Model Order Reduction for Multi-scale, Multi-physics Problems

Background

Karthik Duraisamy

Computational AeroSciences Laboratory

caslab.engin.umich.edu & afcoe.engin.umich.edu

Outline

- Introduction (today)
- Theory (today)
- Practice (tomorrow)
- The leading edge (tomorrow)

Theory Outline

SVD, QR decompositions

Least squares regression

Sampling & Reconstruction

Sensing

Resources

https://caslab.engin.umich.edu/teaching

- Isaac Newton Institute tutorial on Model Order reduction for complex systems (Jan 2023)
  1. Model Order Reduction theory manual
     http://websites.umich.edu/~caslab/docs/Newton/MOR_Theory.pdf
  2. PERFORM (Prototyping environment for reacting flow order reduction methods : code)
  3. PERFORM (Prototyping environment for reacting flow order reduction methods : doc)
  4. Slides (coming soon)

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

- $\boldsymbol{\Sigma}$ is a diagonal matrix.

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

- $\boldsymbol{\Sigma}$ is a diagonal matrix.
- Diagonal entries of $\boldsymbol{\Sigma}$ are the singular values of $\boldsymbol{A}$.

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

- $\boldsymbol{\Sigma}$ is a diagonal matrix.
- Diagonal entries of $\boldsymbol{\Sigma}$ are the singular values of $\boldsymbol{A}$.
- Singular values are real and non-negative.

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

- $\boldsymbol{\Sigma}$ is a diagonal matrix.
- Diagonal entries of $\boldsymbol{\Sigma}$ are the singular values of $\boldsymbol{A}$.
- Singular values are real and non-negative.
- Singular values are typically arranged in descending order.

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

- Columns of $\boldsymbol{U}$ are the left singular vectors of $\boldsymbol{A}$.

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

- Columns of $\boldsymbol{U}$ are the left singular vectors of $\boldsymbol{A}$.
- Columns of $\boldsymbol{V}$ are the right singular vectors of $\boldsymbol{A}$.

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

- Columns of $\boldsymbol{U}$ are the left singular vectors of $\boldsymbol{A}$.
- Columns of $\boldsymbol{V}$ are the right singular vectors of $\boldsymbol{A}$.
- The left and right singular vectors are orthonormal.

Singular Value Decomposition

Any matrix has a singular value decomposition.

SVD of a general matrix $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is given by

$$\underbrace{\mathbf{A}}_{m \times n} = \underbrace{\mathbf{U}}_{m \times m} \underbrace{\boldsymbol{\Sigma}}_{m \times n} \underbrace{\mathbf{V}^*}_{n \times n}$$

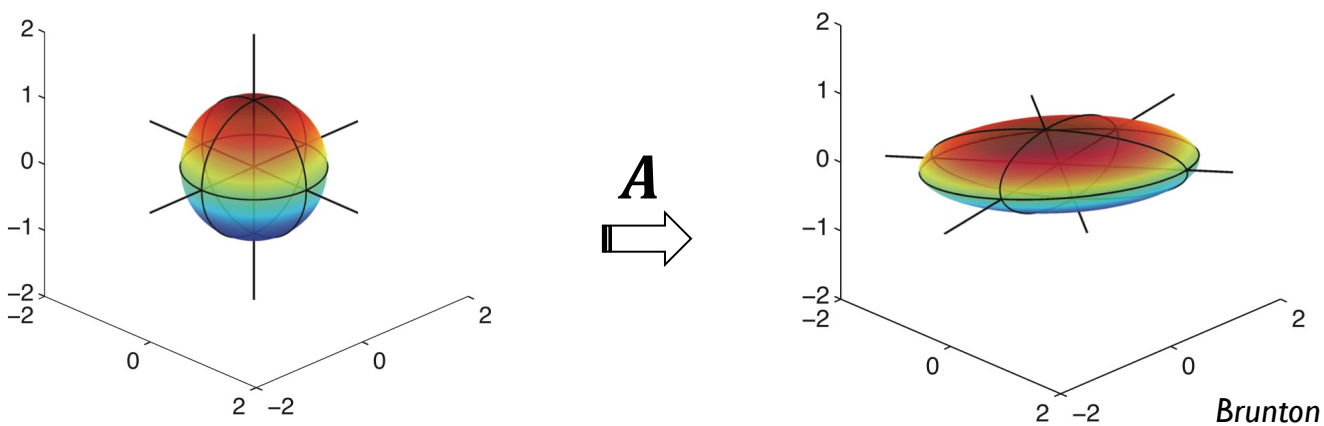
- Columns of $\boldsymbol{U}$ are the left singular vectors of $\boldsymbol{A}$.
- Columns of $\boldsymbol{V}$ are the right singular vectors of $\boldsymbol{A}$.
- The left and right singular vectors are orthonormal.
- $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary matrices.

Geometric Interpretation of SVD

Matrix multiplication introduces a rotation and a stretching action.

Singular values are the lengths of semi-axes of the hyper-ellipsoid obtained as a result of operation of matrix $\boldsymbol{A}$ on the unit hypersphere.

In 3-D:



Brunton and Kutz 2019

Singular Value Decomposition

Columns of $\boldsymbol{A}$ (snapshots matrix) can be

- Measurements from experiments
- Image pixels
- State of a physical system (velocity, pressure, etc.) at discrete points
- …

Rank of $\boldsymbol{A}$ is equal to the number of non-zero singular values.

$$\underbrace{\mathbf{A}}_{m\times n} = \underbrace{\mathbf{U}}_{m\times m}\underbrace{\mathbf{\Sigma}}_{m\times n}\underbrace{\mathbf{V}^*}_{n\times n} \quad \Rightarrow \quad \underbrace{\mathbf{A}}_{m\times n} = \underbrace{\hat{\mathbf{U}}}_{m\times r}\underbrace{\hat{\mathbf{\Sigma}}}_{r\times r}\underbrace{\hat{\mathbf{V}}^*}_{r\times n}$$

SVD Facts

$\boldsymbol{A} = \boldsymbol{U\Sigma V}^*$

1. Every matrix has a SVD and singular values are uniquely determined, but they are not necessarily distinct.

SVD Facts

$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*$

1. Every matrix has a SVD and singular values are uniquely determined, but they are not necessarily distinct.
2. If $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ is rank deficient (of rank $r < n$), then $\boldsymbol{\Sigma}$ will have r positive diagonal entries.

