$
  - the manifold of orthogonal matrices
$$\mathcal{O}(N) = \left\{\mathbf{M} \in \mathbb{R}^{N\times N} \text{ s.t. } \mathbf{M}^T\mathbf{M} = \mathbf{I}_N\right\}$$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

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

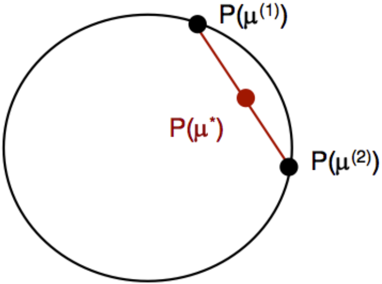


- First example: The circle (sphere $\mathbb{S}(N)$ for $N = 2$)



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

- First 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 to the manifold**

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

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

- Requirement: The interpolation operator $\mathcal{I}$ must preserve the tangent space $\Rightarrow$ for example,

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

- Algorithm

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

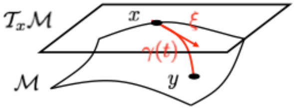
- How does one find $m_{\mathbf{A}}$ and its inverse $m_{\mathbf{A}}^{-1}$?

- 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 χ at this point

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

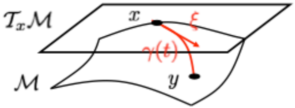
$$\gamma(0; x, \xi) = x, \quad \dot{\gamma}(0, x, \xi) = \xi$$



- 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 χ at this point

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

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



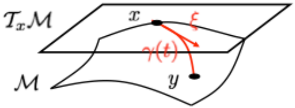
- Exponential map

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

- 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 χ at this point

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

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



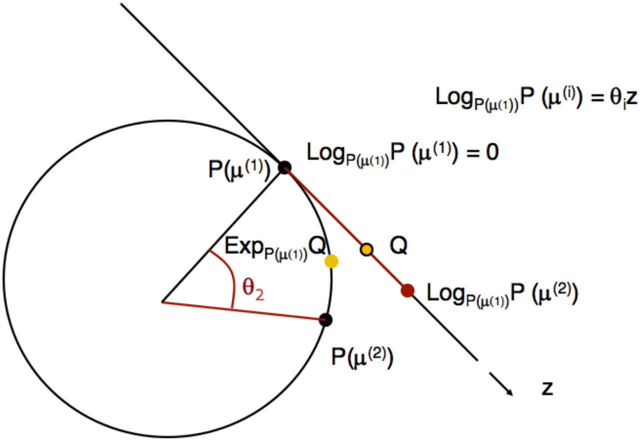
- Exponential map

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

- Logarithmic map (defined in a neighborhood $\mathcal{U}_x$ of x)

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

- Application to the interpolation of points on a circle



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

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

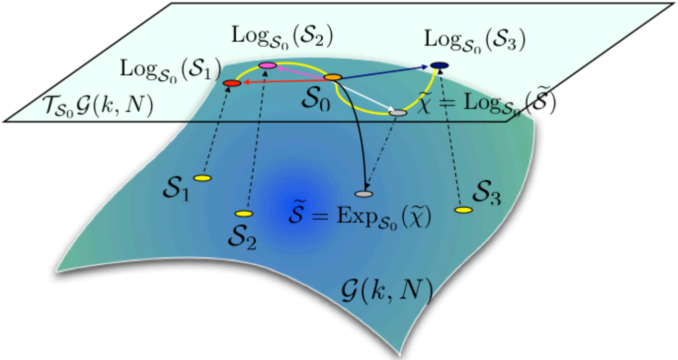
- Exponential map of $\widetilde{\chi} \in \mathcal{T}_{\mathcal{S}_0}\mathcal{G}(k, N) \leftrightarrow \widetilde{\boldsymbol{\Gamma}}$
  1. compute a thin SVD
  $$\widetilde{\boldsymbol{\Gamma}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Z}^T$$
  2. compute
  $$\mathbf{V} = (\mathbf{V}_0\mathbf{Z}\cos\boldsymbol{\Sigma}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

