

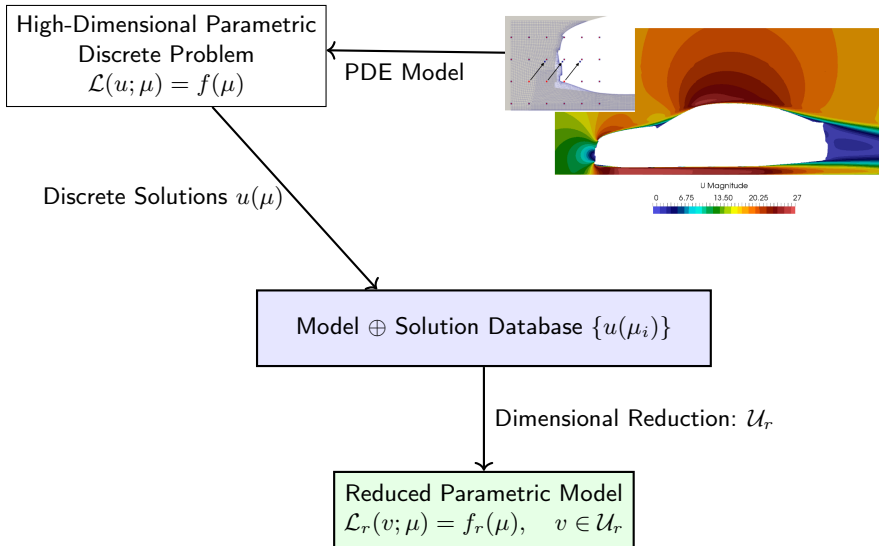
# Introduction to Model Reduction

Angelo Iollo

Institut de Mathématiques de Bordeaux, Université de Bordeaux  
Memphis Team, Centre Inria de l'Université de Bordeaux

Hammamet, February 18-22, 2025

# Multi-Query Parametric Problem Governed by PDEs



- Consider a PDE:

$$\mathcal{L}(u; \mu) = f(\mu), \quad u \in \mathcal{U}, \quad \mu \in \mathcal{P},$$

where:

- Consider a PDE:

$$\mathcal{L}(u; \mu) = f(\mu), \quad u \in \mathcal{U}, \quad \mu \in \mathcal{P},$$

where:

- $\mathcal{L}$ : Differential operator.

- Consider a PDE:

$$\mathcal{L}(u; \mu) = f(\mu), \quad u \in \mathcal{U}, \quad \mu \in \mathcal{P},$$

where:

- $\mathcal{L}$ : Differential operator.
- $u$ : Solution in function space  $\mathcal{U}$ .

- Consider a PDE:

$$\mathcal{L}(u; \mu) = f(\mu), \quad u \in \mathcal{U}, \quad \mu \in \mathcal{P},$$

where:

- $\mathcal{L}$ : Differential operator.
- $u$ : Solution in function space  $\mathcal{U}$ .
- $\mu$ : Parameter in parameter space  $\mathcal{P}$ .

# Fundamental Ansatz for PDE Solution

- Consider a PDE:

$$\mathcal{L}(u; \mu) = f(\mu), \quad u \in \mathcal{U}, \quad \mu \in \mathcal{P},$$

where:

- $\mathcal{L}$ : Differential operator.
  - $u$ : Solution in function space  $\mathcal{U}$ .
  - $\mu$ : Parameter in parameter space  $\mathcal{P}$ .
- Linear space approximation:

$$u_r(\mu) \in \text{span}\{u_1, u_2, \dots, u_k\} \quad u_k = u(\mu_k),$$

where  $\{u_i\}_{i=1}^k$  are previously computed solutions (snapshots).

# Fundamental Ansatz for PDE Solution

- Consider a PDE:

$$\mathcal{L}(u; \mu) = f(\mu), \quad u \in \mathcal{U}, \quad \mu \in \mathcal{P},$$

where:

- $\mathcal{L}$ : Differential operator.
  - $u$ : Solution in function space  $\mathcal{U}$ .
  - $\mu$ : Parameter in parameter space  $\mathcal{P}$ .
- Linear space approximation:

$$u_r(\mu) \in \text{span}\{u_1, u_2, \dots, u_k\} \quad u_k = u(\mu_k),$$

where  $\{u_i\}_{i=1}^k$  are previously computed solutions (snapshots).

- We have:

$$u(\mu) \approx u_r(\mu) = \sum_{i=1}^k c_i(\mu) u_i.$$

# Reduction of Function Space

- Affine space approximation:

$$u(\mu) \approx u_0(\mu) + \sum_{i=1}^k c_i(\mu)(u_i - u_0(\mu_i)),$$

where  $u_0(\mu)$  is a reference solution (a lifting of the b.c. for example).

# Reduction of Function Space

- Affine space approximation:

$$u(\mu) \approx u_0(\mu) + \sum_{i=1}^k c_i(\mu)(u_i - u_0(\mu_i)),$$

where  $u_0(\mu)$  is a reference solution (a lifting of the b.c. for example).

- Informal notion of reduced function space:

# Reduction of Function Space

- Affine space approximation:

$$u(\mu) \approx u_0(\mu) + \sum_{i=1}^k c_i(\mu)(u_i - u_0(\mu_i)),$$

where  $u_0(\mu)$  is a reference solution (a lifting of the b.c. for example).

- Informal notion of reduced function space:
  - Construct a reduced space  $\mathcal{U}_r \subset \mathcal{U}$ , where:

$$\mathcal{U}_r \approx \text{span}\{u_1, u_2, \dots, u_k\}.$$

# Reduction of Function Space

- Affine space approximation:

$$u(\mu) \approx u_0(\mu) + \sum_{i=1}^k c_i(\mu)(u_i - u_0(\mu_i)),$$

where  $u_0(\mu)$  is a reference solution (a lifting of the b.c. for example).

- Informal notion of reduced function space:
  - Construct a reduced space  $\mathcal{U}_r \subset \mathcal{U}$ , where:

$$\mathcal{U}_r \approx \text{span}\{u_1, u_2, \dots, u_k\}.$$

- Solve the reduced problem:

$$\mathcal{L}_r(v; \mu) = f_r(\mu), \quad v \in \mathcal{U}_r.$$

# Reduction of Function Space

- Affine space approximation:

$$u(\mu) \approx u_0(\mu) + \sum_{i=1}^k c_i(\mu)(u_i - u_0(\mu_i)),$$

where  $u_0(\mu)$  is a reference solution (a lifting of the b.c. for example).

- Informal notion of reduced function space:
  - Construct a reduced space  $\mathcal{U}_r \subset \mathcal{U}$ , where:

$$\mathcal{U}_r \approx \text{span}\{u_1, u_2, \dots, u_k\}.$$

- Solve the reduced problem:

$$\mathcal{L}_r(v; \mu) = f_r(\mu), \quad v \in \mathcal{U}_r.$$

- Reduction goal: Retain accuracy while significantly reducing computational complexity.

# Plan of the Course

- Part 1: Constructing the Reduced Space

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.
  - Latin Hypercube Sampling (LHS) for higher dimensions.

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.
  - Latin Hypercube Sampling (LHS) for higher dimensions.
  - Adaptive and goal-oriented greedy sampling based on error indicators.

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.
  - Latin Hypercube Sampling (LHS) for higher dimensions.
  - Adaptive and goal-oriented greedy sampling based on error indicators.
- Part 3: Solving in the Reduced Space

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.
  - Latin Hypercube Sampling (LHS) for higher dimensions.
  - Adaptive and goal-oriented greedy sampling based on error indicators.
- Part 3: Solving in the Reduced Space
  - Interpolation in the parameter space.

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.
  - Latin Hypercube Sampling (LHS) for higher dimensions.
  - Adaptive and goal-oriented greedy sampling based on error indicators.
- Part 3: Solving in the Reduced Space
  - Interpolation in the parameter space.
  - pMOR

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.
  - Latin Hypercube Sampling (LHS) for higher dimensions.
  - Adaptive and goal-oriented greedy sampling based on error indicators.
- Part 3: Solving in the Reduced Space
  - Interpolation in the parameter space.
  - pMOR
  - cMOR

# Plan of the Course

- Part 1: Constructing the Reduced Space
  - Linear approaches:
    - Proper Orthogonal Decomposition (POD).
    - Singular Value Decomposition (SVD).
    - Reduced Basis (RB) methods.
  - Non-linear approaches:
    - Optimal transportation-based methods.
    - Quadratic approximation manifold techniques.
- Part 2: Sampling the Parameter Space
  - Uniform sampling for moderate dimensions.
  - Latin Hypercube Sampling (LHS) for higher dimensions.
  - Adaptive and goal-oriented greedy sampling based on error indicators.
- Part 3: Solving in the Reduced Space
  - Interpolation in the parameter space.
  - pMOR
  - cMOR
  - Mention of closure models for non-intrusive reduced-order systems.

# Learning Objectives

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.

# Learning Objectives

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.
- Learn and apply key mathematical tools:

# Learning Objectives

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.
- Learn and apply key mathematical tools:
  - Proper Orthogonal Decomposition (POD) and Singular Value Decomposition (SVD),

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.
- Learn and apply key mathematical tools:
  - Proper Orthogonal Decomposition (POD) and Singular Value Decomposition (SVD),
  - Reduced Basis (RB) methods.

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.
- Learn and apply key mathematical tools:
  - Proper Orthogonal Decomposition (POD) and Singular Value Decomposition (SVD),
  - Reduced Basis (RB) methods.
  - Construct a pMOR

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.
- Learn and apply key mathematical tools:
  - Proper Orthogonal Decomposition (POD) and Singular Value Decomposition (SVD),
  - Reduced Basis (RB) methods.
  - Construct a pMOR
  - Construct a cMOR

# Learning Objectives

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.
- Learn and apply key mathematical tools:
  - Proper Orthogonal Decomposition (POD) and Singular Value Decomposition (SVD),
  - Reduced Basis (RB) methods.
  - Construct a pMOR
  - Construct a cMOR
- Formulate and solve a simple parametric PDE using reduced-order models.

# Learning Objectives

By the end of this course, participants should be able to:

- Understand the fundamental principles of projection-based model reduction.
- Learn and apply key mathematical tools:
  - Proper Orthogonal Decomposition (POD) and Singular Value Decomposition (SVD),
  - Reduced Basis (RB) methods.
  - Construct a pMOR
  - Construct a cMOR
- Formulate and solve a simple parametric PDE using reduced-order models.
- Understand the links between parameter sampling, accuracy, stability, and computational efficiency.

- **Project then Discretize:**

- Derive reduced-order equations in continuous form.
- Apply numerical discretization to the reduced equations.
- Advantages:
  - Preserves some structure of the original system.
  - Easier to derive.

- **Discretize then Project:**

- Discretize the full-order model first.
- Apply projection to reduce the system size.
- Advantages:
  - Numerical stability induced by the discretization scheme.
  - Consistent with the full-order model.

- **Comparison:**

Accuracy and stability vs. intrusivity

## Part 1: Constructing the Reduced Space

# High-Dimensional Model

- Nonlinear High-Dimensional Model:

$$\frac{d\mathbf{w}}{dt} = \mathbf{f}(\mathbf{w}(t), t), \quad \mathbf{y}(t) = \mathbf{g}(\mathbf{w}(t), t).$$

- Initial condition:

$$\mathbf{w}(0) = \mathbf{w}_0.$$

- Variables:

- $\mathbf{w} \in \mathbb{R}^N$ : State vector.
- $\mathbf{y} \in \mathbb{R}^q$ : Output vector, typically  $q \ll N$ .

- Function  $\mathbf{f}$  defines the dynamics.

# Proper Orthogonal Decomposition (POD)

- Consider a fixed initial condition  $\mathbf{w}_0 \in \mathbb{R}^N$
- Denote the associated state trajectory in the time-interval  $[0, \mathcal{T}]$  by

$$\mathcal{T}_{\mathbf{w}} = \{\mathbf{w}(t)\}_{0 \leq t \leq \mathcal{T}}$$

- The Proper Orthogonal Decomposition (POD) method seeks an orthonormal basis  $\mathbf{V} \in \mathbb{R}^{N \times k}$  defining a projector  $\Pi_{\mathbf{v}} = \mathbf{V}\mathbf{V}^T$  of fixed rank  $k$  that minimizes the integrated projection error

$$J(\Pi_{\mathbf{v}}, \mathbf{w}) = \int_0^{\mathcal{T}} \|\mathbf{w}(t) - \Pi_{\mathbf{v}}\mathbf{w}(t)\|_2^2 dt$$

# Theorem

Let  $\hat{\mathbf{K}} \in \mathbb{R}^{N \times N}$  be the real, symmetric, positive, semi-definite matrix defined as follows

$$\hat{\mathbf{K}} = \int_0^T \mathbf{w}(t) \mathbf{w}(t)^T dt$$

Let  $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \dots \geq \hat{\lambda}_N \geq 0$  denote the ordered eigenvalues of  $\hat{\mathbf{K}}$  and  $\hat{\phi}_i \in \mathbb{R}^N$ ,  $i = 1, \dots, N$ , denote their associated eigenvectors which are also referred to as the POD modes

$$\hat{\mathbf{K}} \hat{\phi}_i = \hat{\lambda}_i \hat{\phi}_i, \quad i = 1, \dots, N$$

The subspace  $\hat{\mathcal{V}} = \text{span}(\hat{\phi}_1, \dots, \hat{\phi}_k)$  of dimension  $k$  minimizes  $J(\Pi_{\mathcal{V}}, \mathbf{w})$ . It is the invariant subspace of  $\hat{\mathbf{K}}$  associated with the eigenvalues  $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \dots \geq \hat{\lambda}_k$

# Snapshot Method for POD

- Solving the eigenvalue problem  $\mathbf{K}\hat{\phi}_i = \hat{\lambda}_i\hat{\phi}_i$  is in general computationally intractable because:

# Snapshot Method for POD

- Solving the eigenvalue problem  $\hat{\mathbf{K}}\hat{\phi}_i = \hat{\lambda}_i\hat{\phi}_i$  is in general computationally intractable because:
  - The dimension  $N$  of the matrix  $\hat{\mathbf{K}}$  is usually large

# Snapshot Method for POD

- Solving the eigenvalue problem  $\hat{\mathbf{K}}\hat{\phi}_i = \hat{\lambda}_i\hat{\phi}_i$  is in general computationally intractable because:
  - The dimension  $N$  of the matrix  $\hat{\mathbf{K}}$  is usually large
  - This matrix is usually dense