SVD Facts

$\boldsymbol{A} = \boldsymbol{U\Sigma V}^*$

1. Every matrix has a SVD and singular values are uniquely determined, but they are not necessarily distinct.
2. If $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ is rank deficient (of rank $r < n$), then $\boldsymbol{\Sigma}$ will have r positive diagonal entries.
3. Singular values of $\boldsymbol{A}$ are the square roots of the eigenvalues of $\boldsymbol{A}^*\boldsymbol{A}$

$$\begin{aligned} \mathbf{A}^*\mathbf{A} &= [\mathbf{U\Sigma V}^*]^* [\mathbf{U\Sigma V}^*] \\ &= \mathbf{V\Sigma}^2\mathbf{V}^*, \end{aligned}$$

SVD Facts

$\boldsymbol{A} = \boldsymbol{U\Sigma V}^*$

1. Every matrix has a SVD and singular values are uniquely determined, but they are not necessarily distinct.
2. If $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ is rank deficient (of rank $r < n$), then $\boldsymbol{\Sigma}$ will have r positive diagonal entries.
3. Singular values of $\boldsymbol{A}$ are the square roots of the eigenvalues of $\boldsymbol{A}^*\boldsymbol{A}$

$$\begin{aligned} \mathbf{A}^*\mathbf{A} &= [\mathbf{U\Sigma V}^*]^* [\mathbf{U\Sigma V}^*] \\ &= \mathbf{V\Sigma}^2\mathbf{V}^*, \end{aligned}$$

4. Same can be proved for the eigenvalues of $\boldsymbol{AA}^*$.

SVD Facts

$\boldsymbol{A} = \boldsymbol{U\Sigma V}^*$

1. Every matrix has a SVD and singular values are uniquely determined, but they are not necessarily distinct.
2. If $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ is rank deficient (of rank $r < n$), then $\boldsymbol{\Sigma}$ will have r positive diagonal entries.
3. Singular values of $\boldsymbol{A}$ are the square roots of the eigenvalues of $\boldsymbol{A}^*\boldsymbol{A}$

$$\begin{aligned}\mathbf{A}^*\mathbf{A} &= [\mathbf{U\Sigma V}^*]^*\,[\mathbf{U\Sigma V}^*] \\ &= \mathbf{V\Sigma}^2\mathbf{V}^*,\end{aligned}$$

4. Same can be proved for the eigenvalues of $\boldsymbol{AA}^*$.
5. The rank of $\boldsymbol{A}$ is equal to the number of its non-zero singular values.
6. $$\|\mathbf{A}\|_2 = \sqrt{\lambda_{\max}(\mathbf{A}^*\mathbf{A})} = \sigma_{\max}(\mathbf{A})$$

SVD Facts

7. Given $\mathbf{A} = \Sigma_{j=1}^{r} \sigma_j \mathbf{u}_j \mathbf{v}_j^*$ for any $0 < q < r$ the matrix $\mathbf{A}_q = \Sigma_{j=1}^{q} \sigma_j \mathbf{u}_j \mathbf{v}_j^*$ satisfies the following properties:

$$\|\mathbf{A} - \mathbf{A}_q\|_2 = \sigma_{q+1}$$

$$\|\mathbf{A} - \mathbf{A}_q\|_F = \sqrt{\sum_{i=q+1}^{r} \sigma_i^2}$$

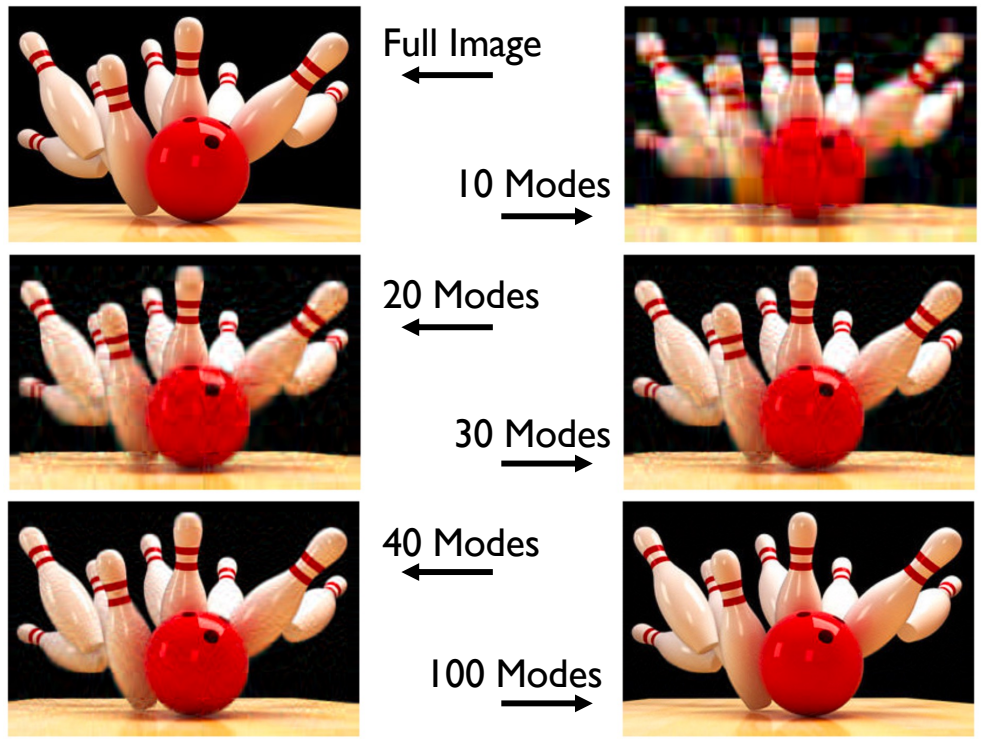
$$\|\mathbf{A}^+ - \mathbf{A}_q^+\|_F = \sqrt{\sum_{i=q+1}^{r} \frac{1}{\sigma_i^2}}$$

Image Compression with SVD

$$\underbrace{\mathbf{A}}_{m\times n} = \underbrace{\mathbf{U}}_{m\times m}\underbrace{\mathbf{\Sigma}}_{m\times n}\underbrace{\mathbf{V}^*}_{n\times n}$$

$$\Downarrow$$

$$\underbrace{\mathbf{A}}_{m\times n} = \underbrace{\hat{\mathbf{U}}}_{m\times r}\underbrace{\hat{\mathbf{\Sigma}}}_{r\times r}\underbrace{\hat{\mathbf{V}}^*}_{r\times n}$$



Outline

SVD, QR decompositions

Least squares regression

Sampling & Reconstruction

Sensing

QR Factorization

If $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ and $\boldsymbol{A}$ has full column rank, then the QR decomposition is given by

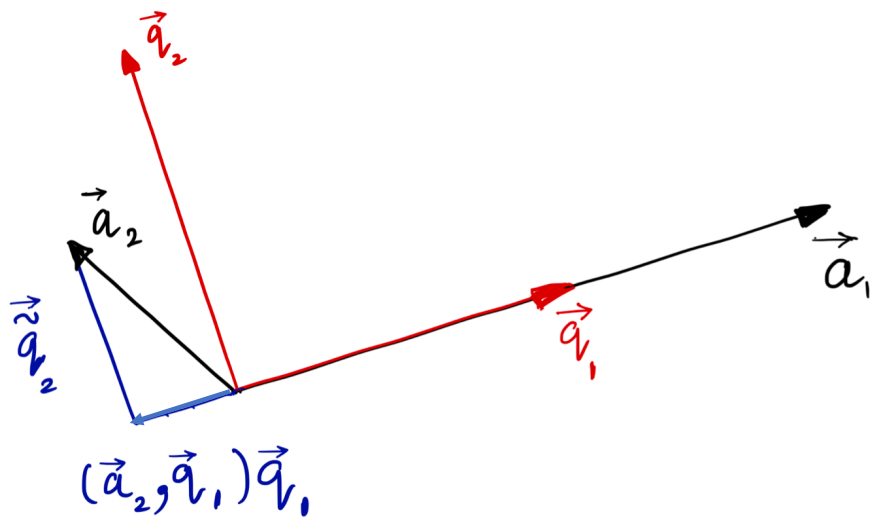
$$\mathbf{A} = \mathbf{QR}$$

- $\mathbf{Q} \in \mathbb{C}^{m \times n} = \{\mathbf{q}_1, \mathbf{q}_2, \ldots \mathbf{q}_n\}$ are orthonormal vectors.
- $R \in \mathbb{C}^{n \times n}$ is an upper-triangular matrix with non-zero diagonal elements.
- QR factorization can be computed by the Gram-Schmidt procedure.