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

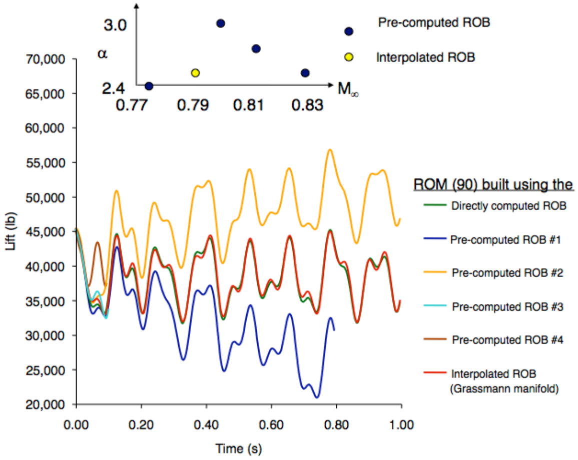
- Exponential map of $\widetilde{\chi} \in \mathcal{T}_{\mathcal{S}_0}\mathcal{G}(k, N) \leftrightarrow \widetilde{\mathbf{\Gamma}}$
  1. compute a thin SVD
$$\widetilde{\mathbf{\Gamma}} = \mathbf{U}\mathbf{\Sigma}\mathbf{Z}^T$$
  2. compute
$$\mathbf{V} = (\mathbf{V}_0\mathbf{Z}\cos\mathbf{\Sigma} + \mathbf{U}\sin\mathbf{\Sigma}) \in \mathcal{ST}(k, N)$$
  3. $\text{range}(\mathbf{V}) = \text{Exp}_{\mathcal{S}_0}(\widetilde{\chi}) \in \mathcal{G}(k, N)$

- Note: The trigonometric operators apply only to the diagonal entries of the relevant matrices

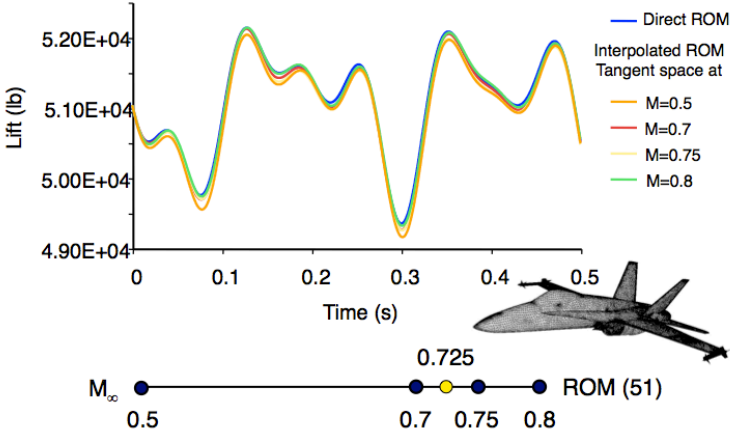
- Interpolation on the tangent space to $\mathcal{G}(k, N)$



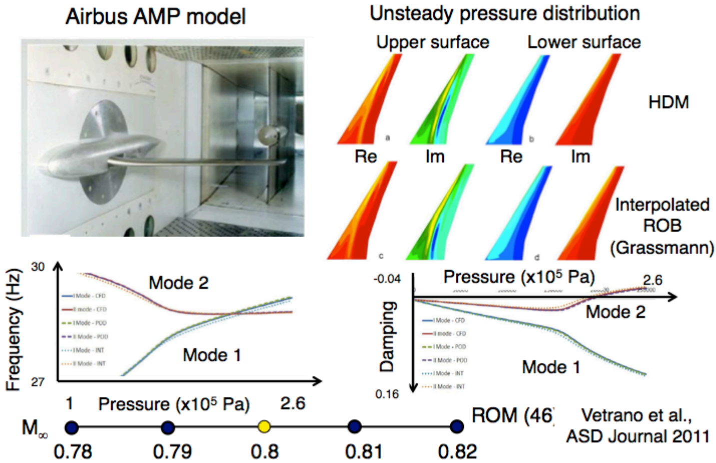
- Prediction of the linearized aeroelastic behavior of an F-16 configuration



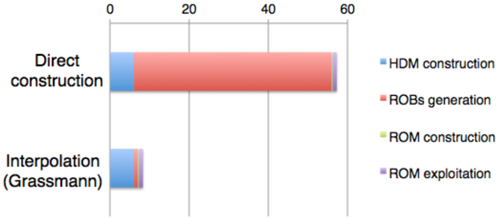
- Prediction of the linearized aeroelastic behavior of an F-18 configuration: Effect of the choice of the tangent plane