# Snapshot Method for POD

- Solving the eigenvalue problem  $\hat{\mathbf{K}}\hat{\phi}_i = \hat{\lambda}_i\hat{\phi}_i$  is in general computationally intractable because:
  - The dimension  $N$  of the matrix  $\hat{\mathbf{K}}$  is usually large
  - This matrix is usually dense
- However, the state data is typically available under the form of discrete "snapshot" vectors

$$\{\mathbf{w}(t_i)\}_{i=1}^{N_{\text{snap}}}$$

# Snapshot Method for POD

- Solving the eigenvalue problem  $\hat{\mathbf{K}}\hat{\phi}_i = \hat{\lambda}_i\hat{\phi}_i$  is in general computationally intractable because:
  - The dimension  $N$  of the matrix  $\hat{\mathbf{K}}$  is usually large
  - This matrix is usually dense
- However, the state data is typically available under the form of discrete "snapshot" vectors

$$\{\mathbf{w}(t_i)\}_{i=1}^{N_{\text{snap}}}$$

- In this case,  $\int_0^T \mathbf{w}(t)\mathbf{w}(t)^T dt$  can be approximated using a quadrature rule as follows

$$\mathbf{K} = \sum_{i=1}^{N_{\text{snap}}} \alpha_i \mathbf{w}(t_i)\mathbf{w}(t_i)^T$$

where  $\alpha_i$ ,  $i = 1, \dots, N_{\text{snap}}$  are the quadrature weights

- Define the snapshot matrix  $\mathbf{S} \in \mathbb{R}^{N \times N_{\text{snap}}}$  as:

$$\mathbf{S} = [\sqrt{\alpha_1} \mathbf{w}(t_1) \quad \cdots \quad \sqrt{\alpha_{N_{\text{snap}}}} \mathbf{w}(t_{N_{\text{snap}}})]$$

- It follows that:

$$\mathbf{K} = \mathbf{S} \mathbf{S}^T$$

- Here,  $\mathbf{K}$  is still a large-scale  $(N \times N)$  matrix

- Note that the non-zero eigenvalues of the matrix  $\mathbf{K} = \mathbf{S}\mathbf{S}^T \in \mathbb{R}^{N \times N}$  are the same as those of the matrix  $\mathbf{R} = \mathbf{S}^T\mathbf{S} \in \mathbb{R}^{N_{\text{snap}} \times N_{\text{snap}}}$
- Since usually  $N_{\text{snap}} \ll N$ , it is more economical to solve instead the symmetric eigenvalue problem

$$\mathbf{R}\psi_i = \lambda_i\psi_i, \quad i = 1, \dots, N_{\text{snap}}$$

- However, if  $\mathbf{S}$  is ill-conditioned,  $\mathbf{R}$  is worse conditioned

$$\kappa_2(\mathbf{R}) = \kappa_2(\mathbf{S})^2$$

# Snapshot Method for POD

- If  $\text{rank}(\mathbf{R}) = r$ , then the first  $r$  POD modes  $\phi_i$  are given by

$$\phi_i = \frac{1}{\sqrt{\lambda_i}} \mathbf{S} \psi_i, \quad i = 1, \dots, r$$

- Let  $\Phi = [\phi_1 \cdots \phi_r]$  and  $\Psi = [\psi_1 \cdots \psi_r]$  with  $\Psi^T \Psi = \mathbf{I}_r \implies \Phi = \mathbf{S} \Psi \Lambda^{-\frac{1}{2}}$  where

$$\Lambda = \begin{bmatrix} \lambda_1 & (0) \\ (0) & \lambda_r \end{bmatrix}$$

- $\mathbf{R} \psi_i = \lambda_i \psi_i, \quad i = 1, \dots, N_{\text{snap}} \implies \Psi^T \mathbf{R} \Psi = \Psi^T \mathbf{S}^T \mathbf{S} \Psi = \Lambda$
- Hence,  $\Phi^T \mathbf{K} \Phi = \Lambda^{-\frac{1}{2}} \Psi^T \mathbf{S}^T \mathbf{S} \Psi \Lambda^{-\frac{1}{2}} = \Lambda^{-\frac{1}{2}} \Lambda \Psi^T \Psi \Lambda^{-\frac{1}{2}} = \Lambda$
- Since the columns of  $\Phi$  are the eigenvectors of  $\mathbf{K}$  ordered by decreasing eigenvalues, the optimal orthogonal basis of size  $k \leq r$  is

$$\mathbf{V} = [\Phi_k \quad \Phi_{r-k}] \begin{bmatrix} I_k \\ 0 \end{bmatrix} = \Phi_k$$

# Relationship with Singular Value Decomposition (SVD)

- For a given matrix  $\mathbf{A} \in \mathbb{R}^{N \times M}$ , there exist orthogonal matrices  $\mathbf{U} \in \mathbb{R}^{N \times N}$  and  $\mathbf{Z} \in \mathbb{R}^{M \times M}$  such that

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{Z}^T$$

where  $\mathbf{U}^T\mathbf{U} = \mathbf{I}_N$ ,  $\mathbf{Z}^T\mathbf{Z} = \mathbf{I}_M$  and matrix  $\mathbf{\Sigma} \in \mathbb{R}^{N \times M}$  has diagonal entries  $\Sigma_{ii} = \sigma_i$  satisfying  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(N,M)} \geq 0$  and zero entries elsewhere

- $\{\sigma_i\}_{i=1}^{\min(N,M)}$  are the singular values of  $\mathbf{A}$
- Columns of  $\mathbf{U}$  and  $\mathbf{Z}$  are left and right singular vectors of  $\mathbf{A}$ , with  $\mathbf{U} = [\mathbf{u}_1 \dots \mathbf{u}_N]$  and  $\mathbf{Z} = [\mathbf{z}_1 \dots \mathbf{z}_M]$

# Schmidt-Eckart-Young-Mirsky Theorem

- Given  $\mathbf{A} \in \mathbb{R}^{N \times M}$  with  $N \geq M$ , which matrix  $\mathbf{X} \in \mathbb{R}^{N \times M}$  with  $\text{rank}(\mathbf{X}) = k < r = \text{rank}(\mathbf{A}) \leq M$  minimizes  $\|\mathbf{A} - \mathbf{X}\|_2$ ?
- Theorem (Schmidt-Eckart-Young-Mirsky):

$$\min_{\mathbf{X}, \text{rank}(\mathbf{X})=k} \|\mathbf{A} - \mathbf{X}\|_2 = \sigma_{k+1}(\mathbf{A}), \quad \text{if } \sigma_k(\mathbf{A}) > \sigma_{k+1}(\mathbf{A})$$

- $\mathbf{X} = \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{z}_i^T$ , minimizes  $\|\mathbf{A} - \mathbf{X}\|_2$ , where  $\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{Z}^T$ .
- This minimizer is also the unique solution of the related problem (Eckart-Young theorem)

$$\min_{\mathbf{X}, \text{rank}(\mathbf{X})=k} \|\mathbf{A} - \mathbf{X}\|_F = \left( \sum_{i=k+1}^r \sigma_i^2 \right)^{1/2}$$

- This result explains the concept of "low-rank" approximation and its connection with SVD

# Relationship POD-SVD

- The discretization of the POD by the method of snapshots requires computing the eigenspectrum of  $\mathbf{K} = \mathbf{S}\mathbf{S}^T$

$$\Phi^T \mathbf{K} \Phi = \Phi^T \mathbf{S} \mathbf{S}^T \Phi = \Lambda$$

corresponding to its non-zero eigenvalues

- Link with the SVD of  $\mathbf{S}$

$$\mathbf{S} = \mathbf{U} \Sigma \mathbf{Z}^T = [\mathbf{U}_r \quad \mathbf{U}_{N-r}] \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} \mathbf{Z}^T$$

$$\implies \mathbf{K} = \mathbf{U} \Sigma^2 \mathbf{U}^T \quad \text{and} \quad \mathbf{U}^T \mathbf{K} \mathbf{U} = \Sigma^2$$

$$\implies \Phi = \mathbf{U}_r \quad \text{and} \quad \Lambda^{\frac{1}{2}} = \Sigma_r \iff \Lambda = \Sigma_r^2$$

- Computing the SVD of  $\mathbf{S}$  is usually preferred since as noted as noted earlier  $\kappa_2(\mathbf{R}) = \kappa_2(\mathbf{S})^2$ .

# Greedy Approach for Basis Selection - Reduced Basis

**Goal:** Select a subset of snapshots to form a reduced basis spanning  $\mathcal{U}_r$ .

## Algorithm:

- ❶ Initialize the reduced space:  $\mathcal{U}_r = \{0\}$ .
- ❷ For each iteration:
  - Find the parameter  $\mu_i \in \mathcal{P}$  that maximizes the error indicator:

$$\mu_i = \arg \max_{\mu \in \mathcal{P}} \Delta(u(\mu), \mathcal{U}_r),$$

where typically  $\Delta(u(\mu), \mathcal{U}_r) = \|u(\mu) - \Pi_{\mathcal{U}_r} u(\mu)\|_{\mathcal{X}} := .$

- Enrich the reduced space:

$$\mathcal{U}_r \leftarrow \mathcal{U}_r \cup \{u(\mu_i)\}.$$

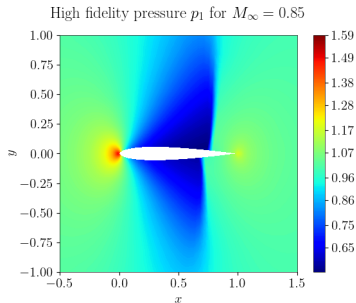
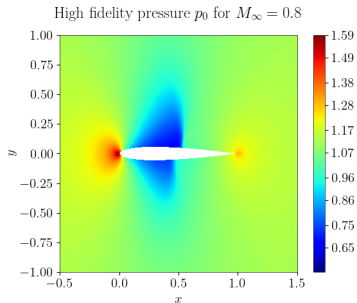
- ❸ Stop when  $\Delta(u(\mu), \mathcal{U}_r) < \epsilon$  or the basis reaches the desired size  $k$ .

## Key Features:

- Maximizes the worst-case error at each step.
- Ensures an optimal reduced basis for a given number of functions.

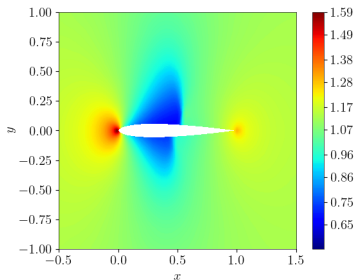
## Non-linear Interpolation

# Motivation

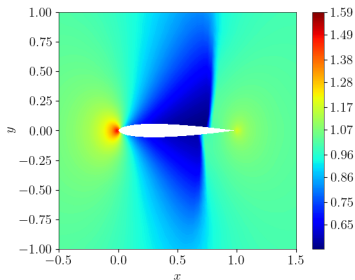


# Motivation

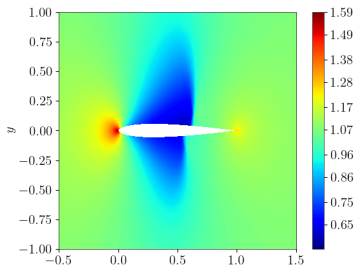
High fidelity pressure  $p_0$  for  $M_\infty = 0.8$



High fidelity pressure  $p_1$  for  $M_\infty = 0.85$

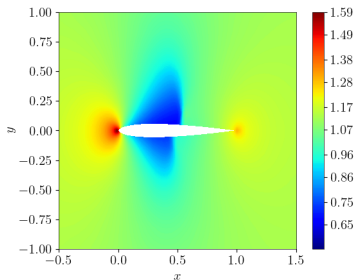


High fidelity pressure field for  $M_\infty = 0.82$

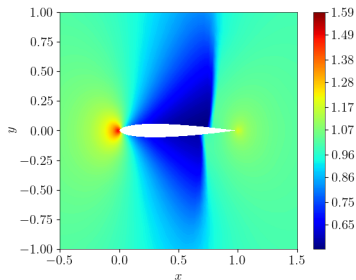


# Motivation

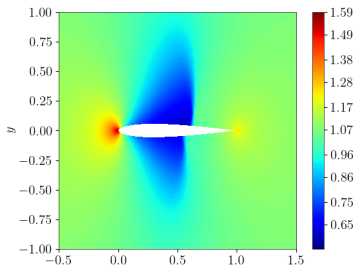
High fidelity pressure  $p_0$  for  $M_\infty = 0.8$



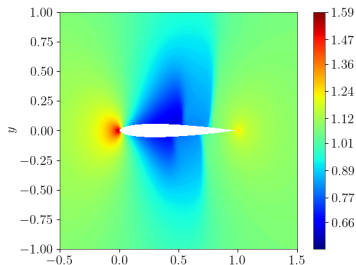
High fidelity pressure  $p_1$  for  $M_\infty = 0.85$



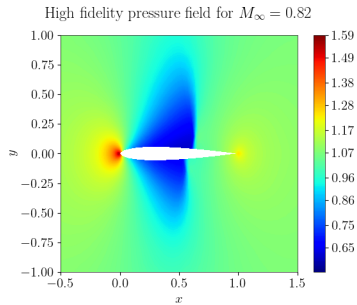
High fidelity pressure field for  $M_\infty = 0.82$



Linear interpolation of  $p_0$  and  $p_1$  for  $M_\infty = 0.82$

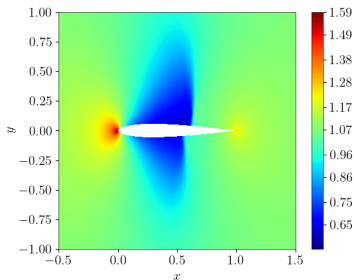


# Motivation

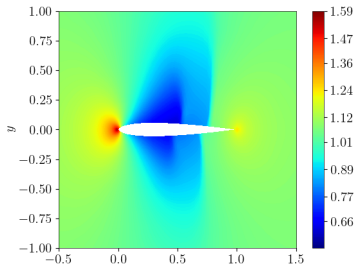


# Motivation

High fidelity pressure field for  $M_\infty = 0.82$

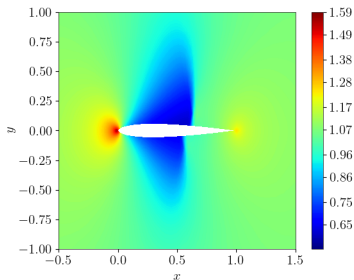


Linear interpolation of  $p_0$  and  $p_1$  for  $M_\infty = 0.82$

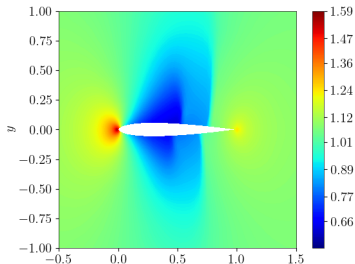


# Motivation

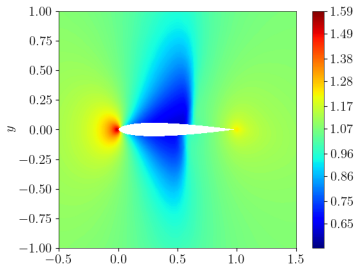
High fidelity pressure field for  $M_\infty = 0.82$



Linear interpolation of  $p_0$  and  $p_1$  for  $M_\infty = 0.82$



Non-linear interpolation of  $p_0$  and  $p_1$  for  $M_\infty = 0.82$



Let  $\mathcal{P}$  be a compact set and  $\varphi : \mathcal{P} \rightarrow \Omega$  be the mapping associating any  $z \in \mathcal{P}$  to a parametric solution  $u = \varphi(z) \in \Omega$  under the constraint given by the PDE:

$$\mathcal{R}(u(z), z) = 0, \quad \forall z \in \mathcal{P}$$

The PDE implicitly defines a manifold in the Hilbert space  $(V, \|\cdot\|)$ , with  $\text{im } \varphi \subset V$ .