Gram-Schmidt Procedure

Given a linearly independent set of vectors $S \equiv \{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, ...\mathbf{a}_n\}$, where $\mathbf{a}_i \in \mathbb{R}^m$, the Gram-Schmidt procedure finds an orthonormal set of vectors $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, ...\mathbf{q}_n\}$ that spans the same subspace as S. The procedure is as follows:

1. $\tilde{\mathbf{q}}_1 = \mathbf{a}_1$
2. $\mathbf{q}_1 = \frac{\tilde{\mathbf{q}}_1}{\|\tilde{\mathbf{q}}_1\|}$
3. $\tilde{\mathbf{q}}_2 = \mathbf{a}_2 - (\mathbf{q}_1^T \mathbf{a}_2)\mathbf{q}_1$
4. $\mathbf{q}_2 = \frac{\tilde{\mathbf{q}}_2}{\|\tilde{\mathbf{q}}_2\|}$
5. $\tilde{\mathbf{q}}_3 = \mathbf{a}_3 - (\mathbf{q}_1^T \mathbf{a}_3)\mathbf{q}_1 - (\mathbf{q}_2^T \mathbf{a}_3)\mathbf{q}_2$
6. $\mathbf{q}_3 = \frac{\tilde{\mathbf{q}}_3}{\|\tilde{\mathbf{q}}_3\|}$
7. Repeat recursively



Gram-Schmidt Procedure

Given a linearly independent set of vectors $S \equiv \{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, ...\mathbf{a}_n\}$, where $\mathbf{a}_i \in \mathbb{R}^m$, the Gram-Schmidt procedure finds an orthonormal set of vectors $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, ...\mathbf{q}_n\}$ that spans the same subspace as S. The procedure is as follows:

1. $\tilde{\mathbf{q}}_1 = \mathbf{a}_1$
2. $\mathbf{q}_1 = \frac{\tilde{\mathbf{q}}_1}{\|\tilde{\mathbf{q}}_1\|}$
3. $\tilde{\mathbf{q}}_2 = \mathbf{a}_2 - (\mathbf{q}_1^T \mathbf{a}_2)\mathbf{q}_1$
4. $\mathbf{q}_2 = \frac{\tilde{\mathbf{q}}_2}{\|\tilde{\mathbf{q}}_2\|}$
5. $\tilde{\mathbf{q}}_3 = \mathbf{a}_3 - (\mathbf{q}_1^T \mathbf{a}_3)\mathbf{q}_1 - (\mathbf{q}_2^T \mathbf{a}_3)\mathbf{q}_2$
6. $\mathbf{q}_3 = \frac{\tilde{\mathbf{q}}_3}{\|\tilde{\mathbf{q}}_3\|}$
7. Repeat recursively

To compute QR factorization of $\boldsymbol{A}$ using the Gram-Schmidt procedure:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \end{bmatrix}$$

Gram-Schmidt Procedure

Given a linearly independent set of vectors $S \equiv \{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, ...\mathbf{a}_n\}$, where $\mathbf{a}_i \in \mathbb{R}^m$, the Gram-Schmidt procedure finds an orthonormal set of vectors $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, ...\mathbf{q}_n\}$ that spans the same subspace as S. The procedure is as follows:

1. $\tilde{\mathbf{q}}_1 = \mathbf{a}_1$
2. $\mathbf{q}_1 = \frac{\tilde{\mathbf{q}}_1}{\|\tilde{\mathbf{q}}_1\|}$
3. $\tilde{\mathbf{q}}_2 = \mathbf{a}_2 - (\mathbf{q}_1^T \mathbf{a}_2)\mathbf{q}_1$
4. $\mathbf{q}_2 = \frac{\tilde{\mathbf{q}}_2}{\|\tilde{\mathbf{q}}_2\|}$
5. $\tilde{\mathbf{q}}_3 = \mathbf{a}_3 - (\mathbf{q}_1^T \mathbf{a}_3)\mathbf{q}_1 - (\mathbf{q}_2^T \mathbf{a}_3)\mathbf{q}_2$
6. $\mathbf{q}_3 = \frac{\tilde{\mathbf{q}}_3}{\|\tilde{\mathbf{q}}_3\|}$
7. Repeat recursively

To compute QR factorization of $\boldsymbol{A}$ using the Gram-Schmidt procedure:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \dots & \mathbf{q}_n \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} \|\tilde{\mathbf{q}}_1\| & \mathbf{q}_1^T \mathbf{a}_2 & \dots & \mathbf{q}_1^T \mathbf{a}_n \\ 0 & \|\tilde{\mathbf{q}}_2\| & \dots & \mathbf{q}_2^T \mathbf{a}_n \\ 0 & 0 & \dots & \mathbf{q}_i^T \mathbf{a}_n \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & \|\tilde{\mathbf{q}}_n\| \end{bmatrix}$$

Outline

SVD, QR decompositions

Least squares regression

Sampling & Reconstruction

Sensing

Least-squares Regression

Consider an over-determined system of equations

$$\mathbf{A}\mathbf{x} = \mathbf{y}$$

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Least-squares Regression

Consider an overdetermined system of equations

$$\mathbf{A}\mathbf{x} = \mathbf{y}$$

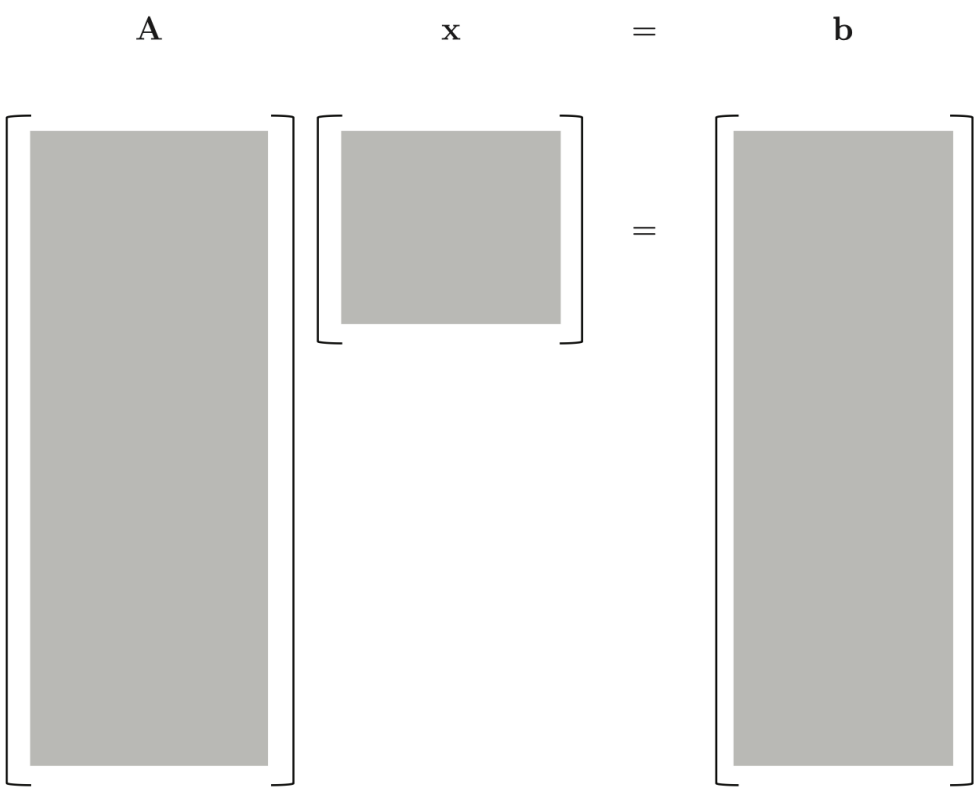
$\boxed{m > n}$

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

y: Observables (data, snapshots of quantities of interest)

x: Model parameters (unknown)