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

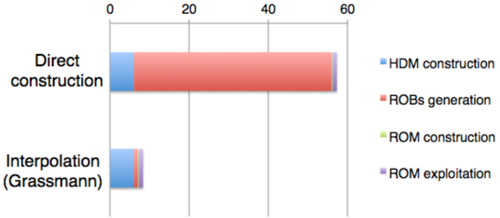


- Construction and exploitation in $t \in [0, 1]$ s of a linearized aeroelastic PROM



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

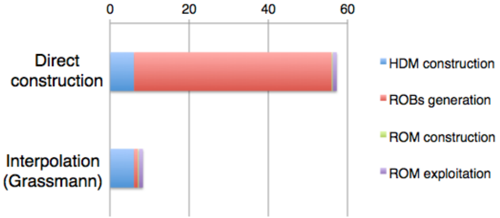
- Construction and exploitation in $t \in [0,1]$ s of a linearized aeroelastic PROM



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

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

- 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

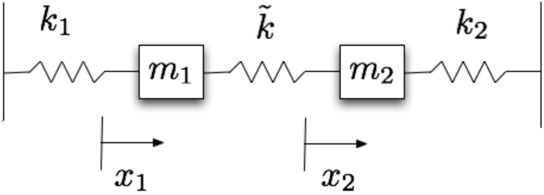
$$(\mathbf{A}(\mu^\star), \mathbf{B}(\mu^\star), \mathbf{C}(\mu^\star), \mathbf{D}(\mu^\star))$$

$\Longrightarrow$ alternative approach is to interpolate the reduced-order operators

$$\left(\mathbf{A}_r(\mu^{(l)}), \mathbf{B}_r(\mu^{(l)}), \mathbf{C}_r(\mu^{(l)}), \mathbf{D}_r(\mu^{(l)})\right)$$

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

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



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

- Projection-based model order reduction by modal truncation: $\mathbf{V}(\boldsymbol{\mu})$ is the matrix of the two eigenmodes of the structural system

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

- Projection-based model order reduction by modal truncation: $\mathbf{V}(\mu)$ is the matrix of the two eigenmodes of the structural system

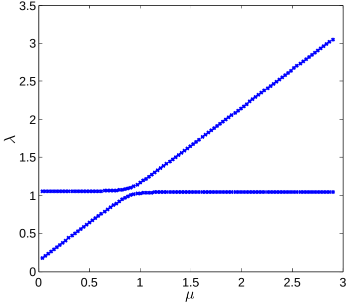
$$\mathbf{K}(\mu)\mathbf{v}_j(\mu) = \lambda_j(\mu)\mathbf{M}\mathbf{v}_j(\mu)$$

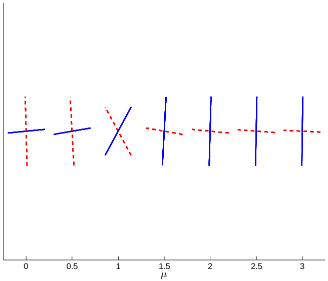
- Matrix of eigenvalues: $\mathbf{K}_r(\mu) = \mathbf{V}(\mu)^T\mathbf{K}(\mu)\mathbf{V}(\mu) = \mathbf{\Lambda}(\mu)$

- Projection-based model order reduction by modal truncation: $\mathbf{V}(\mu)$ is the matrix of the two eigenmodes of the structural system

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

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

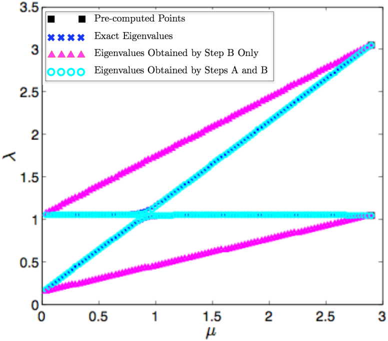




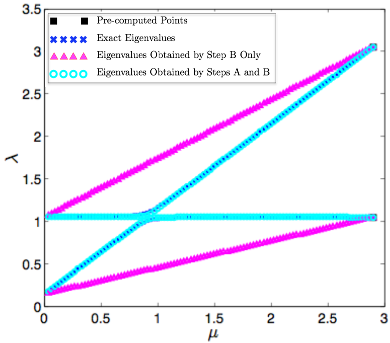
- Note that $\boldsymbol{\Lambda}(\boldsymbol{\mu})$ belongs to the manifold of (diagonal) symmetric positive definite matrices