Let  $V_n$  be an  $n$ -dimensional ( $n < +\infty$ ) subspace of  $V$ .

The Kolmogorov width is defined as:

$$d_{V_n} = \inf_{V_n \subset V} \sup_{v \in \text{im } \varphi} \inf_{v_n \in V_n} \|v - v_n\|$$

with  $V_n$  being a generic  $n$ -dimensional subspace of  $V$ .

# Preliminary notions

We are interested in a continuous medium that occupies the space  $\Omega_0$  in the reference (or initial) configuration and  $\Omega_t$  in the deformed configuration at time  $t$ . All the domains considered hereafter are included in  $\mathbb{R}^2$  or  $\mathbb{R}^3$ .

We are interested in a continuous medium that occupies the space  $\Omega_0$  in the reference (or initial) configuration and  $\Omega_t$  in the deformed configuration at time  $t$ . All the domains considered hereafter are included in  $\mathbb{R}^2$  or  $\mathbb{R}^3$ . We define the direct characteristics  $X$

$$\begin{aligned} X : \Omega_0 \times [0, T] &\longrightarrow \Omega_t \\ (\xi, t) &\longmapsto X(\xi, t) \end{aligned}$$

We are interested in a continuous medium that occupies the space  $\Omega_0$  in the reference (or initial) configuration and  $\Omega_t$  in the deformed configuration at time  $t$ . All the domains considered hereafter are included in  $\mathbb{R}^2$  or  $\mathbb{R}^3$ . We define the direct characteristics  $X$

$$\begin{aligned} X : \Omega_0 \times [0, T] &\longrightarrow \Omega_t \\ (\xi, t) &\mapsto X(\xi, t) \end{aligned}$$

The "velocity" field is given by

$$\begin{aligned} u : \Omega_t \times [0, T] &\longrightarrow \mathbb{R}^3 \\ (x, t) &\mapsto u(x, t) \end{aligned}$$

# Eulerian & Lagrangian formulation

The Eulerian description consists of working with quantities on the deformed configuration  $\Omega_t$  (in the simplest case, the velocity field  $u(x, t)$ ). The Lagrangian description consists of working with quantities on the reference configuration  $\Omega_0$  (the characteristics  $X(\xi, t)$ ).

The Eulerian description consists of working with quantities on the deformed configuration  $\Omega_t$  (in the simplest case, the velocity field  $u(x, t)$ ). The Lagrangian description consists of working with quantities on the reference configuration  $\Omega_0$  (the characteristics  $X(\xi, t)$ ).

The two formulations are equivalent by virtue of the relation

$$\frac{\partial X}{\partial t}(\xi, t) = u(X(\xi, t), t)$$

which is complemented with the initial condition  $X(\xi, 0) = \xi$ .

We assume throughout that the deformations  $X(\cdot, t)$  are smooth, bijective mappings between manifolds where both the map and its inverse are smooth, and do not change the orientation, that is

$$\det(\nabla_{\xi} X(\xi, t)) > 0$$

We assume throughout that the deformations  $X(\cdot, t)$  are smooth, bijective mappings between manifolds where both the map and its inverse are smooth, and do not change the orientation, that is

$$\det(\nabla_{\xi} X(\xi, t)) > 0$$

Under these hypothesis, for volumes, the change of variable  $x = X(\xi, t)$  allows us to revert to the reference configuration

$$\int_{X(\Omega_0, t) = \Omega_t} f(x, t) dx = \int_{\Omega_0} f(X(\xi, t), t) \det(\nabla_{\xi} X(\xi, t)) d\xi$$

Reynolds formula for volumes is written as

$$\frac{d}{dt} \left( \int_{\Omega_t} f(x, t) dx \right) = \int_{\Omega_t} (f_t + \operatorname{div}(fu)) dx$$

Reynolds formula for volumes is written as

$$\frac{d}{dt} \left( \int_{\Omega_t} f(x, t) dx \right) = \int_{\Omega_t} (f_t + \operatorname{div}(fu)) dx$$

## Conservation of Mass in Eulerian Form

The conservation of mass states that the mass variation of a volume  $\Omega_t$  is independent of time

$$\frac{d}{dt} \left( \int_{\Omega_t} \rho(x, t) dx \right) = 0$$

Using Reynolds formula with  $f = \rho$ , we obtain the conservation of mass in the deformed configuration

$$\boxed{\rho_t + \operatorname{div}(\rho u) = 0}$$

We simply revert to the reference configuration

$$\frac{d}{dt} \left( \int_{\Omega_0} \rho(X(\xi, t), t) \det(\nabla_{\xi} X(\xi, t)) d\xi \right) = 0$$

which can be written, since  $X(\xi, 0) = \xi$ ,

$$\boxed{\rho(X(\xi, t), t) \det(\nabla_{\xi} X(\xi, t)) = \rho_0(\xi)}$$

# Inverse mapping

We introduce the backward characteristics  $Y$

$$\begin{aligned} Y : \Omega_t \times [0, T] &\longrightarrow \Omega_0 \\ (x, t) &\mapsto Y(x, t) \end{aligned}$$

# Inverse mapping

We introduce the backward characteristics  $Y$

$$\begin{aligned} Y : \Omega_t \times [0, T] &\longrightarrow \Omega_0 \\ (x, t) &\mapsto Y(x, t) \end{aligned}$$

These functions are related to the direct characteristics  $X$  by the relations

$$X(Y(x, t), t) = x \qquad Y(X(\xi, t), t) = \xi$$

# Inverse mapping

We introduce the backward characteristics  $Y$

$$\begin{aligned} Y : \Omega_t \times [0, T] &\longrightarrow \Omega_0 \\ (x, t) &\mapsto Y(x, t) \end{aligned}$$

These functions are related to the direct characteristics  $X$  by the relations

$$X(Y(x, t), t) = x \qquad Y(X(\xi, t), t) = \xi$$

By derivation of the first equation with respect to  $x$  and the second with respect to  $t$ :

$$[\nabla_\xi X(\xi, t)] = [\nabla_x Y(x, t)]^{-1} \qquad Y_t + (u \cdot \nabla)Y = 0$$

# Inverse mapping

We introduce the backward characteristics  $Y$

$$\begin{aligned} Y : \Omega_t \times [0, T] &\longrightarrow \Omega_0 \\ (x, t) &\mapsto Y(x, t) \end{aligned}$$

These functions are related to the direct characteristics  $X$  by the relations

$$X(Y(x, t), t) = x \qquad Y(X(\xi, t), t) = \xi$$

By derivation of the first equation with respect to  $x$  and the second with respect to  $t$ :

$$[\nabla_\xi X(\xi, t)] = [\nabla_x Y(x, t)]^{-1} \qquad Y_t + (u \cdot \nabla)Y = 0$$

...and mass conservation with respect to  $Y$

$$\rho(x, t) = \det(\nabla_x Y(x, t)) \rho(Y(x, t), 0)$$

where  $\rho(Y(x, t), 0) = \rho_0(Y(x, t))$

# Optimal transportation: a crush primer

We introduce the probability measures  $\mathbb{P}_0, \mathbb{P}_1$  with pdfs  $\rho_0, \rho_1$  and cumulative distribution functions  $F_0, F_1$ ,

$$F_i(x) = \int_{-\infty}^{x_1} \dots \int_{-\infty}^{x_n} \rho_i(x') dx', \quad i = 0, 1, \quad x \in \mathbb{R}^n.$$

# Optimal transportation: a crash primer

We introduce the probability measures  $\mathbb{P}_0, \mathbb{P}_1$  with pdfs  $\rho_0, \rho_1$  and cumulative distribution functions  $F_0, F_1$ ,

$$F_i(x) = \int_{-\infty}^{x_1} \dots \int_{-\infty}^{x_n} \rho_i(x') dx', \quad i = 0, 1, \quad x \in \mathbb{R}^n.$$

We assume that  $\rho_0, \rho_1$  have finite second-order moments. We say that  $X : \mathbb{R}^n \rightarrow \mathbb{R}^n$  transports  $\mathbb{P}_0$  to  $\mathbb{P}_1$  if  $\mathbb{P}_1(B) = \mathbb{P}_0(X^{-1}(B))$  for all  $\mathbb{P}_1$ -measurable sets  $B$ , with  $Y(B) = X^{-1}(B) := \{\xi \in \mathbb{R}^n : X(\xi) \in B\}$ , and we use notation  $\mathbb{P}_1 = X_{\#}\mathbb{P}_0$ .

Note that the latter implies local mass conservation

$$\rho_0(\xi) = \rho_1(X(\xi)) \det \nabla_{\xi} X(\xi), \quad \forall \xi \in \mathbb{R}^n,$$

or equivalently

$$F_0(\xi) = F_1(X(\xi)), \quad \forall \xi \in \mathbb{R}^n.$$

Note that the latter implies local mass conservation

$$\rho_0(\xi) = \rho_1(X(\xi)) \det \nabla_{\xi} X(\xi), \quad \forall \xi \in \mathbb{R}^n,$$

or equivalently

$$F_0(\xi) = F_1(X(\xi)), \quad \forall \xi \in \mathbb{R}^n.$$

With this notation, we can introduce the Monge optimal transport problem as follows: find  $X : \mathbb{R}^n \rightarrow \mathbb{R}^n$  to minimize

$$I(X; \rho_0, \rho_1) = \int_{\mathbb{R}^n} \|X(\xi) - \xi\|_2^2 \rho_0(\xi) d\xi,$$

under the constraint of mass conservation.

# Main result (Y. Brenier)

There exists a unique convex potential  $\Psi(\xi): \mathbb{R}^n \rightarrow \mathbb{R}$ , such that the mapping  $X(\xi) = \nabla_\xi \Psi$  minimizes  $I(X; \rho_0, \rho_1)$  subject to

$$\rho_0(\xi) = \rho_1(\nabla_\xi \Psi) \det(\nabla_\xi^2 \Psi).$$

Let  $J$  be the minimum of  $I(X; \rho_0, \rho_1)$  subject to mass conservation. Existence and uniqueness of  $\Psi$  implies that

$$W_2(\rho_0, \rho_1) = \sqrt{J}$$

is a distance function between probability measures;  $W_2$  is known as the Wasserstein metric.

The Wasserstein distance is a rigorous proxy of the notion of displacement.

# Displacement (McCann) interpolation

We introduce the displacement (or geodesic) interpolant  $\hat{\rho}_s^+$  between  $\rho_0$  and  $\rho_1$  such that

$$\hat{\rho}_s^+(\xi) = \rho_1(T(s, \xi)) \det(\nabla_\xi T(s, \xi)),$$

and the mapping  $T(s, \xi) : [0, 1] \times \mathbb{R}^n \rightarrow \mathbb{R}^n$  is defined as

$$T(s, \xi) = (1 - s) \xi + s \nabla_\xi \Psi(\xi).$$

Note that  $\hat{\rho}_s^+ = \rho_0$  for  $s = 0$  and  $\hat{\rho}_s^+ = \rho_1$  for  $s = 1$ .

We further introduce the inverse map  $W : \mathbb{R}_+ \times \mathbb{R}^n \rightarrow \mathbb{R}^n$  such that

$$W(s, \cdot) = T^{-1}(s, \cdot), \quad \forall s \in [0, 1],$$

and the reverse McCann interpolation, which is obtained by inverting the role of  $\rho_0$  and  $\rho_1$ :

$$\widehat{\rho}_s^-(x) = \rho_1(W(s, x)) \det(\nabla_x W(s, x)).$$

Linear models can be interpreted as a generalization of convex interpolations of two snapshots  $u_0, u_1$ , that is

$$\hat{u}(s, x) = (1 - s)u_0(x) + su_1(x) \quad s \in [0, 1], x \in \Omega.$$

The use of linear methods relies on the assumption that the problem of interest exhibits a global nature.

# Convex Displacement Interpolation (CDI)

- We define the nonlinear interpolation:

$$\widehat{u}(s, x) = (1 - s)u_0 \circ T_g^{-1}(s, x) + su_1 \circ W_g^{-1}(1 - s, x)$$

where  $s \in [0, 1], x \in \Omega$ .

- We define the nonlinear interpolation:

$$\widehat{u}(s, x) = (1 - s)u_0 \circ T_g^{-1}(s, x) + su_1 \circ W_g^{-1}(1 - s, x)$$

where  $s \in [0, 1], x \in \Omega$ .

- We refer to  $\widehat{u}$  as *convex displacement interpolation* due to the analogy with displacement interpolation and the more elementary convex interpolation.

# Convex Displacement Interpolation (CDI)

- We define the nonlinear interpolation:

$$\widehat{u}(s, x) = (1 - s)u_0 \circ T_g^{-1}(s, x) + su_1 \circ W_g^{-1}(1 - s, x)$$

where  $s \in [0, 1], x \in \Omega$ .

- We refer to  $\widehat{u}$  as *convex displacement interpolation* due to the analogy with displacement interpolation and the more elementary convex interpolation.
- In several relevant PDE problems this formula is an exact interpolation with respect to the parameter using the exact OT mapping

- We linearize  $T^{-1}(s, \cdot)$  and  $W^{-1}(s, \cdot)$

$$T_g^{-1}(s, \cdot) \approx (1-s)T_g^{-1}(0, \cdot) + sT_g^{-1}(1, \cdot) = (1-s) \cdot + sY_g(\cdot) = W_g(s, \cdot),$$

and

$$W_g^{-1}(s, \cdot) \approx (1-s)W_g^{-1}(0, \cdot) + sW_g^{-1}(1, \cdot) = (1-s) \cdot + sX(\cdot) = T_g(s, \cdot)$$

- We linearize  $T^{-1}(s, \cdot)$  and  $W^{-1}(s, \cdot)$

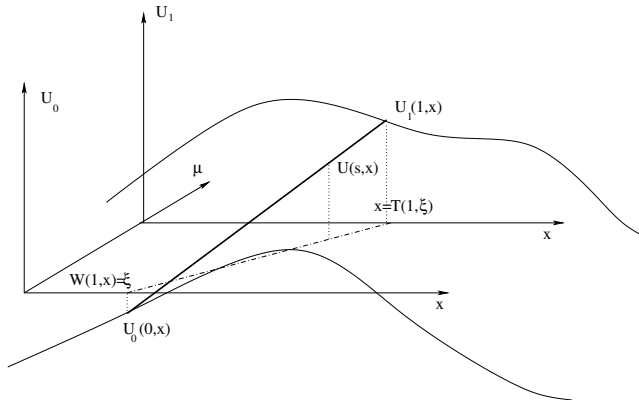
$$T_g^{-1}(s, \cdot) \approx (1-s)T_g^{-1}(0, \cdot) + sT_g^{-1}(1, \cdot) = (1-s) \cdot + sY_g(\cdot) = W_g(s, \cdot),$$

and

$$W_g^{-1}(s, \cdot) \approx (1-s)W_g^{-1}(0, \cdot) + sW_g^{-1}(1, \cdot) = (1-s) \cdot + sX(\cdot) = T_g(s, \cdot)$$

- The linearised formula is then:

$$\hat{u}(s, x) = (1-s)u_0 \circ W_g(s, x) + su_1 \circ T_g(1-s, x),$$



Remarks:

- ① CDI is symmetric
- ② CDI includes convex interpolation (e.g.  $T(1, \xi) = \xi$ )
- ③ CDI respects the maximum principle

$$\hat{U}(s, x) \leq \max(U_0(x), U_1(T_g(1, x))) = \max(U_0(W(1, x)), U_1(x))$$

# OT of multivariate normal density distributions

We define the normal density distribution  $\mathcal{N}$  with mean  $\mu \in \mathbb{R}^n$  and symmetric positive definite covariance  $\Sigma \in \mathbb{R}^{n \times n}$ :

$$\mathcal{N}(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}.$$

# OT of multivariate normal density distributions

We define the normal density distribution  $\mathcal{N}$  with mean  $\mu \in \mathbb{R}^n$  and symmetric positive definite covariance  $\Sigma \in \mathbb{R}^{n \times n}$ :

$$\mathcal{N}(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}.$$

Given the densities  $\pi = \mathcal{N}(\cdot; \mu_0, \Sigma_0)$  and  $\nu = \mathcal{N}(\cdot; \mu_1, \Sigma_1)$ , we find that the displacement interpolant  $\hat{\mathcal{N}}_s$  is Gaussian with mean and covariance given by

$$\mu_s = (1-s)\mu_0 + s\mu_1, \quad \Sigma_s = \Sigma_0^{-1/2} \left( (1-s)\Sigma_0 + s \left( \Sigma_0^{1/2} \Sigma_1 \Sigma_0^{1/2} \right)^{1/2} \right)^2 \Sigma_0^{-1/2}$$

for all  $s \in [0, 1]$ .