Least-squares Regression



Brunton and Kutz, 2019

Least-squares Regression

$\mathbf{Ax} = \mathbf{y}$

An over-determined system usually has no solution.

💡 But we can search for parameters that fit the equations best.

This can be done by solving the following optimization problem:

$$\hat{\mathbf{x}} = arg \min_{\mathbf{x}} S(\mathbf{x})$$

Objective function:

$$S(\mathbf{x}) = \left\|\mathbf{y} - \mathbf{Ax}\right\|_2^2$$

Least-squares Regression

$\mathbf{A}\mathbf{x} = \mathbf{y}$

The optimization problem has a unique solution if $\boldsymbol{A}$ is full column rank.

$$\hat{\mathbf{x}} = arg \min_{\mathbf{x}} S(\mathbf{x})$$

$$\boxed{S(\mathbf{x}) = \left\|\mathbf{y} - \mathbf{A}\mathbf{x}\right\|_2^2 = (\mathbf{y} - \mathbf{A}\mathbf{x})^{\mathrm{T}}(\mathbf{y} - \mathbf{A}\mathbf{x}) = \mathbf{y}^{\mathrm{T}}\mathbf{y} - \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{y} - \mathbf{y}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}}$$

Substitute $(\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{y})^{\mathrm{T}} = \mathbf{y}^{\mathrm{T}}\mathbf{A}\mathbf{x}$

$$\boxed{S(\mathbf{x}) = \mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}}$$

Least-squares Regression

$\mathbf{A}\mathbf{x} = \mathbf{y}$

$$\boxed{S(\mathbf{x}) = \mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}}$$

Differentiate with respect to x:

$$-\mathbf{A}^{\mathrm{T}}\mathbf{y} + (\mathbf{A}^{\mathrm{T}}\mathbf{A})\mathbf{x} = 0$$

First-order condition

$\boldsymbol{A^T A}$: The Gramian matrix of $\boldsymbol{A}$

$\boldsymbol{A^T y}$: The moment matrx

Least-squares Regression

$\mathbf{A}\mathbf{x} = \mathbf{y}$

$$\boxed{S(\mathbf{x}) = \mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}}$$

Differentiate with respect to x:

$$-\mathbf{A}^{\mathrm{T}}\mathbf{y} + (\mathbf{A}^{\mathrm{T}}\mathbf{A})\mathbf{x} = 0$$

First-order condition

The solution to the optimization problem:

$$\hat{\mathbf{x}} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{y} = \mathbf{A}^{+}\mathbf{y}$$

The second-order condition for the minimum:

$$\boldsymbol{A^T A} > \mathbf{0}$$

Pseudoinverse (from last lecture)

Any matrix has a pseudoinverse.

But $\boldsymbol{A}^+$ can only be computed explicitly under these conditions:

1. If $\boldsymbol{A}$ is full column rank, then $\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}$ is invertible and $\boxed{\boldsymbol{A}^+ = (\boldsymbol{A}^*\boldsymbol{A})^{-\mathbf{1}}\boldsymbol{A}^*}$.

   - This is called the left inverse as in this case $\boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{I}$.
   - In this case the pseudoinverse can also be computed by the QR decomposition:

$$\mathbf{A}^+ = \mathbf{R}^{-1}\mathbf{Q}^T$$

Least-squares Regression

$\mathbf{A}\mathbf{x} = \mathbf{y}$

$$\boxed{S(\mathbf{x}) = \mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}}$$

Differentiate with respect to x:

$$-\mathbf{A}^{\mathrm{T}}\mathbf{y} + (\mathbf{A}^{\mathrm{T}}\mathbf{A})\mathbf{x} = 0$$

First order condition

The solution to the optimization problem:

$$\hat{\mathbf{x}} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{y} = \mathbf{A}^{+}\mathbf{y}$$

$\boldsymbol{A}\hat{\boldsymbol{x}}$ is the orthogonal projection of $\boldsymbol{y}$ onto the $range(\boldsymbol{A})$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}\mathbf{y}$$

Least-squares Regression

$\mathbf{Ax} = \mathbf{y}$

These statements are equivalent:

- $A\hat{x}$ is the orthogonal projection of y onto the $range(A)$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}\mathbf{y}$$

- The residual $\mathbf{r} = \mathbf{A}\hat{\mathbf{x}} - \mathbf{y}$ is orthogonal to the range of A.

Least-squares Regression

$\mathbf{Ax} = \mathbf{y}$

These statements are equivalent:

- $A\hat{x}$ is the orthogonal projection of y onto the $range(A)$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}\mathbf{y}$$

- The residual $\mathbf{r} = \mathbf{A}\hat{\mathbf{x}} - \mathbf{y}$ is orthogonal to the range of A.
- r is in the null space of A^T

Least-squares Regression

$\mathbf{Ax} = \mathbf{y}$

These statements are equivalent:

- $A\hat{x}$ is the orthogonal projection of y onto the $range(A)$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}\mathbf{y}$$

- The residual $\mathbf{r} = \mathbf{A}\hat{\mathbf{x}} - \mathbf{y}$ is orthogonal to the range of $\boldsymbol{A}$.
- r is in the null space of $\boldsymbol{A}^{\boldsymbol{T}}$
- $\mathbf{A}^{T}(\mathbf{A}\hat{\mathbf{x}} - \mathbf{y}) = 0$

Least-squares Regression

$\mathbf{Ax} = \mathbf{y}$

These statements are equivalent:

- $A\hat{x}$ is the orthogonal projection of y onto the $range(A)$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}\mathbf{y}$$

- The residual $\mathbf{r} = \mathbf{A}\hat{\mathbf{x}} - \mathbf{y}$ is orthogonal to the range of $\boldsymbol{A}$.
- r is in the null space of $\boldsymbol{A}^{\boldsymbol{T}}$
- $\mathbf{A}^{T}(\mathbf{A}\hat{\mathbf{x}} - \mathbf{y}) = 0$
- $(\mathbf{r}, \mathbf{Az}) = 0 \quad \forall \mathbf{z} \in \mathbb{R}^{n}$

Least-squares Regression

$\mathbf{Ax} = \mathbf{y}$

These statements are equivalent:

- $A\hat{x}$ is the orthogonal projection of y onto the $range(A)$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \underbrace{\mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}}_{\text{Projection matrix}}\mathbf{y}$$