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

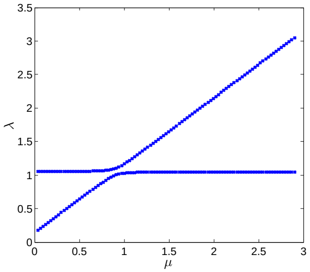


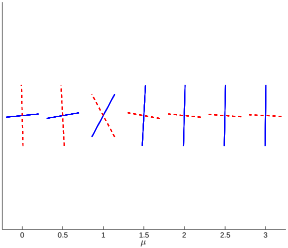
- Note that $\mathbf{\Lambda}(\mu)$ belongs to the manifold of (diagonal) symmetric positive definite matrices
- Perform interpolation of $\mathbf{\Lambda}(\mu)$ on this manifold using $(\mathbf{\Lambda}(0), \mathbf{\Lambda}(2.9))$
- Result is shown in magenta color



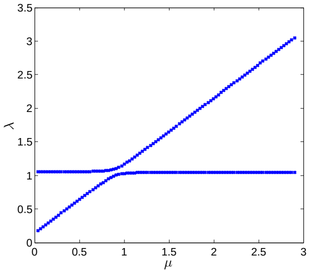
- Observe that the result is wrong, even for such a simple system

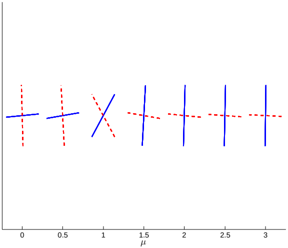
- The issue is the lack of consistency between the coordinates of the reduced-order matrices, triggered in this case by **mode veering**





- 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**

$$\min_{\mathbf{Q}_l \ \mathbf{Q}_l^T \mathbf{Q}_l = \mathbf{I}_k} \left\| \mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right) \mathbf{Q}_l - \mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right) \right\|_F, \ \forall l = 1, \cdots, N_s$$

Two-step solution

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

$$\min_{\mathbf{Q}_l \;\; \mathbf{Q}_l^T\mathbf{Q}_l=\mathbf{I}_k} \left\| \mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)\mathbf{Q}_l - \mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right) \right\|_F, \quad \forall l = 1, \cdots, N_s$$

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

Two-step solution

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

$$\min_{\mathbf{Q}_l \; \mathbf{Q}_l^T \mathbf{Q}_l = \mathbf{I}_k} \left\| \mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right) \mathbf{Q}_l - \mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right) \right\|_F, \quad \forall l = 1, \cdots, N_s$$

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

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

Two-step solution

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

$$\min_{\mathbf{Q}_l \ \mathbf{Q}_l^T \mathbf{Q}_l = \mathbf{I}_k} \left\| \mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right) \mathbf{Q}_l - \mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right) \right\|_F, \quad \forall l = 1, \cdots, N_s$$

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

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

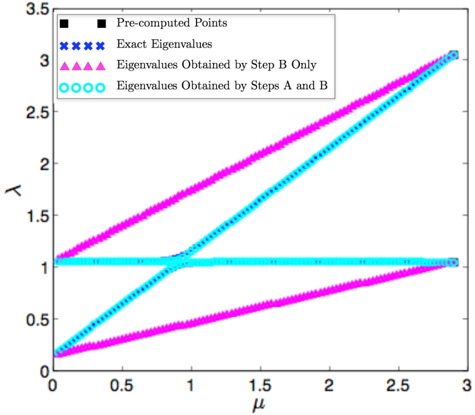
$$\left(\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)\mathbf{Q}_l\right)^T\mathbf{B}\left(\boldsymbol{\mu}^{(l)}\right) = \mathbf{Q}_l^T\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)^T\mathbf{B}\left(\boldsymbol{\mu}^{(l)}\right) = \mathbf{Q}_l^T\mathbf{B}_r\left(\boldsymbol{\mu}^{(l)}\right)$$