The forward mapping  $T$  is also available in closed form:

$$T(s, \xi) = (1 - s) \xi + s \left( \mu_1 + \Sigma_0^{-1/2} \left( \Sigma_0^{1/2} \Sigma_1 \Sigma_0^{1/2} \right)^{1/2} \Sigma_0^{-1/2} (\xi - \mu_0) \right).$$

Finally, the Wasserstein distance between Gaussian density distributions is given by:

$$\begin{aligned} W_2(\mathcal{N}(\mu_0, \Sigma_0), \mathcal{N}(\mu_1, \Sigma_1)) &= \\ &= \sqrt{\|\mu_1 - \mu_0\|_2^2 + \text{Tr} \left( \Sigma_0 + \Sigma_1 - 2 \left( \Sigma_0^{1/2} \Sigma_1 \Sigma_0^{1/2} \right)^{1/2} \right)} \end{aligned}$$

# Gaussian model of coherent structures

- Given the field  $U : \mathbb{R}^n \rightarrow \mathbb{R}^d$  and the scalar testing function  $\mathcal{T}(\cdot; U) : \mathbb{R}^n \rightarrow \mathbb{R}$ , the set

$$\mathcal{C}_{\mathcal{T}}(U) := \{x \in \mathbb{R}^n : \mathcal{T}(x; U) > 0\}$$

identifies coherent structures associated with the criterion  $\mathcal{T}$

# Gaussian model of coherent structures

- Given the field  $U : \mathbb{R}^n \rightarrow \mathbb{R}^d$  and the scalar testing function  $\mathcal{T}(\cdot; U) : \mathbb{R}^n \rightarrow \mathbb{R}$ , the set

$$\mathcal{C}_{\mathcal{T}}(U) := \{x \in \mathbb{R}^n : \mathcal{T}(x; U) > 0\}$$

identifies coherent structures associated with the criterion  $\mathcal{T}$

- For example, if  $U$  is a velocity field and  $\mathcal{T}(x; U) = \|\nabla \times U(x)\|_2 - \tau$  with  $\tau > 0$ ,  $\mathcal{C}_{\mathcal{T}}(U)$  identifies the region where enstrophy exceeds a threshold  $\tau$

# Gaussian model of coherent structures

- Given the field  $U : \mathbb{R}^n \rightarrow \mathbb{R}^d$  and the scalar testing function  $\mathcal{T}(\cdot; U) : \mathbb{R}^n \rightarrow \mathbb{R}$ , the set

$$\mathcal{C}_{\mathcal{T}}(U) := \{x \in \mathbb{R}^n : \mathcal{T}(x; U) > 0\}$$

identifies coherent structures associated with the criterion  $\mathcal{T}$

- For example, if  $U$  is a velocity field and  $\mathcal{T}(x; U) = \|\nabla \times U(x)\|_2 - \tau$  with  $\tau > 0$ ,  $\mathcal{C}_{\mathcal{T}}(U)$  identifies the region where enstrophy exceeds a threshold  $\tau$
- Objective: fit a Gaussian probability density function  $\phi(x; \mu, \Sigma)$  to  $\mathcal{C}_{\mathcal{T}}(U)$

$$\phi(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^{\top} \Sigma^{-1} (x-\mu)}.$$

# Gaussian model of coherent structures

- Given the field  $U : \mathbb{R}^n \rightarrow \mathbb{R}^d$  and the scalar testing function  $\mathcal{T}(\cdot; U) : \mathbb{R}^n \rightarrow \mathbb{R}$ , the set

$$\mathcal{C}_{\mathcal{T}}(U) := \{x \in \mathbb{R}^n : \mathcal{T}(x; U) > 0\}$$

identifies coherent structures associated with the criterion  $\mathcal{T}$

- For example, if  $U$  is a velocity field and  $\mathcal{T}(x; U) = \|\nabla \times U(x)\|_2 - \tau$  with  $\tau > 0$ ,  $\mathcal{C}_{\mathcal{T}}(U)$  identifies the region where enstrophy exceeds a threshold  $\tau$
- Objective: fit a Gaussian probability density function  $\phi(x; \mu, \Sigma)$  to  $\mathcal{C}_{\mathcal{T}}(U)$

$$\phi(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}.$$

- Define a finite-dimensional discretization of the domain of interest  $P_{\text{hf}} = \{x_i\}_{i=1}^{N_{\text{hf}}}$  and we define

$$P_{\text{hf}}^+ := \{x \in P_{\text{hf}} : \mathcal{T}(x; U) > 0\} = \{y_j\}_{j=1}^{N_{\text{hf}}^+}$$

# Gaussian model of coherent structures

- Given the field  $U : \mathbb{R}^n \rightarrow \mathbb{R}^d$  and the scalar testing function  $\mathcal{T}(\cdot; U) : \mathbb{R}^n \rightarrow \mathbb{R}$ , the set

$$\mathcal{C}_{\mathcal{T}}(U) := \{x \in \mathbb{R}^n : \mathcal{T}(x; U) > 0\}$$

identifies coherent structures associated with the criterion  $\mathcal{T}$

- For example, if  $U$  is a velocity field and  $\mathcal{T}(x; U) = \|\nabla \times U(x)\|_2 - \tau$  with  $\tau > 0$ ,  $\mathcal{C}_{\mathcal{T}}(U)$  identifies the region where enstrophy exceeds a threshold  $\tau$
- Objective: fit a Gaussian probability density function  $\phi(x; \mu, \Sigma)$  to  $\mathcal{C}_{\mathcal{T}}(U)$

$$\phi(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}.$$

- Define a finite-dimensional discretization of the domain of interest  $P_{\text{hf}} = \{x_i\}_{i=1}^{N_{\text{hf}}}$  and we define

$$P_{\text{hf}}^+ := \{x \in P_{\text{hf}} : \mathcal{T}(x; U) > 0\} = \{y_j\}_{j=1}^{N_{\text{hf}}^+}$$

- Assume that  $\{y_j\}_j$  are IID realizations of a multivariate Gaussian distribution

# Gaussian model of coherent structures

- Estimate mean and variance by MLE:

$$\left\{ \begin{array}{l} \mu_{\text{mle}}[U] = \frac{1}{N_{\text{hf}}^+} \sum_{j=1}^{N_{\text{hf}}^+} y_j, \\ \Sigma_{\text{mle}}[U] = \frac{1}{N_{\text{hf}}^+} \sum_{j=1}^{N_{\text{hf}}^+} (y_j - \mu_{\text{mle}}[U]) (y_j - \mu_{\text{mle}}[U])^T \end{array} \right.$$

# Gaussian model of coherent structures

- Estimate mean and variance by MLE:

$$\left\{ \begin{array}{l} \mu_{\text{mle}}[U] = \frac{1}{N_{\text{hf}}^+} \sum_{j=1}^{N_{\text{hf}}^+} y_j, \\ \Sigma_{\text{mle}}[U] = \frac{1}{N_{\text{hf}}^+} \sum_{j=1}^{N_{\text{hf}}^+} (y_j - \mu_{\text{mle}}[U]) (y_j - \mu_{\text{mle}}[U])^T \end{array} \right.$$

- Given the Gaussian densities  $\pi = \phi(\cdot; \mu_0, \Sigma_0)$  and  $\nu = \phi(\cdot; \mu_1, \Sigma_1)$ , the displacement interpolant  $\hat{\phi}_s$  is Gaussian for all  $s \in [0, 1]$  with mean and covariance given by

$$\mu_s = (1-s) \mu_0 + s \mu_1, \quad \Sigma_s = \Sigma_0^{-1/2} \left( (1-s) \Sigma_0 + s \left( \Sigma_0^{1/2} \Sigma_1 \Sigma_0^{1/2} \right)^{1/2} \right)^2 \Sigma_0^{-1/2}$$

# Gaussian model of coherent structures

- Estimate mean and variance by MLE:

$$\begin{cases} \mu_{\text{mle}}[U] = \frac{1}{N_{\text{hf}}^+} \sum_{j=1}^{N_{\text{hf}}^+} y_j, \\ \Sigma_{\text{mle}}[U] = \frac{1}{N_{\text{hf}}^+} \sum_{j=1}^{N_{\text{hf}}^+} (y_j - \mu_{\text{mle}}[U]) (y_j - \mu_{\text{mle}}[U])^T \end{cases}$$

- Given the Gaussian densities  $\pi = \phi(\cdot; \mu_0, \Sigma_0)$  and  $\nu = \phi(\cdot; \mu_1, \Sigma_1)$ , the displacement interpolant  $\hat{\phi}_s$  is Gaussian for all  $s \in [0, 1]$  with mean and covariance given by

$$\mu_s = (1-s)\mu_0 + s\mu_1, \quad \Sigma_s = \Sigma_0^{-1/2} \left( (1-s)\Sigma_0 + s \left( \Sigma_0^{1/2} \Sigma_1 \Sigma_0^{1/2} \right)^{1/2} \right)^2 \Sigma_0^{-1/2}$$

- The forward and backward mappings  $T_g$  and  $W_g$  are hence available in closed form:

$$T_g(s, \xi) = (1-s)\xi + s \left( \mu_1 + \Sigma_0^{-1/2} \left( \Sigma_0^{1/2} \Sigma_1 \Sigma_0^{1/2} \right)^{1/2} \Sigma_0^{-1/2} (\xi - \mu_0) \right)$$

## Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$

## Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$
- $\Omega \subset \mathbb{R}^d$  is the open computational domain

## Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$
- $\Omega \subset \mathbb{R}^d$  is the open computational domain
- the parametric field of interest  $u_\mu : \Omega \times \mathcal{P} \rightarrow \mathbb{R}^D$

## Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$
- $\Omega \subset \mathbb{R}^d$  is the open computational domain
- the parametric field of interest  $u_\mu : \Omega \times \mathcal{P} \rightarrow \mathbb{R}^D$
- the solution manifold  $\mathcal{M} = \{u_\mu := u(\cdot; \mu) : \mu \in \mathcal{P}\}$

# Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$
- $\Omega \subset \mathbb{R}^d$  is the open computational domain
- the parametric field of interest  $u_\mu : \Omega \times \mathcal{P} \rightarrow \mathbb{R}^D$
- the solution manifold  $\mathcal{M} = \{u_\mu := u(\cdot; \mu) : \mu \in \mathcal{P}\}$
- the training set  $\mathcal{P}_{\text{train}} = \{\mu^k\}_{k=1}^{n_{\text{train}}} \subset \mathcal{P}$

# Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$
- $\Omega \subset \mathbb{R}^d$  is the open computational domain
- the parametric field of interest  $u_\mu : \Omega \times \mathcal{P} \rightarrow \mathbb{R}^D$
- the solution manifold  $\mathcal{M} = \{u_\mu := u(\cdot; \mu) : \mu \in \mathcal{P}\}$
- the training set  $\mathcal{P}_{\text{train}} = \{\mu^k\}_{k=1}^{n_{\text{train}}} \subset \mathcal{P}$
- the dataset of solutions  $\mathcal{D}_{\text{train}} = \{u_\mu : \mu \in \mathcal{P}_{\text{train}}\}$

# Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$
- $\Omega \subset \mathbb{R}^d$  is the open computational domain
- the parametric field of interest  $u_\mu : \Omega \times \mathcal{P} \rightarrow \mathbb{R}^D$
- the solution manifold  $\mathcal{M} = \{u_\mu := u(\cdot; \mu) : \mu \in \mathcal{P}\}$
- the training set  $\mathcal{P}_{\text{train}} = \{\mu^k\}_{k=1}^{n_{\text{train}}} \subset \mathcal{P}$
- the dataset of solutions  $\mathcal{D}_{\text{train}} = \{u_\mu : \mu \in \mathcal{P}_{\text{train}}\}$
- Set of  $\kappa$  nearest neighbors to  $\mu$ :  $\mathcal{P}_{\text{nn}}^\mu = \{\nu^i\}_{i=1}^\kappa \subset \mathcal{P}_{\text{train}}$

# Generalization: notation

- $\mu$ : the vector of model parameters in the region  $\mathcal{P} \subset \mathbb{R}^p$
- $\Omega \subset \mathbb{R}^d$  is the open computational domain
- the parametric field of interest  $u_\mu : \Omega \times \mathcal{P} \rightarrow \mathbb{R}^D$
- the solution manifold  $\mathcal{M} = \{u_\mu := u(\cdot; \mu) : \mu \in \mathcal{P}\}$
- the training set  $\mathcal{P}_{\text{train}} = \{\mu^k\}_{k=1}^{n_{\text{train}}} \subset \mathcal{P}$
- the dataset of solutions  $\mathcal{D}_{\text{train}} = \{u_\mu : \mu \in \mathcal{P}_{\text{train}}\}$
- Set of  $\kappa$  nearest neighbors to  $\mu$ :  $\mathcal{P}_{\text{nn}}^\mu = \{\nu^i\}_{i=1}^\kappa \subset \mathcal{P}_{\text{train}}$

CDI is now:

$$\hat{u}_\mu = \sum_{\nu \in \mathcal{P}_{\text{nn}}^\mu} \omega_\mu^\nu \tilde{u}_\nu, \quad \text{where } \tilde{u}_\nu = u_\nu \circ \Phi_\nu,$$

for a proper choice of the weights  $\{\omega_\mu^\nu : \nu \in \mathcal{P}_{\text{nn}}^\mu\}$  and mappings  $\Phi_\nu$  as found next.