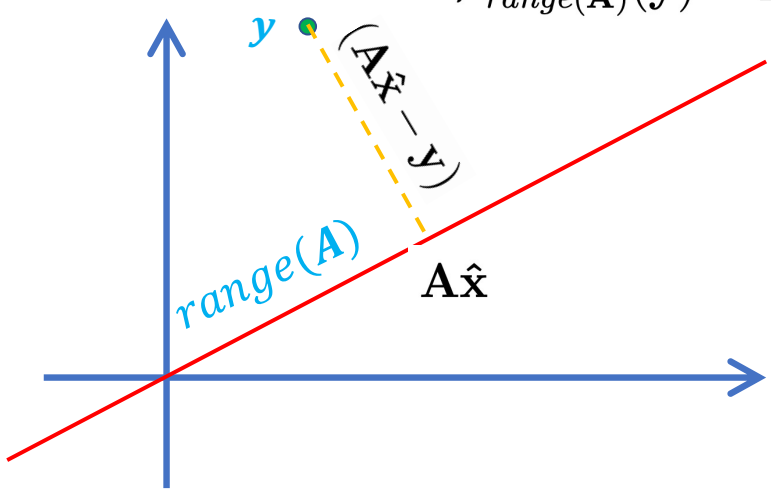
Least-squares Regression

$\mathbf{Ax} = \mathbf{y}$

These statements are equivalent:

- $A\hat{x}$ is the orthogonal projection of y onto the $range(A)$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \underbrace{\mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}}_{\text{Projection matrix}}\mathbf{y}$$



Least-squares Regression

$\mathbf{A}\mathbf{x} = \mathbf{y}$

These statements are equivalent:

- $A\hat{x}$ is the orthogonal projection of y onto the $range(A)$:

$$\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{T}\mathbf{y}$$

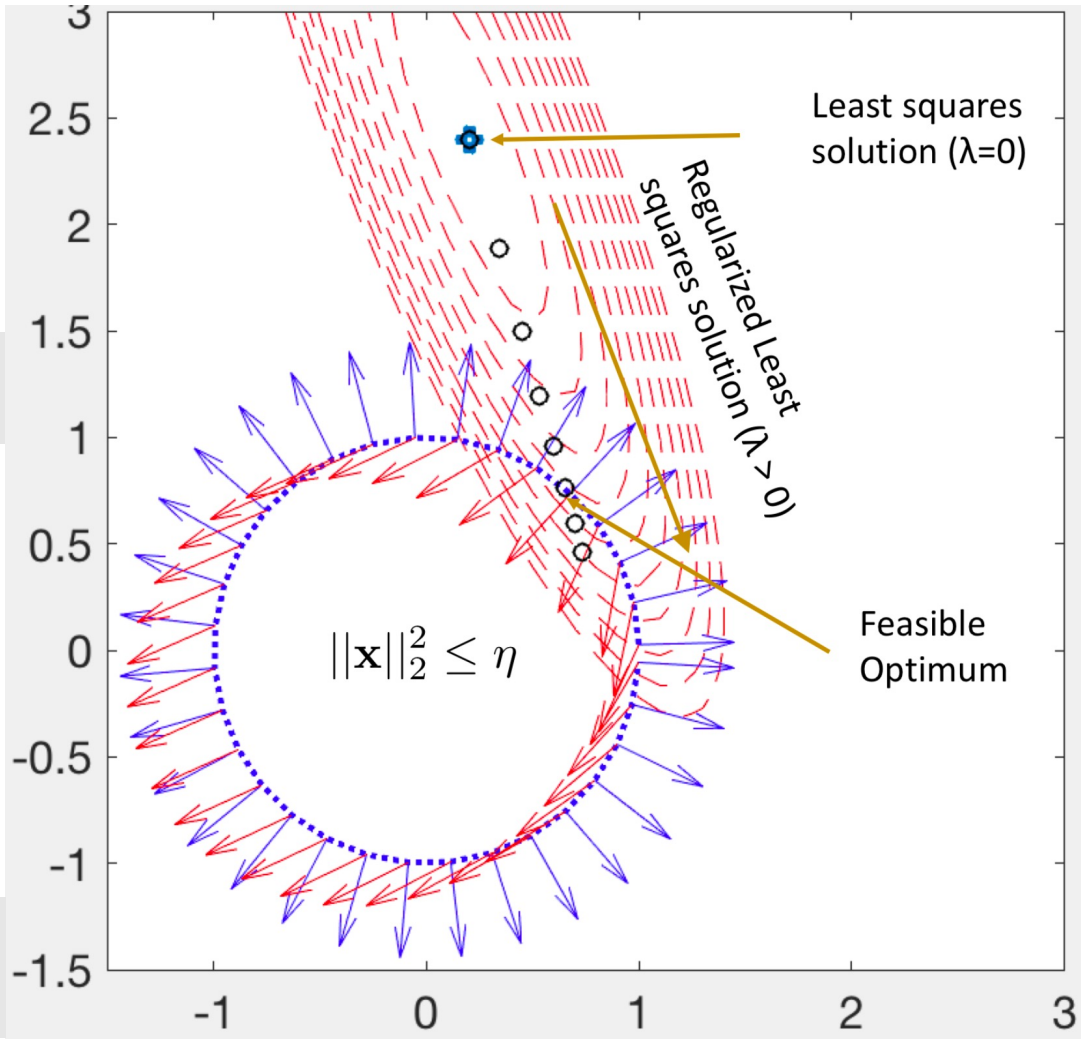
If $\mathbf{y} \in range(\mathbf{A})$ then we can satisfy $\mathbf{y} = \mathbf{A}\hat{\mathbf{x}}$ precisely.

Otherwise, we can satisfy $\mathcal{P}_{range(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\hat{\mathbf{x}}$.

What about regularized Least Squares?

$$\min_{\mathbf{x}_1} \ \mathcal{F}(\mathbf{x}_1, \lambda) = \|\mathbf{A}\mathbf{x_1} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}_1\|_2^2$$

$$\min_{\mathbf{x}_2} \ \|\mathbf{A}\mathbf{x_2} - \mathbf{y}\|_2^2 \ \ s.t. \ \ \|\mathbf{x}_2\|_2^2 \leq \eta$$



Consider $p(y|\theta) = \mathcal{N}(y; A\theta, C)$ with $A \in \mathbb{R}^{n\times m}$. Assume $p(\theta) = \mathcal{N}(\theta; \theta_0, B)$. Thus, $Y \sim \mathcal{N}(A\theta_0, ABA^T + C)$
To mimic a linear regression problem, consider the goal of estimating

$$\theta^* = \arg\min_{\theta} L(\theta)$$

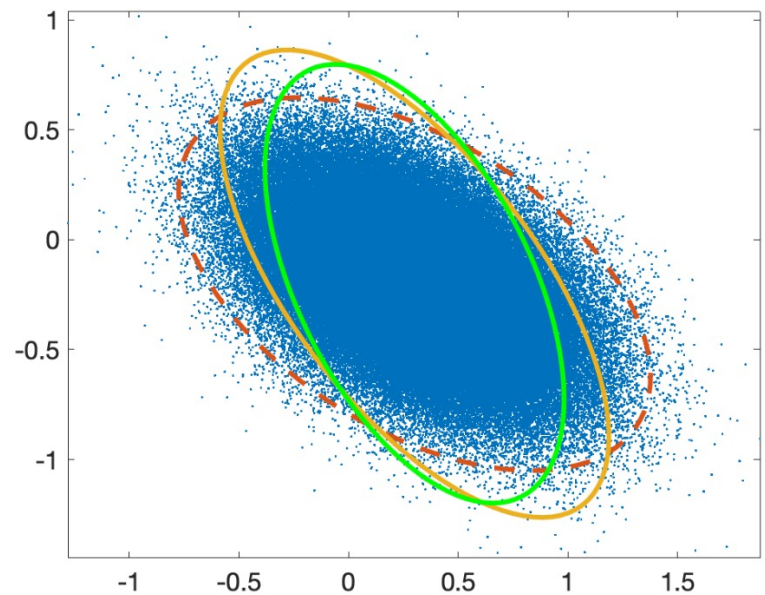
What about SGD?