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

and therefore

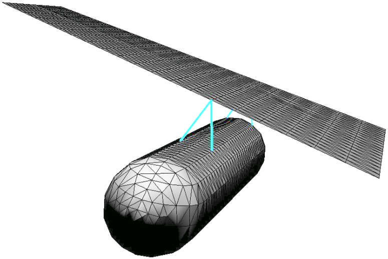
- first, transform *directly* each PROM $\left(\mathbf{A}_r\left(\boldsymbol{\mu}^{(l)}\right), \mathbf{B}_r\left(\boldsymbol{\mu}^{(l)}\right), \mathbf{C}_r\left(\boldsymbol{\mu}^{(l)}\right), \mathbf{D}_r\left(\boldsymbol{\mu}^{(l)}\right)\right)$ to $\left(\mathbf{Q}_l^T\mathbf{A}_r\left(\boldsymbol{\mu}^{(l)}\right)\mathbf{Q}_l,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

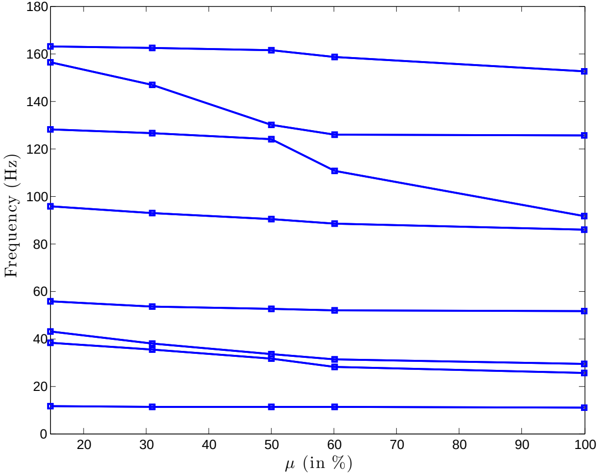
- Result is shown in cyan color

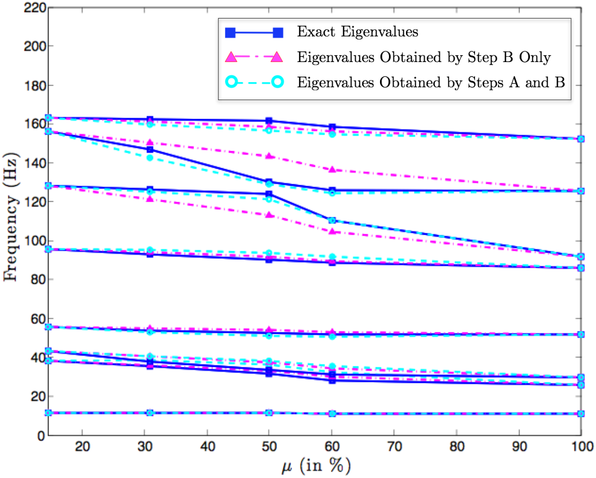


- Observe that the result is very accurate

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




Frequency (Hz)
μ (in %)
0
20
40
60
80
100
120
140
160
180
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

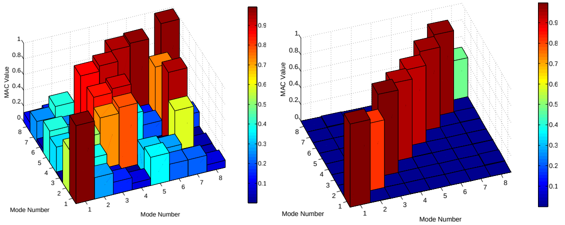
- Modal Assurance Criterion (MAC) between two modes ϕ and ψ

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

- Modal Assurance Criterion (MAC) between two modes ϕ and ψ

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

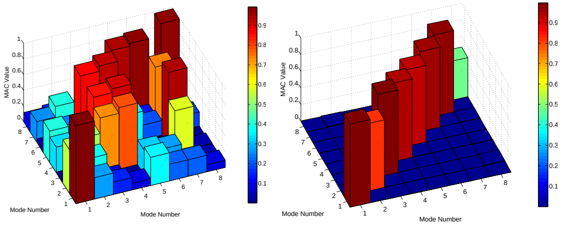
- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right)$ before and after Step A?