---

**Algorithm** : offline/online decomposition.

---

## Offline stage

*performed once*

- 1: Generate the dataset  $\mathcal{D}_{\text{train}} = \{u_\mu : \mu \in \mathcal{P}_{\text{train}}\}$ .
- 2: Identify the point clouds  $\{X_\mu^{\text{raw}} : \mu \in \mathcal{P}_{\text{train}}\}$ .
- 3: Define the template set  $X^{\text{ref}}$  and the sorted point clouds  $\{X_\mu : \mu \in \mathcal{P}_{\text{train}}\}$ .

# Generalization: Algorithm

---

**Algorithm** : offline/online decomposition.

---

## Offline stage

*performed once*

- 1: Generate the dataset  $\mathcal{D}_{\text{train}} = \{u_\mu : \mu \in \mathcal{P}_{\text{train}}\}$ .
- 2: Identify the point clouds  $\{X_\mu^{\text{raw}} : \mu \in \mathcal{P}_{\text{train}}\}$ .
- 3: Define the template set  $X^{\text{ref}}$  and the sorted point clouds  $\{X_\mu : \mu \in \mathcal{P}_{\text{train}}\}$ .

## Online stage

*performed for any  $\mu \in \mathcal{P}$*

- 1: Estimate the new points  $\hat{X}_\mu = \{\hat{x}_{i,\mu}\}_{i=1}^N$ .
- 2: Select the neighboring parameters  $\mathcal{P}_{\text{nn}}^\mu = \{\nu^i\}_{i=1}^{\kappa} \subset \mathcal{P}_{\text{train}}$ .
- 3: Compute the mappings  $\Phi_\nu$  based on  $\hat{X}_\mu$  and  $\hat{X}_\nu$  for all  $\nu \in \mathcal{P}_{\text{nn}}^\mu$ .

- 4: Compute the weights  $\{\psi_i : \nu_i \in \mathcal{P}_{\text{nn}}^\mu\}$  and return the estimate

# Sensor (or Feature) Selection

- Define a problem-dependent scalar testing function  $\mathcal{T}$  & consider a discrete set of test points  $P_{\text{hf}} \subset \overline{\Omega}$ .

# Sensor (or Feature) Selection

- Define a problem-dependent scalar testing function  $\mathcal{T}$  & consider a discrete set of test points  $P_{\text{hf}} \subset \overline{\Omega}$ .
- Compute the following for all  $\mu \in \mathcal{P}_{\text{train}}$ :

$$X_{\mu}^{\text{raw}} = \left\{ x \in P_{\text{hf}} : \mathcal{T}(x, u_{\mu}^{\text{hf}}) \geq t_{\mu} \right\}$$

The threshold  $t_{\mu}$  is chosen equal to the  $\gamma_{\text{thr}}$  quantile over the training set:

$$t_{\mu} = \text{quantile} \left( \{ \mathcal{T}(x, u_{\mu}^{\text{hf}}) : x \in P_{\text{hf}} \}, \gamma_{\text{thr}} \right)$$

# Sensor (or Feature) Selection

- Define a problem-dependent scalar testing function  $\mathcal{T}$  & consider a discrete set of test points  $P_{\text{hf}} \subset \overline{\Omega}$ .
- Compute the following for all  $\mu \in \mathcal{P}_{\text{train}}$ :

$$X_{\mu}^{\text{raw}} = \left\{ x \in P_{\text{hf}} : \mathcal{T}(x, u_{\mu}^{\text{hf}}) \geq t_{\mu} \right\}$$

The threshold  $t_{\mu}$  is chosen equal to the  $\gamma_{\text{thr}}$  quantile over the training set:

$$t_{\mu} = \text{quantile} \left( \{ \mathcal{T}(x, u_{\mu}^{\text{hf}}) : x \in P_{\text{hf}} \}, \gamma_{\text{thr}} \right)$$

- $\mathcal{T}$ : for compressible flow problems, as seen, Ducros sensor.

# Sensor (or Feature) Selection

- Define a problem-dependent scalar testing function  $\mathcal{T}$  & consider a discrete set of test points  $P_{\text{hf}} \subset \overline{\Omega}$ .
- Compute the following for all  $\mu \in \mathcal{P}_{\text{train}}$ :

$$X_{\mu}^{\text{raw}} = \left\{ x \in P_{\text{hf}} : \mathcal{T}(x, u_{\mu}^{\text{hf}}) \geq \mathfrak{t}_{\mu} \right\}$$

The threshold  $\mathfrak{t}_{\mu}$  is chosen equal to the  $\gamma_{\text{thr}}$  quantile over the training set:

$$\mathfrak{t}_{\mu} = \text{quantile} \left( \left\{ \mathcal{T}(x, u_{\mu}^{\text{hf}}) : x \in P_{\text{hf}} \right\}, \gamma_{\text{thr}} \right)$$

- $\mathcal{T}$ : for compressible flow problems, as seen, Ducros sensor.
- For 2D recirculating channel flows, we rely on the isolines of the streamfunction:

$$\Psi_{\mu}(x = [x_1, x_2]) = \int_{y_{\text{btm}}(x_1)}^{x_2} (u_{\mu}(x_1, s))_1 ds,$$

where  $y_{\text{btm}}(x_1) = \inf\{x_2 : [x_1, x_2] \in \Omega\}$ ; then,  $\mathcal{T}(x, u_{\mu}^{\text{hf}}) = -\Psi_{\mu}(x)$  and  $\mathfrak{t}_{\mu} \equiv 0$ ; that is,

- The raw point clouds  $X_\mu^{\text{raw}}$  are typically not of the same size and are not sorted.
- Point Set Registration (PSR) is used to find matched point clouds:

$$X_\mu = \text{PSR} \left( X^{\text{ref}}, X_\mu^{\text{raw}} \right)$$

- This involves finding a map  $T : \mathbb{R}^d \rightarrow \mathbb{R}^d$  that minimizes the distance between point clouds.
- The distance is defined as:

$$\text{dist} (Y, T(X)) = \max_{y \in Y} \left( \min_{x \in X} \|y - T(x)\|_2 \right)$$

- Gaussian-based PSR utilizes maximum likelihood estimation (MLE) to estimate the mean and covariance matrix:

$$\mu_X = \frac{1}{N} \sum_{i=1}^N x_i, \quad \Sigma_X = \frac{1}{N} \sum_{i=1}^N (x_i - \mu_X)(x_i - \mu_X)^\top$$

- The optimal transport map  $T_{X,Y}$  is defined as:

$$T_{X,Y}(x) = \mu_Y + \Sigma_X^{-1/2} \left( \Sigma_X^{1/2} \Sigma_Y \Sigma_X^{1/2} \right)^{1/2} \Sigma_X^{-1/2} (x - \mu_X)$$

- The deformed point cloud  $\tilde{X}$  is then defined as  $\tilde{X} = \{\tilde{x}_i = T_{X,Y}(x_i)\}_{i=1}^N$ .

# Regression of Sorted Point Clouds

- For approximating the mapping  $\mu \mapsto \hat{X}_\mu$ , Radial Basis Function (RBF) regression is used.
- Proper Orthogonal Decomposition (POD) is then applied for an equivalent representation:

$$X_{\mu^k} = \sum_{i=1}^M Z_i \beta_k^i$$

- Here,  $Z_1, \dots, Z_M \in \mathbb{R}^{N \times d}$  are the basis vectors and  $\beta_1^1, \dots, \beta_{n_{\text{train}}}^M \in \mathbb{R}$  are the coefficients.

# Regression of Sorted Point Clouds

- For approximating the mapping  $\mu \mapsto \hat{X}_\mu$ , Radial Basis Function (RBF) regression is used.
- Proper Orthogonal Decomposition (POD) is then applied for an equivalent representation:

$$X_{\mu^k} = \sum_{i=1}^M Z_i \beta_k^i$$

- Here,  $Z_1, \dots, Z_M \in \mathbb{R}^{N \times d}$  are the basis vectors and  $\beta_1^1, \dots, \beta_{n_{\text{train}}}^M \in \mathbb{R}$  are the coefficients.
- The coefficients  $\hat{\beta}^i$  are classically estimated by minimizing:

$$\hat{\beta}^i = \arg \min_{\beta \in \mathfrak{H}_\phi} \lambda \|\beta\|_{\mathfrak{H}_\phi}^2 + \sum_{k=1}^{n_{\text{train}}} \left( \beta(\mu^k) - \beta_k^i \right)^2$$

$\mathfrak{H}_\phi$  denotes the native space associated with the kernel  $\phi$  and  $\lambda > 0$  is a regularization coefficient.

# Regression of Sorted Point Clouds

- For approximating the mapping  $\mu \mapsto \hat{X}_\mu$ , Radial Basis Function (RBF) regression is used.
- Proper Orthogonal Decomposition (POD) is then applied for an equivalent representation:

$$X_{\mu^k} = \sum_{i=1}^M Z_i \beta_k^i$$

- Here,  $Z_1, \dots, Z_M \in \mathbb{R}^{N \times d}$  are the basis vectors and  $\beta_1^1, \dots, \beta_{n_{\text{train}}}^M \in \mathbb{R}$  are the coefficients.
- The coefficients  $\hat{\beta}^i$  are classically estimated by minimizing:

$$\hat{\beta}^i = \arg \min_{\beta \in \mathfrak{H}_\phi} \lambda \|\beta\|_{\mathfrak{H}_\phi}^2 + \sum_{k=1}^{n_{\text{train}}} \left( \beta(\mu^k) - \beta_k^i \right)^2$$

$\mathfrak{H}_\phi$  denotes the native space associated with the kernel  $\phi$  and  $\lambda > 0$  is a regularization coefficient.

- We finally have the parameterized new points estimation:

$$\hat{\mu} \mapsto \hat{X}_{\hat{\mu}} = \sum_{i=1}^M \hat{\beta}^i Z_i$$

# Choice of Nearest Neighbors

- Select parameters  $\nu^1, \dots, \nu^\kappa$  that minimize the Euclidean distance to  $\mu$ :

$$\text{dist}(\mu, \nu) = \|\mu - \nu\|_2$$

- The weights for the neighbors are then defined as:

$$\omega_\mu^\nu = \frac{\bar{\omega}_\mu^\nu}{\sum_{\nu' \in \mathcal{P}_{\text{nn}}^\mu} \bar{\omega}_\mu^{\nu'}}, \quad \bar{\omega}_\mu^{\nu'} = \frac{1}{\text{dist}^p(\mu, \nu')}$$

- Inverse distance weighting (IDW) is used with  $p = 2$ .

An alternative approach consists in formulating the problem of registration as a minimization problem of the form

$$\min_{\Phi \in \mathcal{W}_\Omega} \frac{1}{N} \sum_{i=1}^N \|\Phi(x_i^{\text{ref}}) - \hat{x}_{i,\mu}\|_2^2 + \mathfrak{P}(\Phi),$$

that can be solved using a gradient-based (quasi-Newton) method.

An alternative approach consists in formulating the problem of registration as a minimization problem of the form

$$\min_{\Phi \in \mathcal{W}_\Omega} \frac{1}{N} \sum_{i=1}^N \|\Phi(x_i^{\text{ref}}) - \hat{x}_{i,\mu}\|_2^2 + \mathfrak{P}(\Phi),$$

that can be solved using a gradient-based (quasi-Newton) method.

The optimization statement depends on the choice of the penalty term  $\mathfrak{P}$  and of the search space  $\mathcal{W}_\Omega$ : the former should enforce

- local bijectivity  $\det(\nabla \Phi) > 0$
- promote the smoothness (in a Sobolev sense)
- $\Phi(\Omega) = \Omega$
- ...

- **Nonlinear approximation manifold generated by a RB and an ANN**

$$\tilde{u} = u_{\text{ref}} + \mathbf{V}q + \overline{\mathbf{V}}\mathcal{N}(q)$$

- **Nonlinear approximation manifold generated by a RB and an ANN**

$$\tilde{u} = u_{\text{ref}} + \mathbf{V}q + \overline{\mathbf{V}}\mathcal{N}(q)$$

- $\mathbf{V} \in \mathbb{R}^{N \times n}$  is constructed using the first  $n \ll N$  columns of  $\mathbf{U}$

- **Nonlinear approximation manifold generated by a RB and an ANN**

$$\tilde{u} = u_{\text{ref}} + \mathbf{V}q + \overline{\mathbf{V}}\mathcal{N}(q)$$

- $\mathbf{V} \in \mathbb{R}^{N \times n}$  is constructed using the first  $n \ll N$  columns of  $\mathbf{U}$
- $\overline{\mathbf{V}} \in \mathbb{R}^{N \times \bar{n}}$  is constructed using a subset of the next  $\bar{n} \ll N$  columns of  $\mathbf{U}$ ; and  $\bar{n} \gg n$

- **Nonlinear approximation manifold generated by a RB and an ANN**

$$\tilde{u} = u_{\text{ref}} + \mathbf{V}q + \overline{\mathbf{V}}\mathcal{N}(q)$$

- $\mathbf{V} \in \mathbb{R}^{N \times n}$  is constructed using the first  $n \ll N$  columns of  $\mathbf{U}$
- $\overline{\mathbf{V}} \in \mathbb{R}^{N \times \bar{n}}$  is constructed using a subset of the next  $\bar{n} \ll N$  columns of  $\mathbf{U}$ ; and  $\bar{n} \gg n$
- $\mathbf{V}$  and  $\overline{\mathbf{V}}$  satisfy the orthogonality properties

$$\mathbf{V}^T \mathbf{V} = \mathbf{I}_n, \quad \overline{\mathbf{V}}^T \overline{\mathbf{V}} = \mathbf{I}_{\bar{n}}, \quad \mathbf{V}^T \overline{\mathbf{V}} = 0, \quad \overline{\mathbf{V}}^T \mathbf{V} = 0$$

- **Nonlinear approximation manifold generated by a RB and an ANN**

$$\tilde{u} = u_{\text{ref}} + \mathbf{V}q + \overline{\mathbf{V}}\mathcal{N}(q)$$

- $\mathbf{V} \in \mathbb{R}^{N \times n}$  is constructed using the first  $n \ll N$  columns of  $\mathbf{U}$
- $\overline{\mathbf{V}} \in \mathbb{R}^{N \times \bar{n}}$  is constructed using a subset of the next  $\bar{n} \ll N$  columns of  $\mathbf{U}$ ; and  $\bar{n} \gg n$
- $\mathbf{V}$  and  $\overline{\mathbf{V}}$  satisfy the orthogonality properties

$$\mathbf{V}^T \mathbf{V} = \mathbf{I}_n, \quad \overline{\mathbf{V}}^T \overline{\mathbf{V}} = \mathbf{I}_{\bar{n}}, \quad \mathbf{V}^T \overline{\mathbf{V}} = 0, \quad \overline{\mathbf{V}}^T \mathbf{V} = 0$$

- $\mathcal{N}$  is an ANN representing a map  $\mathbb{R}^n \rightarrow \mathbb{R}^{\bar{n}}$  whose size  $n_{\text{ANN}}$  scales with  $\bar{n} \ll N$

# Offline Training of the ANN Representing the Map $f(q)$

- Let  $\mathcal{N}(q; \gamma)$  be the ANN representing the map  $f(q) : \mathbb{R}^n \rightarrow \mathbb{R}^{\bar{n}}$ , where the vector-valued hyperparameter  $\gamma \in \mathbb{R}^{n_{ANN}}$ .