where

$$L(\theta) = \mathbb{E}_Y[(y - A\theta)^T C^{-1}(y - A\theta)] + (\theta - \theta_0)^T B^{-1}(\theta - \theta_0)$$

SGD Converges to:

$$\theta_\mu = [A^T C^{-1} A + B^{-1}]^{-1}(A^T C^{-1}\mu_Y + B^{-1}\theta_0)$$
$$\theta_\Sigma = \alpha[A^T C^{-1} A + B^{-1}]^{-1} A^T C^{-1} J J^T C^{-1} A$$
$$= \alpha[A^T C^{-1} A + B^{-1}]^{-1} A^T C^{-1} \Sigma_Y C^{-1} A$$



Outline

SVD, QR decompositions

Least squares regression

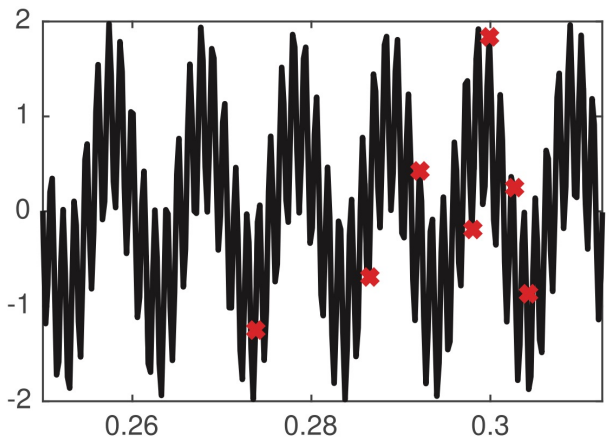
Sampling & Reconstruction

Sensing

Compression, Sensing and Reconstruction

It is not always possible/efficient to collect high-dimensional measurement data.

Is it possible to measure the quantity of interest at a few sensor locations and use these measurements to reconstruct the entire signal?



Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

Sampling and Reconstruction

Let's consider a signal $\boldsymbol{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\boldsymbol{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

Goal:

Instead of using the entire signal $\mathbf{x}$, we want to subsample $\mathbf{x}$, and see if we can reconstruct $\mathbf{x}$ using sparse measurements.

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

In order to subsample $\mathbf{x}$, let's define a matrix $\mathbf{P} \in \mathbb{R}^{p \times m}$, so that,

$$\mathbf{y} = \mathbf{P}\mathbf{x} \in \mathbb{R}^p$$

is the signal obtained by subsampling $\mathbf{x}$.

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

Example:

If $\mathbf{x} \in \mathbb{R}^5$, and we want to sample only the fourth and second measurements of $\mathbf{x}$, then matrix $\boldsymbol{P}$ takes the form,

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

In order to subsample $\mathbf{x}$, let's define a matrix $\boldsymbol{P} \in \mathbb{R}^{p \times m}$, such that y,

$$\mathbf{y} = \mathbf{P}\mathbf{x} \in \mathbb{R}^p$$

is the signal obtained by subsampling $\mathbf{x}$.

Therefore,

$$\mathbf{P}\mathbf{x} = \mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

We subsample $\mathbf{x}$ by,

$$\mathbf{P}\mathbf{x} = \mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

Therefore, the basis coefficients $\mathbf{a}$ can be estimated as,

$$\hat{\mathbf{a}} = [\mathbf{P}\Psi]^{+}\mathbf{y}$$

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

We subsample $\mathbf{x}$ by,

$$\mathbf{P}\mathbf{x} = \mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

Therefore, the basis coefficients $\mathbf{a}$ can be estimated as,

$$\hat{\mathbf{a}} = [\mathbf{P}\Psi]^{+}\mathbf{y}$$

And reconstruct $\mathbf{x}$ as,

$$\hat{\mathbf{x}} = \Psi[\mathbf{P}\Psi]^{+}\mathbf{y}$$

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

We subsample $\mathbf{x}$ by,

$$\mathbf{P}\mathbf{x} = \mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

And reconstruct $\mathbf{x}$ as,

$$\hat{\mathbf{x}} = \Psi[\mathbf{P}\Psi]^{+}\mathbf{y}$$

Note:

If the original signal had p non-zero coefficients, then p measurements reconstruct the signal exactly.

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

And reconstruct $\mathbf{x}$ as,

$$\hat{\mathbf{x}} = \Psi [\mathbf{P}\Psi]^+ \mathbf{y}$$

Questions:

- How do we know $\hat{\mathbf{a}}$ is sparse?
- What if we have noisy measurements?
- How to choose optimal sensor locations?

Sampling and Reconstruction

Let's consider a signal $\mathbf{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

And reconstruct $\mathbf{x}$ as,

$$\hat{\mathbf{x}} = \Psi [\mathbf{P}\Psi]^{+} \mathbf{y}$$

Questions:

- How do we know $\hat{\mathbf{a}}$ is sparse?
- What if we have noisy measurements?
- How to choose optimal sensor locations?
- How many measurements do we collect?

Sampling and Reconstruction

Questions:

- How do we know $\hat{\mathbf{a}}$ is sparse?

Sparse basis coefficients can be found by solving,

$$\hat{\mathbf{a}} = min_{\mathbf{a}} \ \ \|\mathbf{a}\|_1 \ \ such \ \ that \ \ \mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

This optimization problem is convex.

Sampling and Reconstruction

Questions:

- What if we have noisy measurements?

$$\mathbf{y} = \mathbf{P}\Psi\mathbf{a} + \epsilon$$

$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

Sparse basis coefficients can be found by solving,

$$\hat{\mathbf{a}} = min_{\mathbf{a}} \;\; \|\mathbf{a}\|_1 \;\; such \;\; that \;\; \|\mathbf{y} - \mathbf{P}\Psi\mathbf{a}\|_2 \leq \sigma$$

Sampling and Reconstruction

Questions:

- How to choose optimal sensor locations? (how to define matrix $\boldsymbol{P}$?)