- Modal Assurance Criterion (MAC) between two modes ϕ and ψ

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

- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right)$ before and after Step A?

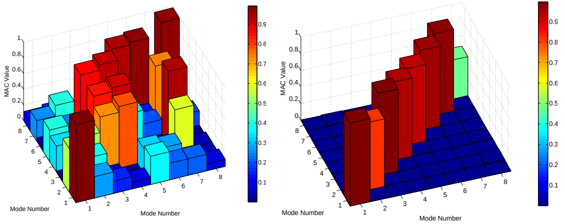


- When ϕ and ψ are normalized, $\text{MAC}(\phi, \psi) = |\phi^T \psi|^2$

- Modal Assurance Criterion (MAC) between two modes ϕ and ψ

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

- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right)$ before and after Step A?

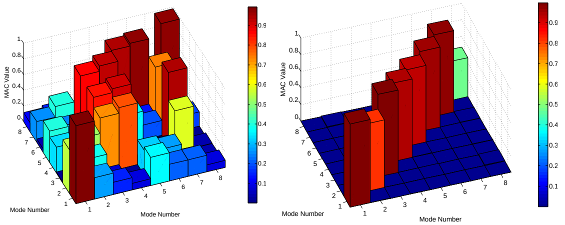


- When ϕ and ψ are normalized, $\text{MAC}(\phi,\psi) = |\phi^T\psi|^2$
- $\mathbf{P}_{l,l_0}$ is the matrix of square roots of the MACs between the modes contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)$ and those contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right)$

- Modal Assurance Criterion (MAC) between two modes ϕ and ψ

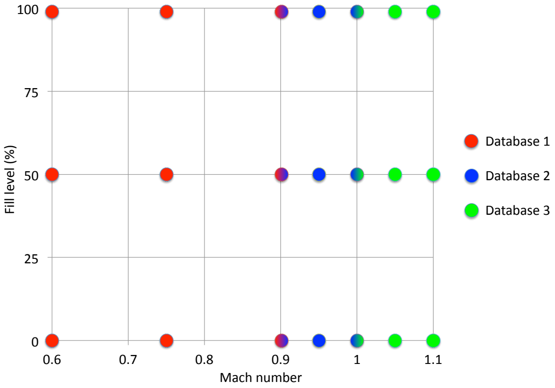
$$\text{MAC}(\phi, \psi) = \frac{|\phi^T \psi|^2}{(\phi^T \phi)(\psi^T \psi)}$$

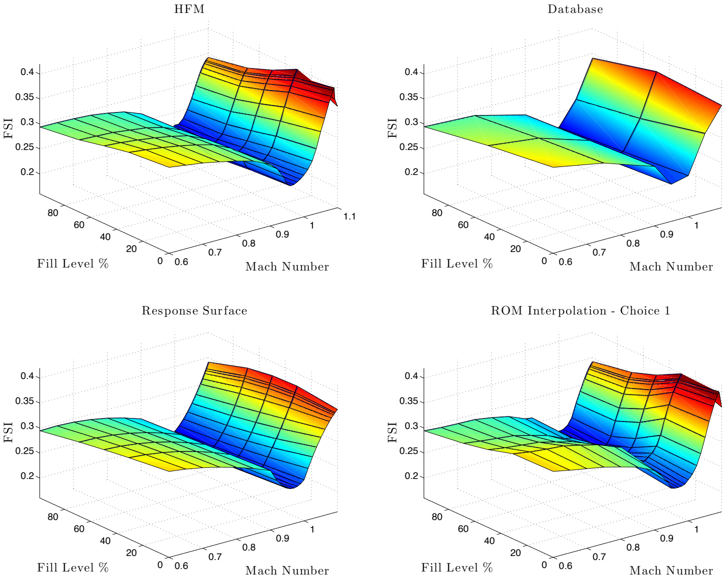
- What is the MAC between the vectors in the ROBs $\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)$ and $\mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right)$ before and after Step A?