# Offline Training of the ANN Representing the Map $f(q)$

- Let  $\mathcal{N}(q; \gamma)$  be the ANN representing the map  $f(q) : \mathbb{R}^n \rightarrow \mathbb{R}^{\bar{n}}$ , where the vector-valued hyperparameter  $\gamma \in \mathbb{R}^{n_{ANN}}$ .
- Construct  $\mathcal{N}(q; \gamma)$  such that ideally

$$u_i = u_{ref} + \mathbf{V}q_i + \overline{\mathbf{V}}\mathcal{N}(q_i; \gamma), \quad i = 1, \dots, N_s$$

# Offline Training of the ANN Representing the Map $f(q)$

- Let  $\mathcal{N}(q; \gamma)$  be the ANN representing the map  $f(q) : \mathbb{R}^n \rightarrow \mathbb{R}^{\bar{n}}$ , where the vector-valued hyperparameter  $\gamma \in \mathbb{R}^{n_{ANN}}$ .
- Construct  $\mathcal{N}(q; \gamma)$  such that ideally

$$u_i = u_{ref} + \mathbf{V}q_i + \bar{\mathbf{V}}\mathcal{N}(q_i; \gamma), \quad i = 1, \dots, N_s$$

- From the above and the orthogonality properties of  $V$  and  $\bar{V}$ , it follows that

$$q_i = V^T(u_i - u_{ref}) \quad \text{and} \quad \mathcal{N}(q_i; \gamma) \equiv \bar{q}_i, \quad i = 1, \dots, N_s$$

# Offline Training of the ANN Representing the Map $f(q)$

- Let  $\mathcal{N}(q; \gamma)$  be the ANN representing the map  $f(q) : \mathbb{R}^n \rightarrow \mathbb{R}^{\bar{n}}$ , where the vector-valued hyperparameter  $\gamma \in \mathbb{R}^{n_{ANN}}$ .
- Construct  $\mathcal{N}(q; \gamma)$  such that ideally

$$u_i = u_{ref} + \mathbf{V}q_i + \bar{\mathbf{V}}\mathcal{N}(q_i; \gamma), \quad i = 1, \dots, N_s$$

- From the above and the orthogonality properties of  $V$  and  $\bar{V}$ , it follows that

$$q_i = V^T(u_i - u_{ref}) \quad \text{and} \quad \mathcal{N}(q_i; \gamma) \equiv \bar{q}_i, \quad i = 1, \dots, N_s$$

- Hence

$$q \text{ (input)} \rightarrow \mathcal{N}(q; \gamma) \rightarrow \bar{q} \text{ (output)}$$

# Offline Training of the ANN Representing the Map $f(q)$

- Let  $\mathcal{N}(q; \gamma)$  be the ANN representing the map  $f(q) : \mathbb{R}^n \rightarrow \mathbb{R}^{\bar{n}}$ , where the vector-valued hyperparameter  $\gamma \in \mathbb{R}^{n_{ANN}}$ .
- Construct  $\mathcal{N}(q; \gamma)$  such that ideally

$$u_i = u_{ref} + \mathbf{V}q_i + \bar{\mathbf{V}}\mathcal{N}(q_i; \gamma), \quad i = 1, \dots, N_s$$

- From the above and the orthogonality properties of  $V$  and  $\bar{V}$ , it follows that

$$q_i = V^T(u_i - u_{ref}) \quad \text{and} \quad \mathcal{N}(q_i; \gamma) \equiv \bar{q}_i, \quad i = 1, \dots, N_s$$

- Hence

$$q \text{ (input)} \rightarrow \mathcal{N}(q; \gamma) \rightarrow \bar{q} \text{ (output)}$$

•

$$\gamma = \arg \min_{\gamma'} \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} (\bar{q}_i - \mathcal{N}(q_i; \gamma'))^2$$

## Part 2: Sampling the Parameter Space

# Parameter Sample Selection for Computing Snapshots

- How one chooses the  $s$  parameter samples  $\mu^1, \dots, \mu^s$  where to compute the snapshots  $\{\mathbf{w}(\mu^1), \dots, \mathbf{w}(\mu^s)\}$ ?

# Parameter Sample Selection for Computing Snapshots

- How one chooses the  $s$  parameter samples  $\mu^1, \dots, \mu^s$  where to compute the snapshots  $\{\mathbf{w}(\mu^1), \dots, \mathbf{w}(\mu^s)\}$ ?
  - The location of the samples in the parameter space will determine the accuracy of the resulting global ROM in the entire parameter domain  $\mathcal{D} \subset \mathbb{R}^d$

# Parameter Sample Selection for Computing Snapshots

- How one chooses the  $s$  parameter samples  $\mu^1, \dots, \mu^s$  where to compute the snapshots  $\{\mathbf{w}(\mu^1), \dots, \mathbf{w}(\mu^s)\}$ ?
  - The location of the samples in the parameter space will determine the accuracy of the resulting global ROM in the entire parameter domain  $\mathcal{D} \subset \mathbb{R}^d$
- Possible approaches:

# Parameter Sample Selection for Computing Snapshots

- How one chooses the  $s$  parameter samples  $\mu^1, \dots, \mu^s$  where to compute the snapshots  $\{\mathbf{w}(\mu^1), \dots, \mathbf{w}(\mu^s)\}$ ?
  - The location of the samples in the parameter space will determine the accuracy of the resulting global ROM in the entire parameter domain  $\mathcal{D} \subset \mathbb{R}^d$
- Possible approaches:
  - Uniform sampling for parameter spaces of moderate dimensions and moderately computationally intensive High-Dimensional Models (HDMs)

# Parameter Sample Selection for Computing Snapshots

- How one chooses the  $s$  parameter samples  $\mu^1, \dots, \mu^s$  where to compute the snapshots  $\{\mathbf{w}(\mu^1), \dots, \mathbf{w}(\mu^s)\}$ ?
  - The location of the samples in the parameter space will determine the accuracy of the resulting global ROM in the entire parameter domain  $\mathcal{D} \subset \mathbb{R}^d$
- Possible approaches:
  - Uniform sampling for parameter spaces of moderate dimensions and moderately computationally intensive High-Dimensional Models (HDMs)
  - Latin Hypercube Sampling (LHS) for higher-dimensional parameter spaces and moderately computationally intensive HDMs

# Parameter Sample Selection for Computing Snapshots

- How one chooses the  $s$  parameter samples  $\mu^1, \dots, \mu^s$  where to compute the snapshots  $\{\mathbf{w}(\mu^1), \dots, \mathbf{w}(\mu^s)\}$ ?
  - The location of the samples in the parameter space will determine the accuracy of the resulting global ROM in the entire parameter domain  $\mathcal{D} \subset \mathbb{R}^d$
- Possible approaches:
  - Uniform sampling for parameter spaces of moderate dimensions and moderately computationally intensive High-Dimensional Models (HDMs)
  - Latin Hypercube Sampling (LHS) for higher-dimensional parameter spaces and moderately computationally intensive HDMs
  - Adaptive, goal-oriented, greedy sampling that exploits an error indicator to focus on the PROM accuracy, for higher-dimensional parameter spaces and computationally intensive HDMs

# Latin Hypercube Sampling (LHS)

## Key Idea:

- Divide each parameter range into  $M$  non-overlapping intervals of equal probability.

## Algorithm:

- 1 For each parameter  $\mu_i \in [\mu_i^{\min}, \mu_i^{\max}]$ ,  $1 \leq i \leq d$ , divide the range into  $M$  intervals:

$$\mu_i^{\min} = \mu_i^1 < \mu_i^2 < \dots < \mu_i^M = \mu_i^{\max}.$$

- 2 Randomly sample one value from each interval.
- 3 Combine these samples randomly across all dimensions to generate  $M$  points:

$$\{\mu^1, \mu^2, \dots, \mu^M\}, \quad \mu^j \in \mathcal{P}.$$

## Remarks:

- $M$  independent of the number of dimensions of the parameter space.

# Latin Hypercube Sampling (LHS)

## Key Idea:

- Divide each parameter range into  $M$  non-overlapping intervals of equal probability.
- Sample exactly one point from each interval for each parameter.

## Algorithm:

- 1 For each parameter  $\mu_i \in [\mu_i^{\min}, \mu_i^{\max}]$ ,  $1 \leq i \leq d$ , divide the range into  $M$  intervals:

$$\mu_i^{\min} = \mu_i^1 < \mu_i^2 < \dots < \mu_i^M = \mu_i^{\max}.$$

- 2 Randomly sample one value from each interval.
- 3 Combine these samples randomly across all dimensions to generate  $M$  points:

$$\{\mu^1, \mu^2, \dots, \mu^M\}, \quad \mu^j \in \mathcal{P}.$$

## Remarks:

- $M$  independent of the number of dimensions of the parameter space.

# Latin Hypercube Sampling (LHS)

## Key Idea:

- Divide each parameter range into  $M$  non-overlapping intervals of equal probability.
- Sample exactly one point from each interval for each parameter.
- Ensure no two samples occupy the same interval along any dimension.

## Algorithm:

- 1 For each parameter  $\mu_i \in [\mu_i^{\min}, \mu_i^{\max}]$ ,  $1 \leq i \leq d$ , divide the range into  $M$  intervals:

$$\mu_i^{\min} = \mu_i^1 < \mu_i^2 < \dots < \mu_i^M = \mu_i^{\max}.$$

- 2 Randomly sample one value from each interval.
- 3 Combine these samples randomly across all dimensions to generate  $M$  points:

$$\{\mu^1, \mu^2, \dots, \mu^M\}, \quad \mu^j \in \mathcal{P}.$$

## Remarks:

- $M$  independent of the number of dimensions of the parameter space.

# Greedy approach

- Ideally, one can build a ROM progressively and update it (increase its dimension) by considering additional samples  $\mu^{(i)}$  and corresponding solution snapshots at the locations of the parameter space where the current ROM is the most inaccurate:

$$\mu^i = \arg \max_{\mu \in \mathcal{D}} \|\mathcal{E}_{\text{PROM}}(\mu)\| = \arg \max_{\mu \in \mathcal{D}} \|\mathbf{w}(\mu) - \mathbf{V}\mathbf{q}(\mu)\|$$

# Greedy approach

- Ideally, one can build a ROM progressively and update it (increase its dimension) by considering additional samples  $\mu^{(i)}$  and corresponding solution snapshots at the locations of the parameter space where the current ROM is the most inaccurate:

$$\mu^i = \arg \max_{\mu \in \mathcal{D}} \|\mathcal{E}_{\text{PROM}}(\mu)\| = \arg \max_{\mu \in \mathcal{D}} \|\mathbf{w}(\mu) - \mathbf{V}\mathbf{q}(\mu)\|$$

- $\mathbf{q}(\mu)$  can be efficiently computed

# Greedy approach

- Ideally, one can build a ROM progressively and update it (increase its dimension) by considering additional samples  $\mu^{(i)}$  and corresponding solution snapshots at the locations of the parameter space where the current ROM is the most inaccurate:

$$\mu^i = \arg \max_{\mu \in \mathcal{D}} \|\mathcal{E}_{\text{PROM}}(\mu)\| = \arg \max_{\mu \in \mathcal{D}} \|\mathbf{w}(\mu) - \mathbf{V}\mathbf{q}(\mu)\|$$

- $\mathbf{q}(\mu)$  can be efficiently computed
- But the cost of obtaining  $\mathbf{w}(\mu)$  can be high  $\Rightarrow$  eventually an intractable approach

# Greedy approach

- Ideally, one can build a ROM progressively and update it (increase its dimension) by considering additional samples  $\mu^{(i)}$  and corresponding solution snapshots at the locations of the parameter space where the current ROM is the most inaccurate:

$$\mu^i = \arg \max_{\mu \in \mathcal{D}} \|\mathcal{E}_{\text{PROM}}(\mu)\| = \arg \max_{\mu \in \mathcal{D}} \|\mathbf{w}(\mu) - \mathbf{V}\mathbf{q}(\mu)\|$$

- $\mathbf{q}(\mu)$  can be efficiently computed
- But the cost of obtaining  $\mathbf{w}(\mu)$  can be high  $\Rightarrow$  eventually an intractable approach
- Idea: rely on an economical a posteriori error estimator/indicator

# Greedy approach

- Ideally, one can build a ROM progressively and update it (increase its dimension) by considering additional samples  $\mu^{(i)}$  and corresponding solution snapshots at the locations of the parameter space where the current ROM is the most inaccurate:

$$\mu^i = \arg \max_{\mu \in \mathcal{D}} \|\mathcal{E}_{\text{PROM}}(\mu)\| = \arg \max_{\mu \in \mathcal{D}} \|\mathbf{w}(\mu) - \mathbf{V}\mathbf{q}(\mu)\|$$

- $\mathbf{q}(\mu)$  can be efficiently computed
- But the cost of obtaining  $\mathbf{w}(\mu)$  can be high  $\Rightarrow$  eventually an intractable approach
- Idea: rely on an economical a posteriori error estimator/indicator
  - Error indicator based on the norm of the (affordable) residual

$$\|r(\mu)\| = \|f(\mathbf{V}\mathbf{q}(\mu); \mu)\|$$

# Greedy approach

- Ideally, one can build a ROM progressively and update it (increase its dimension) by considering additional samples  $\mu^{(i)}$  and corresponding solution snapshots at the locations of the parameter space where the current ROM is the most inaccurate:

$$\mu^i = \arg \max_{\mu \in \mathcal{D}} \|\mathcal{E}_{\text{PROM}}(\mu)\| = \arg \max_{\mu \in \mathcal{D}} \|\mathbf{w}(\mu) - \mathbf{V}\mathbf{q}(\mu)\|$$

- $\mathbf{q}(\mu)$  can be efficiently computed
- But the cost of obtaining  $\mathbf{w}(\mu)$  can be high  $\Rightarrow$  eventually an intractable approach
- Idea: rely on an economical a posteriori error estimator/indicator
  - Error indicator based on the norm of the (affordable) residual

$$\|r(\mu)\| = \|f(\mathbf{V}\mathbf{q}(\mu); \mu)\|$$

- For this purpose,  $\mathcal{D}$  is typically replaced by a large discrete set of candidate parameters  $\{\mu^{\star 1}, \dots, \mu^{\star c}\} \subset \mathcal{D}$

# Greedy Procedure Based on Residual Norm

---

## Algorithm Greedy Sampling Algorithm

---

- 1: Randomly select a first sample  $\mu^1$
- 2: Solve the HDM-based problem

$$f(w(\mu^1); \mu^1) = 0$$

- 3: Build a corresponding ROB  $V$
- 4: **for**  $i = 2, \dots$  **do**
- 5:     Solve

$$\mu^{(i)} = \arg \max_{\mu \in \{\mu^*, \dots, \mu^{*c}\}} \|r(\mu)\|$$

- 6:     Solve the HDM-based problem  $f(w(\mu^i); \mu^i) = 0$
- 7:     Build a ROB  $V$  based on the snapshots (or in this case, samples)

$$\{w(\mu^1), \dots, w(\mu^i)\}$$

## Part 3: Solving in the Reduced Space

## Response Surface

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Common basis functions:

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Common basis functions:
  - Polynomial (e.g., linear, quadratic).