Rule:

Rows of $\boldsymbol{P}$ have to be orthogonal to the columns of Ψ.

$$\boxed{\mathbf{P}\mathbf{x} = \mathbf{y} = \mathbf{P}\Psi\mathbf{a}}$$

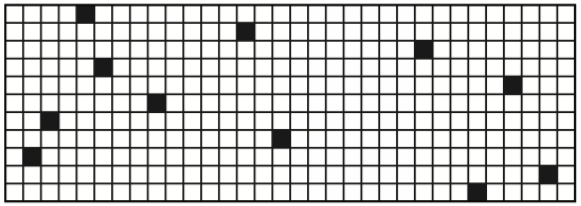
Sampling and Reconstruction

Questions:

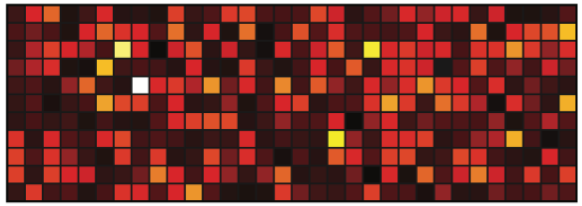
- How to choose optimal sensor locations? (how to define matrix $\boldsymbol{P}$?)

Possible choices for matrix $\boldsymbol{P}$,

(a) Random single pixel



(b) Gaussian random



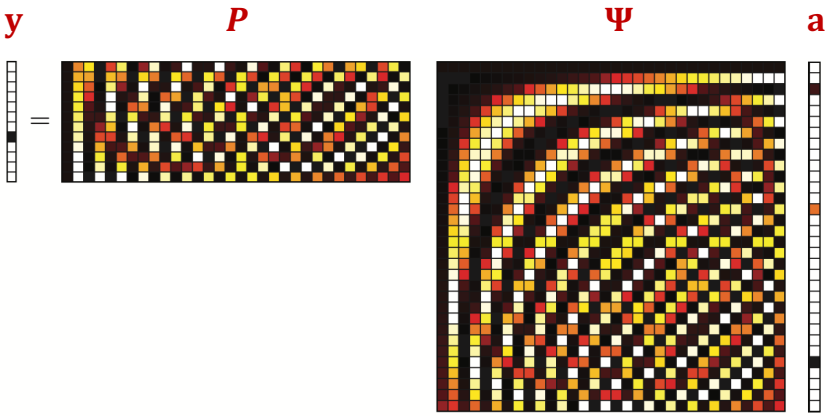
Brunton and Kutz, 2019

Sampling and Reconstruction

Questions:

- How to choose optimal sensor locations? (how to define matrix $\boldsymbol{P}$?)

An example of a bad choice for matrix $\boldsymbol{P}$,



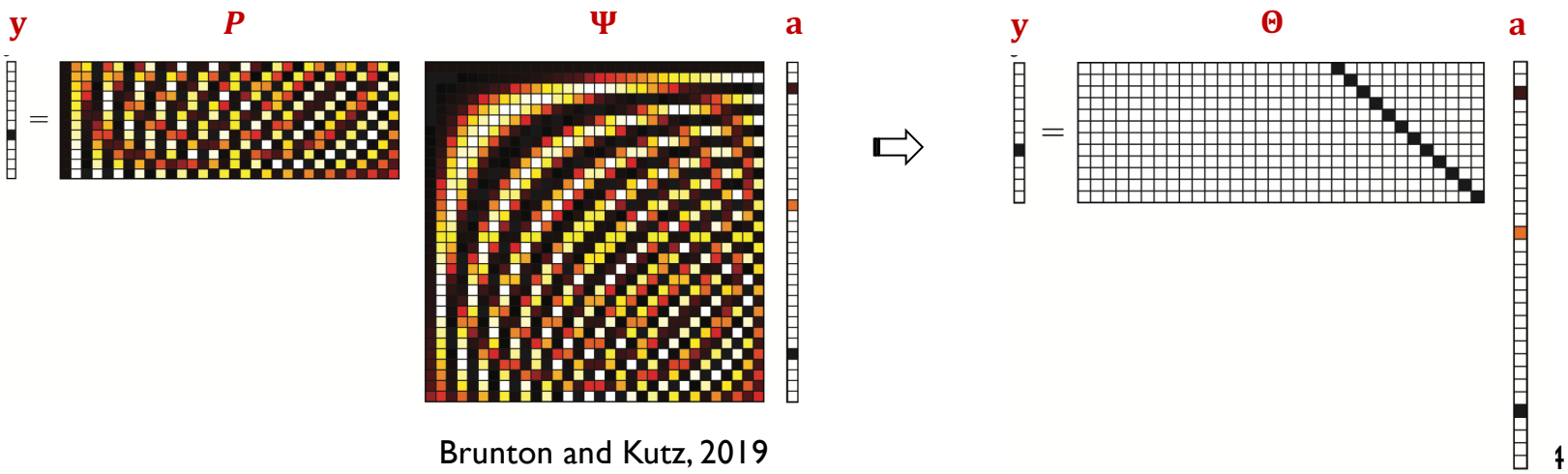
Brunton and Kutz, 2019

Sampling and Reconstruction

Questions:

- How to choose optimal sensor locations? (how to define matrix $\boldsymbol{P}$?)

An example of a bad choice for matrix $\boldsymbol{P}$,



Brunton and Kutz, 2019

Sampling and Reconstruction

Questions:

- How many measurements do we collect?

Theorem:

According to the Shannon-Nyquist sampling theorem, in order to recover a signal we should sample the signal twice the rate of its highest frequency.

Outline

SVD, QR decompositions

Least squares regression

Sampling & Reconstruction

Sensing

Sensing: Empirical Interpolation

Goal:

We want to design a sparse measurement matrix $\mathbf{P}$ such that the inverse problem is as well-conditioned as possible.

$$\mathbf{y} = \mathbf{P}\Psi\mathbf{a} \qquad \mathbf{a} \in \mathbb{R}^n$$

Sensing: Empirical Interpolation

Goal:

We want to design a sparse measurement matrix $\mathbf{P}$ such that the inverse problem is as well-conditioned as possible. $\mathbf{y} = \mathbf{P}\Psi\mathbf{a}$ $\mathbf{a} \in \mathbb{R}^n$

In general, we may use randomly placed sensors to estimate $\mathbf{a}$. However, when $p = n$ (the number of measurements equals the number of modes), $\mathbf{P}\Psi$ is often numerically singular.

Sensing: Empirical Interpolation

Goal:

We want to design a sparse measurement matrix $\mathbf{P}$ such that the inverse problem is as well-conditioned as possible. $\mathbf{y} = \mathbf{P}\Psi\mathbf{a}$ $\mathbf{a} \in \mathbb{R}^n$

In general, we may use randomly placed sensors to estimate $\mathbf{a}$. However, when $p = n$ (the number of measurements equals the number of modes), $\mathbf{P}\Psi$ is often numerically singular.

Solution:

- Oversampling
- Using QR factorization

Sensing: Empirical Interpolation

Goal:

We want to design a sparse measurement matrix $\mathbf{P}$ such that the inverse problem is as well-conditioned as possible.

$$\mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

Idea:

If we know the type of the signal, it is possible to design optimized sensors using low-rank features extracted from patterns in the data.

Given a basis Ψ, we can find a $\boldsymbol{P}^*$ by solving,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \Psi[\mathbf{P}\Psi]^+\mathbf{y}\|_2$$

Sensing: Empirical Interpolation

Idea:

If we know the type of he signal, it is possible to design optimized sensors using low-rank features extracted from patterns in the data.