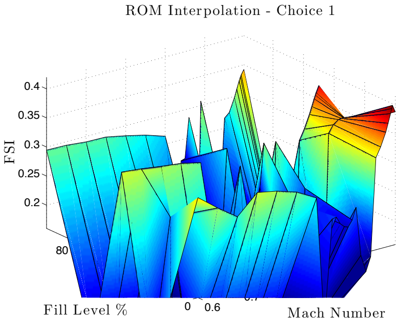
- When ϕ and ψ are normalized, $\text{MAC}(\phi, \psi) = |\phi^T \psi|^2$
- $\mathbf{P}_{l,l_0}$ is the matrix of square roots of the MACs between the modes contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(l)}\right)$ and those contained in $\mathbf{V}\left(\boldsymbol{\mu}^{(l_0)}\right)$
- This is 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_∞
- Database approach



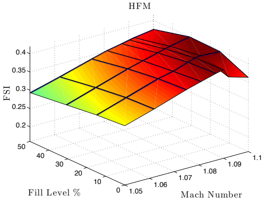

HFM
Database
Response Surface
ROM Interpolation - Choice 1
FSI
Fill Level %
Mach Number

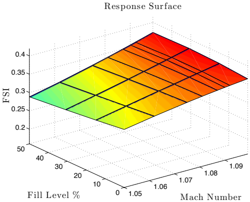
- Effect of Step A

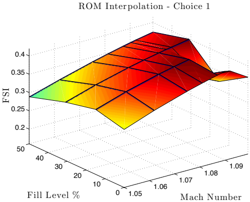


- Skipping Step A leads to inaccurate interpolation results (left figure)
- Step A ensures a consistent interpolation (right figure)

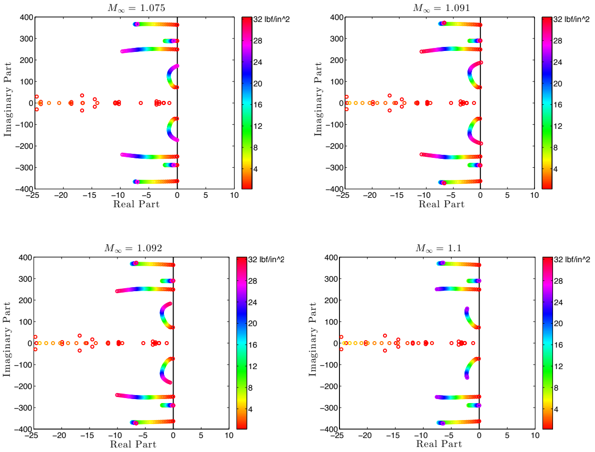
- The consistent interpolation on a matrix manifold is able to detect aeroelastic bifurcations



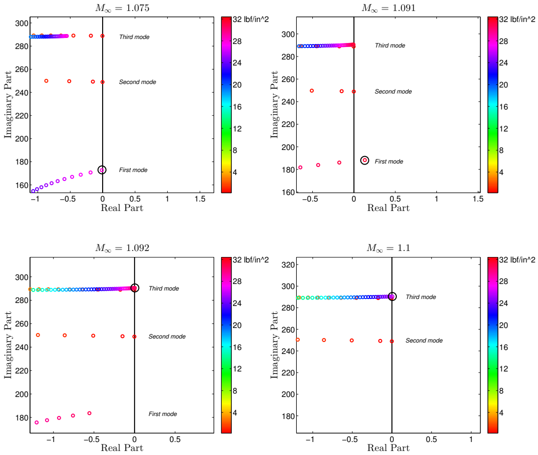




- The consistent interpolation on a matrix manifold is able to detect aeroelastic bifurcations (0% fuel fill level)



- The consistent interpolation on a matrix manifold is able to detect aeroelastic bifurcations (0% fuel fill level)

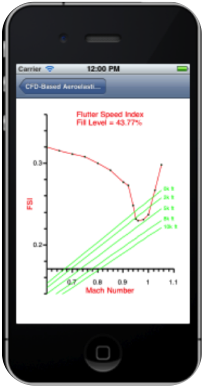


- CPU performance

Approach	Offline phase CPU time (# procs)	Online phase CPU time (# procs)
HDM	- (-)	9,152,000 s $\approx$ 106 days (32)
Response Surface	28,000 s $\approx$ 7 h (32)	2 s (1)
PROM Interpolation	28,000 s $\approx$ 7 h (32)	30 s (1)

- Online speedup $= 305,000$x
- Offline+Online speedup $= 327$x

- Mobile computing using a database of PROMs



- 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.