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Common basis functions:
  - Polynomial (e.g., linear, quadratic).
  - Radial basis functions.

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Common basis functions:
  - Polynomial (e.g., linear, quadratic).
  - Radial basis functions.
- Steps:

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Common basis functions:
  - Polynomial (e.g., linear, quadratic).
  - Radial basis functions.
- Steps:
  - 1 Sample the parameter space, i.e., chose  $\mu_1, \dots, \mu_{\text{snap}}$ .

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Common basis functions:
  - Polynomial (e.g., linear, quadratic).
  - Radial basis functions.
- Steps:
  - 1 Sample the parameter space, i.e., chose  $\mu_1, \dots, \mu_{\text{snap}}$ .
  - 2 Evaluate the high-fidelity model at sample points.

- Approximate the system response (e.g. RB latent variables) using a surrogate model:

$$\hat{q}(\mu) = \sum_{i=1}^n c_i \kappa_i(\mu),$$

where  $\kappa_i(\mu)$  are basis functions, and  $c_i$  are coefficients. Both depend on the parameter space sampling.

- Common basis functions:
  - Polynomial (e.g., linear, quadratic).
  - Radial basis functions.
- Steps:
  - 1 Sample the parameter space, i.e., chose  $\mu_1, \dots, \mu_{\text{snap}}$ .
  - 2 Evaluate the high-fidelity model at sample points.
  - 3 Fit the surrogate model to the sampled data.

## Projection-based MOR

- Nonlinear High-Dimensional Model:

$$\frac{d\mathbf{w}}{dt} = \mathbf{f}(\mathbf{w}(t), t), \quad \mathbf{y}(t) = \mathbf{g}(\mathbf{w}(t), t).$$

- Initial condition:

$$\mathbf{w}(0) = \mathbf{w}_0.$$

- Variables:

- $\mathbf{w} \in \mathbb{R}^N$ : State vector.
- $\mathbf{y} \in \mathbb{R}^q$ : Output vector, typically  $q \ll N$ .
- Function  $\mathbf{f}$  defines the dynamics.

# Projection-based Reduced-Order Model (PROM)

- The goal is to construct a Projection-based Reduced-Order Model (PROM):

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

$$\mathbf{y}(t) \approx \mathbf{g}_r(\mathbf{q}(t), t)$$

- where:

- $\mathbf{q} \in \mathbb{R}^k$ : Vector of reduced-order state variables,  $k \ll N$
- $\mathbf{y} \in \mathbb{R}^q$ : Vector of output variables
- $\mathbf{f}_r(\cdot, \cdot) \in \mathbb{R}^k$ : reduced dynamics

# Residual of the model

- The residual  $\mathbf{r}(t) \in \mathbb{R}^N$  is introduced by approximating  $\mathbf{w}(t)$  as  $\mathbf{V}\mathbf{q}(t)$ , where  $\mathbf{V} \in \mathbb{R}^{N \times k}$  is the matrix whose columns are the POD modes:

$$\mathbf{V} \frac{d\mathbf{q}}{dt}(t) = \mathbf{f}(\mathbf{V}\mathbf{q}(t), t) + \mathbf{r}(t) \iff \mathbf{r}(t) = \mathbf{V} \frac{d\mathbf{q}}{dt}(t) - \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

- Constrain this residual to be orthogonal to a subspace  $\mathcal{W}$  defined by a test basis  $\mathbf{W} \in \mathbb{R}^{N \times k}$  – that is, compute  $\mathbf{q}(t)$  such that

$$\mathbf{W}^T \mathbf{r}(t) = 0$$

- This leads to the governing equations of the Petrov-Galerkin PROM

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

- If  $\mathbf{W} = \mathbf{V}$  the projection method is called a Galerkin PROM

# Characterizing the Error of PROM Solutions

- Error of the solution computed using a PROM relative to the solution obtained using the HDM:

$$\mathcal{E}_{\text{PROM}}(t) = \mathbf{w}(t) - \tilde{\mathbf{w}}(t) = \mathbf{w}(t) - \mathbf{V}\mathbf{q}(t)$$

- Assume a Galerkin projection and an associated orthogonal basis  $\mathbf{V}$
- The error vector can be decomposed into two orthogonal components:

$$\mathcal{E}_{\text{PROM}}(t) = \mathbf{w}(t) - \Pi_{\mathbf{V}}\mathbf{w}(t) + \Pi_{\mathbf{V}}\mathbf{w}(t) - \mathbf{V}\mathbf{q}(t)$$

$$= (\mathbf{I}_N - \Pi_{\mathbf{V}})\mathbf{w}(t) + \mathbf{V}(\mathbf{V}^T\mathbf{w}(t) - \mathbf{q}(t))$$

$$= \mathcal{E}_{\mathbf{V}^\perp}(t) + \mathcal{E}_{\mathbf{V}}(t)$$

- Error component orthogonal to  $\mathbf{V}$ :

$$\mathcal{E}_{\mathbf{V}^\perp}(t) = (\mathbf{I}_N - \Pi_{\mathbf{V}})\mathbf{w}(t)$$

- *Interpretation*: The exact trajectory does not strictly belong to  $\text{range}(\mathbf{V}) \Rightarrow$  projection error
- Error component parallel to  $\mathbf{V}$ :

$$\mathcal{E}_{\mathbf{V}}(t) = \mathbf{V}(\mathbf{V}^T \mathbf{w}(t) - \mathbf{q}(t))$$

- *Interpretation*: An "equivalent" but different dynamical system is solved  $\Rightarrow$  modeling error
- Sometime  $\mathcal{E}_{\mathbf{V}^\perp}(t)$  can be computed without executing the PROM and therefore can provide an a priori error estimate

- Consider the case of a Galerkin projection
- One can derive an ODE governing the behavior of the error component lying in  $\mathcal{V}$  in terms of that lying in  $\mathcal{V}^\perp$
- In the case of an **autonomous linear system**

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

the error ODE has the simple form

$$\frac{d\mathcal{E}_{\mathbf{V}}}{dt}(t) = \mathbf{V}\mathbf{V}^T \mathbf{A} (\mathcal{E}_{\mathbf{V}}(t) + \mathcal{E}_{\mathbf{V}^\perp}(t)) = \Pi_{\mathbf{V}} \mathbf{A} (\mathcal{E}_{\mathbf{V}}(t) + \mathcal{E}_{\mathbf{V}^\perp}(t))$$

where  $\mathcal{E}_{\mathbf{V}^\perp}(t)$  acts as a forcing term.

The solution of the ODE is:

$$\mathcal{E}_{\mathbf{V}}(t) = e^{\Pi_{\mathbf{V}} \mathbf{A} t} \mathcal{E}_{\mathbf{V}}(0) + \int_0^t e^{\Pi_{\mathbf{V}} \mathbf{A} (t-\tau)} \Pi_{\mathbf{V}} \mathcal{E}_{\mathbf{V}^\perp}(\tau) d\tau.$$

For  $\mathcal{E}_{\mathbf{V}^\perp} = 0$ , we have the energy estimate of the homogeneous solution:

$$\|\mathcal{E}_{\mathbf{V}}(t)\|_2^2 = \mathcal{E}_{\mathbf{V}}(0)^\top e^{(\Pi_{\mathbf{V}} \mathbf{A})^\top t} e^{\Pi_{\mathbf{V}} \mathbf{A} t} \mathcal{E}_{\mathbf{V}}(0).$$

**Bounds:**

- Define  $\mathbf{M}(t) = e^{(\Pi_{\mathbf{V}} \mathbf{A})^\top t} e^{\Pi_{\mathbf{V}} \mathbf{A} t}$ .
- Energy satisfies:

$$\|\mathcal{E}_{\mathbf{V}}(t)\|_2^2 \leq \|e^{\Pi_{\mathbf{V}} \mathbf{A} t}\|_2^2 \|\mathcal{E}_{\mathbf{V}}(0)\|_2^2,$$

where  $\|e^{\Pi_{\mathbf{V}} \mathbf{A} t}\|_2^2 = \lambda_{\max}(\mathbf{M}(t))$  and  $\lambda_{\max}$  is the largest eigenvalue of  $\mathbf{M}(t)$ .

# Energy of the Inhomogeneous Solution

For  $\mathcal{E}_{\mathbf{V}^\perp} \neq 0$ , we have

$$\left\| \int_0^t e^{\Pi_{\mathbf{V}} \mathbf{A}(t-\tau)} \Pi_{\mathbf{V}} \mathcal{E}_{\mathbf{V}^\perp}(\tau) d\tau \right\|_2 \leq \sup_{\tau \in [0, t]} \|e^{\Pi_{\mathbf{V}} \mathbf{A}(t-\tau)}\|_2 \int_0^t \|\mathcal{E}_{\mathbf{V}^\perp}(\tau)\|_2 d\tau$$

**Energy Estimate:**

$$\|\mathcal{E}_{\mathbf{V}}(t)\|_2 \leq C_1 \|\mathcal{E}_{\mathbf{V}}(0)\|_2 + C_2 \int_0^t \|\mathcal{E}_{\mathbf{V}^\perp}(\tau)\|_2 d\tau,$$

where:

- $C_1 = \|e^{\Pi_{\mathbf{V}} \mathbf{A}t}\|_2$
- $C_2 = \sup_{\tau \in [0, t]} \|e^{\Pi_{\mathbf{V}} \mathbf{A}(t-\tau)}\|_2$

$C_1$  and  $C_2$  can be bounded by a function of  $t$  and the eigenvalues of  $\Pi_{\mathbf{V}} \mathbf{A}$ .

## Collocated MOR

# From PDE to ODE: Spatial Semidiscretization

**PDE:**

$$\frac{\partial w}{\partial t} + \mathcal{L}(w; \mu) = f(x, t; \mu), \quad w(x, 0; \mu) = w_0(x; \mu),$$

where:

- $w(x, t; \mu)$ : Solution in space  $x \in \Omega$  and time  $t \geq 0$  for parameter  $\mu \in \mathcal{P}$ .
- $\mathcal{L}(w; \mu)$ : Spatial differential operator.
- $f(x, t; \mu)$ : Source term.

**Spatial Semidiscretization:** Let  $\Omega$  be a computational domain defined as:

$$\Omega = \bigcup_{\Omega_i \in \mathcal{S}} \Omega_i, \quad \text{with } \mathcal{S} = \{T^S\},$$

where  $\{T^S\}$  represents a set of non-intersecting polytopes such that:

$$\Omega_i \cap \Omega_j = \emptyset \quad \text{for } i \neq j, \quad \text{and} \quad \bigcup_{\Omega_i \in T} \bar{\Omega}_i = \bar{\Omega}.$$

# Quadrature rule and Weighted Scalar Product

## Definition:

- Let the space discretization of two scalar functions  $u$  and  $v$  defined over  $\Omega$  be  $\mathbf{u}, \mathbf{v} \in \mathbb{R}^N$ :

$$\mathbf{u} = [u_1, u_2, \dots, u_N]^T, \quad \mathbf{v} = [v_1, v_2, \dots, v_N]^T.$$

## Weighted Discrete Scalar Product:

$$\int_{\Omega} u v dx \approx (\mathbf{u}, \mathbf{v})_D = \mathbf{u}^T \mathbf{D} \mathbf{v} = \sum_{i=1}^N \zeta_i u_i v_i,$$

where:

- $\mathbf{D} \in \mathbb{R}^{N \times N}$  is a diagonal weight matrix :

$$\mathbf{D} = \text{diag}(\zeta_1(\Omega_1), \zeta_2(\Omega_2), \dots, \zeta_N(\Omega_N)), \quad \zeta_i > 0.$$

- $\zeta_i$ : weights related to quadrature rules and the grid.

# Projection onto Reduced Space by Hyper-Reduction

**Key Idea:** Instead of  $\mathbf{q}(t)$ , the coefficients of the basis, we will use a subset of the components of  $\mathbf{w}(t)$ ,  $w_{j \in J}(t)$  with  $J \subset \{1, \dots, N\}$ , as the independent variables of the problem.

**Offline stage:** the HDM generates a database of  $N_{\text{snap}}$  snapshots  $\mathbf{w}(t_i)$  defined on the grid cells  $\Omega_1, \Omega_2, \dots, \Omega_N$ . As before they are collected in a matrix  $\mathbf{S} \in \mathbb{R}^{N \times N_{\text{snap}}}$ .

**POD Problem Reformulation:**

$$\min_{\mathbf{V} \in \mathbb{R}^{N \times k}} \|\mathbf{S} - \mathbf{V}\mathbf{V}^T \mathbf{D} \mathbf{S}\|_{F_D}^2 \quad \text{subject to} \quad \mathbf{V}^T \mathbf{D} \mathbf{V} = \mathbf{I}$$

**Def: Weighted Frobenius Norm:**

$$\|\mathbf{A}\|_{F_D} := \text{tr}(\mathbf{A}^T \mathbf{D} \mathbf{A})$$

# Solution and Projection Representation

Let:  $\tilde{\mathbf{S}} = \mathbf{D}^{-1/2}\mathbf{S}$ , then the previous POD problem becomes again:

$$\min_{\tilde{\mathbf{V}} \in \mathbb{R}^{N \times k}} \|\tilde{\mathbf{S}} - \tilde{\mathbf{V}}\tilde{\mathbf{V}}^T \tilde{\mathbf{S}}\|_F^2 \quad \text{subject to} \quad \tilde{\mathbf{V}}^T \tilde{\mathbf{V}} = \mathbf{I}$$

where  $\tilde{\mathbf{V}} = \mathbf{D}^{-1/2}\mathbf{V}$ .

Therefore:

$$\tilde{\mathbf{S}} = \tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{Z}}^T = [\tilde{\mathbf{U}}_r \quad \tilde{\mathbf{U}}_{N-r}] \begin{bmatrix} \tilde{\Sigma}_r & 0 \\ 0 & 0 \end{bmatrix} \tilde{\mathbf{Z}}^T$$

and hence:

$$\Phi = \mathbf{D}^{-1/2}\tilde{\mathbf{U}}_r.$$

The POD modes are recovered by a standard SVD and  $(\Phi, \Phi)_D = \mathbf{I}$ .

- As before, we take a subset of  $k$  columns of  $\Phi$  to define  $\mathbf{V} \in \mathbb{R}^{N \times k}$ , according to an energy criterion. We denote by  $\phi_{ij}$  the element of row  $i$  and column  $j$  of  $\mathbf{V}$ .

# Hyper-reduction

- As before, we take a subset of  $k$  columns of  $\Phi$  to define  $\mathbf{V} \in \mathbb{R}^{N \times k}$ , according to an energy criterion. We denote by  $\phi_{ij}$  the element of row  $i$  and column  $j$  of  $\mathbf{V}$ .

- Consider now

$$\mathbf{q}(t) = \{q_i(t)\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D = \left\{ \sum_{j=1}^N \zeta_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}$$

# Hyper-reduction

- As before, we take a subset of  $k$  columns of  $\Phi$  to define  $\mathbf{V} \in \mathbb{R}^{N \times k}$ , according to an energy criterion. We denote by  $\phi_{ij}$  the element of row  $i$  and column  $j$  of  $\mathbf{V}$ .
- Consider now
$$\mathbf{q}(t) = \{q_i(t)\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D = \left\{ \sum_{j=1}^N \zeta_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}$$
- We can approximate the discrete scalar product above by a quadrature:
$$\mathbf{q}(t) = \{q_i(t)\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D \approx \left\{ \sum_{j \in J} \tilde{\zeta}_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k} \quad \text{where } J \subset \{1, \dots, N\}.$$

# Hyper-reduction

- As before, we take a subset of  $k$  columns of  $\Phi$  to define  $\mathbf{V} \in \mathbb{R}^{N \times k}$ , according to an energy criterion. We denote by  $\phi_{ij}$  the element of row  $i$  and column  $j$  of  $\mathbf{V}$ .
- Consider now
$$\mathbf{q}(t) = \{q_i(t)\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D = \left\{ \sum_{j=1}^N \zeta_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}$$
- We can approximate the discrete scalar product above by a quadrature:
$$\mathbf{q}(t) = \{q_i(t)\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D \approx \left\{ \sum_{j \in J} \tilde{\zeta}_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k} \quad \text{where } J \subset \{1, \dots, N\}.$$
- The empirical quadrature weights  $\tilde{\zeta}_j$  and points are determined to best approximate the original projections.

# Hyper-reduction

- As before, we take a subset of  $k$  columns of  $\Phi$  to define  $\mathbf{V} \in \mathbb{R}^{N \times k}$ , according to an energy criterion. We denote by  $\phi_{ij}$  the element of row  $i$  and column  $j$  of  $\mathbf{V}$ .
- Consider now
$$\mathbf{q}(t) = \{q_i(t)\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D = \left\{ \sum_{j=1}^N \zeta_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}$$
- We can approximate the discrete scalar product above by a quadrature:
$$\mathbf{q}(t) = \{q_i(t)\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D \approx \left\{ \sum_{j \in J} \tilde{\zeta}_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k} \quad \text{where } J \subset \{1, \dots, N\}.$$
- The empirical quadrature weights  $\tilde{\zeta}_j$  and points are determined to best approximate the original projections.
- To do that, define:

$$G_i = \begin{bmatrix} w_1(t^1) \phi_{i1} & \cdots & w_N(t^1) \phi_{iN} \\ \vdots & \ddots & \vdots \\ w_1(t^{N_{\text{snap}}}) \phi_{i1} & \cdots & w_N(t^{N_{\text{snap}}}) \phi_{iN} \end{bmatrix} \in \mathbb{R}^{N_{\text{snap}} \times N}$$

# Non-Negative Empirical Weights and Sparsity

- Define also:

$$\mathbf{d}_i = \begin{bmatrix} \sum_{j=1}^N \zeta_j w_j(t^1) \phi_{ij} \\ \vdots \\ \sum_{j=1}^N \zeta_j w_j(t^{N_{\text{snap}}}) \phi_{ij} \end{bmatrix} \in \mathbb{R}^{N_{\text{snap}}}$$

# Non-Negative Empirical Weights and Sparsity

- Define also:

$$\mathbf{d}_i = \begin{bmatrix} \sum_{j=1}^N \zeta_j w_j(t^1) \phi_{ij} \\ \vdots \\ \sum_{j=1}^N \zeta_j w_j(t^{N_{\text{snap}}}) \phi_{ij} \end{bmatrix} \in \mathbb{R}^{N_{\text{snap}}}$$

- The objective is to find non-negative empirical weights  $\tilde{\zeta} = \{\tilde{\zeta}_j\}_{j \in J}$  and a set  $J \subset \{1, \dots, N\}$  to promote sparsity, accurately approximating the projections.

# Non-Negative Empirical Weights and Sparsity

- Define also:

$$\mathbf{d}_i = \begin{bmatrix} \sum_{j=1}^N \zeta_j w_j(t^1) \phi_{ij} \\ \vdots \\ \sum_{j=1}^N \zeta_j w_j(t^{N_{\text{snap}}}) \phi_{ij} \end{bmatrix} \in \mathbb{R}^{N_{\text{snap}}}$$

- The objective is to find non-negative empirical weights  $\tilde{\zeta} = \{\tilde{\zeta}_j\}_{j \in J}$  and a set  $J \subset \{1, \dots, N\}$  to promote sparsity, accurately approximating the projections.
- This problem can be formulated as:

$$\tilde{\zeta} = \arg \min_{\zeta \in \Lambda} \|\zeta\|_0,$$

$$\Lambda = \{\zeta \in \mathbb{R}_+^N : \|\mathbf{G}\zeta - \mathbf{d}\|_2 \leq \epsilon \|\mathbf{d}\|_2\}$$

# Non-Negative Empirical Weights and Sparsity

- Define also:

$$\mathbf{d}_i = \begin{bmatrix} \sum_{j=1}^N \zeta_j w_j(t^1) \phi_{ij} \\ \vdots \\ \sum_{j=1}^N \zeta_j w_j(t^{N_{\text{snap}}}) \phi_{ij} \end{bmatrix} \in \mathbb{R}^{N_{\text{snap}}}$$

- The objective is to find non-negative empirical weights  $\tilde{\zeta} = \{\tilde{\zeta}_j\}_{j \in J}$  and a set  $J \subset \{1, \dots, N\}$  to promote sparsity, accurately approximating the projections.
- This problem can be formulated as:

$$\tilde{\zeta} = \arg \min_{\zeta \in \Lambda} \|\zeta\|_0,$$

$$\Lambda = \{\zeta \in \mathbb{R}_+^N : \|\mathbf{G}\zeta - \mathbf{d}\|_2 \leq \epsilon \|\mathbf{d}\|_2\}$$

- $\mathbf{G} \in \mathbb{R}^{kN_{\text{snap}} \times N}$  and  $\mathbf{d} \in \mathbb{R}^{kN_{\text{snap}}}$  are the concatenation of  $\mathbf{G}_i$  and  $\mathbf{d}_i$  for each one of the  $k$ -dimensional POD basis.

- The problem  $\tilde{\zeta} = \arg \min_{\zeta \in \Lambda} \|\zeta\|_0$ , is NP-hard.

- The problem  $\tilde{\zeta} = \arg \min_{\zeta \in \Lambda} \|\zeta\|_0$ , is NP-hard.
- It is approximated by solving a the Non-Negative Least Squares problem (C. Farhat):

$$\tilde{\zeta} = \arg \min_{\zeta \in \mathbb{R}_+^N} \|G\zeta - \mathbf{d}\|_2^2,$$

using the Lawson-Hanson algorithm.

- The problem  $\tilde{\zeta} = \arg \min_{\zeta \in \Lambda} \|\zeta\|_0$ , is NP-hard.
- It is approximated by solving a the Non-Negative Least Squares problem (C. Farhat):

$$\tilde{\zeta} = \arg \min_{\zeta \in \mathbb{R}_+^N} \|G\zeta - \mathbf{d}\|_2^2,$$

using the Lawson-Hanson algorithm.

- This algorithm encourages sparsity in the solution and ensures that the solution meets the error tolerance  $\epsilon$ , i.e.,  $\|G\tilde{\zeta} - \mathbf{d}\|_2 \leq \epsilon \|\mathbf{d}\|_2$ .

# Non-Negative Empirical Weights and Sparsity

- The problem  $\tilde{\zeta} = \arg \min_{\zeta \in \Lambda} \|\zeta\|_0$ , is NP-hard.
- It is approximated by solving a the Non-Negative Least Squares problem (C. Farhat):

$$\tilde{\zeta} = \arg \min_{\zeta \in \mathbb{R}_+^N} \|G\zeta - \mathbf{d}\|_2^2,$$

using the Lawson-Hanson algorithm.

- This algorithm encourages sparsity in the solution and ensures that the solution meets the error tolerance  $\epsilon$ , i.e.,  $\|G\tilde{\zeta} - \mathbf{d}\|_2 \leq \epsilon \|\mathbf{d}\|_2$ .
- The resulting empirical weights  $\tilde{\zeta}$  are then used to select a reduced set of quadrature points, corresponding to the non-zero weights.

# Collocation and Hyper-Reduced Prolongation Operator

- We have now the approximation:

$$\mathbf{q}(t) = \{q_i\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D \approx \left\{ \sum_{j \in J} \tilde{\zeta}_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}, \text{ where } J \subset \{1, \dots, N\} \text{ is a small set of quadrature points.}$$

# Collocation and Hyper-Reduced Prolongation Operator

- We have now the approximation:

$$\mathbf{q}(t) = \{q_i\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D \approx \left\{ \sum_{j \in J} \tilde{\zeta}_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}, \text{ where } J \subset \{1, \dots, N\} \text{ is a small set of quadrature points.}$$

- Hence:

$$\frac{d\mathbf{q}(t)}{dt} = \left\{ \frac{dq_i}{dt} \right\}_{1 \leq i \leq k} = \left\{ \sum_{j \in J} \tilde{\zeta}_j \phi_{ij} \frac{dw_j(t)}{dt} \right\}_{1 \leq i \leq k}$$

# Collocation and Hyper-Reduced Prolongation Operator

- We have now the approximation:

$$\mathbf{q}(t) = \{q_i\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D \approx \left\{ \sum_{j \in J} \tilde{\zeta}_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}, \text{ where } J \subset \{1, \dots, N\} \text{ is a small set of quadrature points.}$$

- Hence:

$$\frac{d\mathbf{q}(t)}{dt} = \left\{ \frac{dq_i}{dt} \right\}_{1 \leq i \leq k} = \left\{ \sum_{j \in J} \tilde{\zeta}_j \phi_{ij} \frac{dw_j(t)}{dt} \right\}_{1 \leq i \leq k}$$

- For  $i \in J$  we obtain

$$\frac{dw_i(t)}{dt} = \sum_{m=1}^N A_{im} w_m(t) = \sum_{m=1}^N A_{im} \sum_{s=1}^k \phi_{sm} \sum_{j \in J} \tilde{\zeta}_j \phi_{sj} w_j(t).$$

# Collocation and Hyper-Reduced Prolongation Operator

- We have now the approximation:

$$\mathbf{q}(t) = \{q_i\}_{1 \leq i \leq k} = (\mathbf{V}, \mathbf{w}(t))_D \approx \left\{ \sum_{j \in J} \tilde{\zeta}_j w_j(t) \phi_{ij} \right\}_{1 \leq i \leq k}, \text{ where } J \subset \{1, \dots, N\} \text{ is a small set of quadrature points.}$$

- Hence:

$$\frac{d\mathbf{q}(t)}{dt} = \left\{ \frac{dq_i}{dt} \right\}_{1 \leq i \leq k} = \left\{ \sum_{j \in J} \tilde{\zeta}_j \phi_{ij} \frac{dw_j(t)}{dt} \right\}_{1 \leq i \leq k}$$

- For  $i \in J$  we obtain

$$\frac{dw_i(t)}{dt} = \sum_{m=1}^N A_{im} w_m(t) = \sum_{m=1}^N A_{im} \sum_{s=1}^k \phi_{sm} \sum_{j \in J} \tilde{\zeta}_j \phi_{sj} w_j(t).$$

- We conclude that to advance  $\mathbf{w}(t)$  in time, one only needs to advance the solution at the quadrature points using the same spatial discretization of the HDM, i.e., the same scheme.

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.
- The present approach hyper-reduces the discrete solution projection operator itself.

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.
- The present approach hyper-reduces the discrete solution projection operator itself.
- Objective: Identify a subset of cells in the computational domain for local solution of the HDM.

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.
- The present approach hyper-reduces the discrete solution projection operator itself.
- Objective: Identify a subset of cells in the computational domain for local solution of the HDM.
- Steps:

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.
- The present approach hyper-reduces the discrete solution projection operator itself.
- Objective: Identify a subset of cells in the computational domain for local solution of the HDM.
- Steps:
  - 1 Solve the HDM locally in selected cells.

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.
- The present approach hyper-reduces the discrete solution projection operator itself.
- Objective: Identify a subset of cells in the computational domain for local solution of the HDM.
- Steps:
  - 1 Solve the HDM locally in selected cells.
  - 2 Extend the local solution to local stencils.

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.
- The present approach hyper-reduces the discrete solution projection operator itself.
- Objective: Identify a subset of cells in the computational domain for local solution of the HDM.
- Steps:
  - 1 Solve the HDM locally in selected cells.
  - 2 Extend the local solution to local stencils.
  - 3 Propagate the solution to the rest of the domain.

- Traditional hyper-reduction methods focus on reducing the computational cost of assembling matrices for the discrete projection of non-linear terms.
- The present approach hyper-reduces the discrete solution projection operator itself.
- Objective: Identify a subset of cells in the computational domain for local solution of the HDM.
- Steps:
  - 1 Solve the HDM locally in selected cells.
  - 2 Extend the local solution to local stencils.
  - 3 Propagate the solution to the rest of the domain.
- Error analysis follows the same path as that shown before, with an additional error introduced by the approximate projection.

## Closure Models

# Neural Correction for Residual Dynamics (L. Mathelin)

- Recall that the residual  $\mathbf{r}(t) \in \mathbb{R}^N$  is introduced by approximating  $\mathbf{w}(t)$  as  $\mathbf{V}\mathbf{q}(t)$ :

$$\mathbf{V} \frac{d\mathbf{q}}{dt}(t) = \mathbf{f}(\mathbf{V}\mathbf{q}(t), t) + \mathbf{r}(t) \iff \mathbf{r}(t) = \mathbf{V} \frac{d\mathbf{q}}{dt}(t) - \mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$$

**Objective:** Address the impact of unresolved dynamics in ROM using a neural correction term.

**Corrected Dynamics:**

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

where:

- $\mathbf{f}(\mathbf{V}\mathbf{q}(t), t)$ : Resolved dynamics.
- $\mathbf{R}(\mathbf{y})$ : Neural correction term, learned from data.

## Memory Evolution:

$$\frac{d\mathbf{y}}{dt} = \mathbf{E}(\mathbf{y}) - \Lambda\mathbf{y},$$

where:

- $\mathbf{y}$ : Memory variable.  $\mathbf{E}(\mathbf{y})$ : Coupling term learned from data.
- $\Lambda$ : Dissipative operator for stability.

## Key Features:

- Neural networks approximate  $\mathbf{R}(\mathbf{y})$  and  $\mathbf{E}(\mathbf{y})$ .
- Applicable to systems with scarce and noisy data.

## Outcome:

- Significantly improved predictions compared to classical ROM.
- Efficient handling of long time horizons and unresolved dynamics.