Given a basis Ψ, we can find a $\boldsymbol{P}^*$ by solving,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \underbrace{\Psi[\mathbf{P}\Psi]^+\mathbf{y}}_{\text{Reconstruction of } \mathbf{x} \text{ using sparse basis coefficients}}\|_2$$

Sampling and Reconstruction

Let's consider a signal $\boldsymbol{x} \in \mathbb{R}^m$, represented by basis $\Psi \in \mathbb{R}^{m \times n}$ and basis coefficients $\mathbf{a} \in \mathbb{R}^n$,

$$\mathbf{x} = \Psi \mathbf{a}$$

We subsample $\boldsymbol{x}$ by,

$$\mathbf{P}\mathbf{x} = \mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

Therefore, the basis coefficients $\mathbf{a}$ can be estimated as,

$$\hat{\mathbf{a}} = [\mathbf{P}\Psi]^{+}\mathbf{y}$$

And reconstruct $\boldsymbol{x}$ as,

$$\hat{\mathbf{x}} = \Psi[\mathbf{P}\Psi]^{+}\mathbf{y}$$

Sensing: Empirical Interpolation

Idea:

If we know the type of the signal, it is possible to design optimized sensors using low-rank features extracted from patterns in the data.

Given a basis Ψ, we can find a $\boldsymbol{P}^*$ by solving,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{+}\mathbf{y}\|_2$$

If $p = n$, then,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{y}\|_2$$

Sensing: Empirical Interpolation

Idea:

If we know the type of the signal, it is possible to design optimized sensors using low-rank features extracted from patterns in the data.

Given a basis Ψ, we can find a $\boldsymbol{P}^*$ by solving,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{y}\|_2$$

For an orthonormal basis Ψ,

$$\|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{P}\mathbf{x}\|_2 \leq \|[\mathbf{P}\Psi]^{-1}\|_2 \|[\mathbf{I} - \Psi\Psi^T]\mathbf{x}\|_2$$

Sensing: Empirical Interpolation

Idea:

If we know the type of the signal, it is possible to design optimized sensors using low-rank features extracted from patterns in the data.

Given a basis Ψ, we can find a $\boldsymbol{P}^*$ by solving,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{y}\|_2$$

For an orthonormal basis Ψ,

$$\|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{P}\mathbf{x}\|_2 \leq \|[\mathbf{P}\Psi]^{-1}\|_2 \|\underbrace{[\mathbf{I} - \Psi\Psi^T]\mathbf{x}}_{\text{Projection error}}\|_2$$

Sensing: Empirical Interpolation

Idea:

If we know the type of the signal, it is possible to design optimized sensors using low-rank features extracted from patterns in the data.

Given a basis Ψ, we can find a $\boldsymbol{P}^*$ by solving,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{y}\|_2$$

For an orthonormal basis Ψ,

$$\|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{P}\mathbf{x}\|_2 \leq \overbrace{\|[\mathbf{P}\Psi]^{-1}\|_2}^{\text{Sampling error}} \underbrace{\|[\mathbf{I} - \Psi\Psi^T]\mathbf{x}\|_2}_{\text{Projection error}}$$

Sensing: Discrete Empirical Interpolation

Given a basis Ψ, we can find a $\boldsymbol{P}^*$ by solving,

$$\mathbf{P}^* = min_{\mathbf{P}} \|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{y}\|_2$$

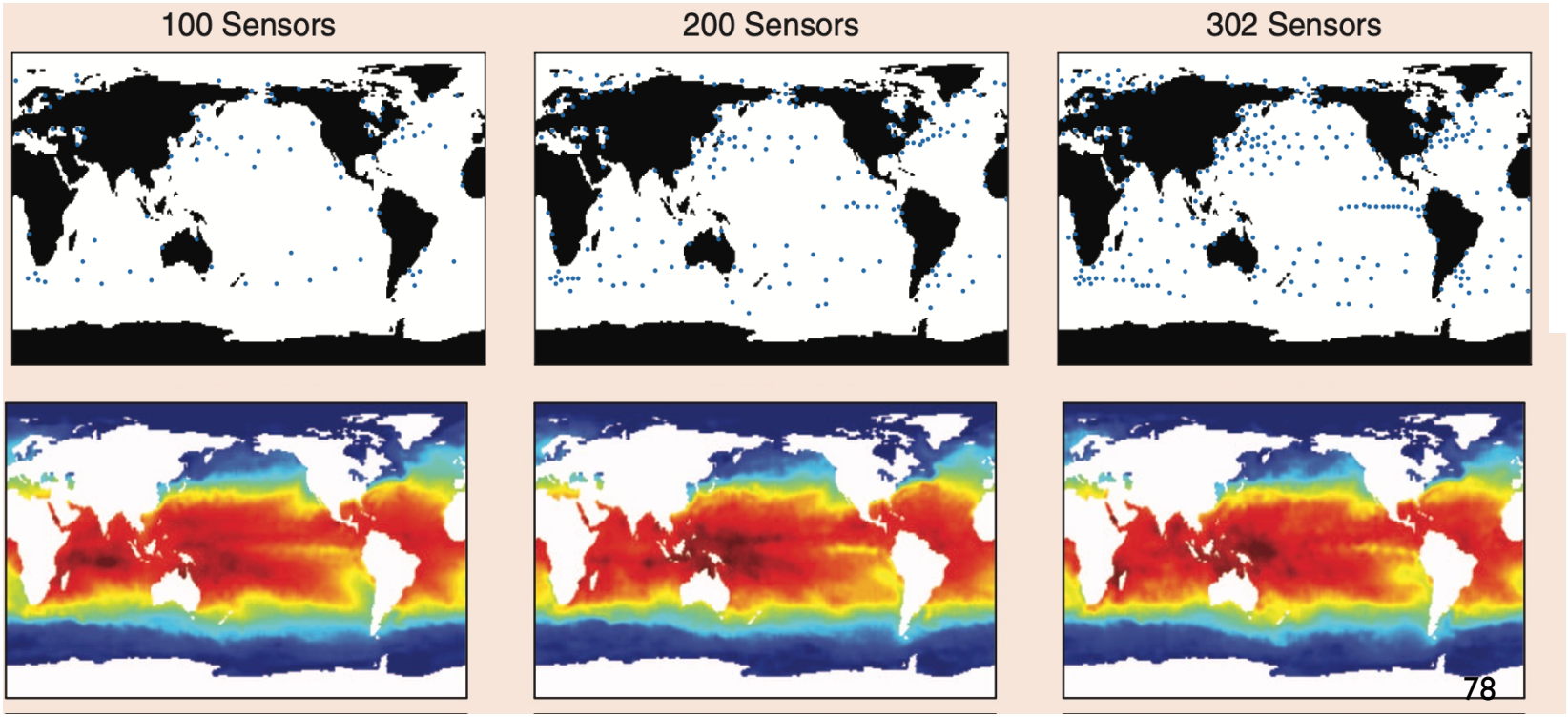
For an orthonormal basis Ψ,

$$\|\mathbf{x} - \Psi[\mathbf{P}\Psi]^{-1}\mathbf{P}\mathbf{x}\|_2 \leq \|[\mathbf{P}\Psi]^{-1}\|_2 \|[\mathbf{I} - \Psi\Psi^T]\mathbf{x}\|_2$$

Therefore, the sensor selection process reduces to,

$$\mathbf{P}^* = min_{\mathbf{P}} \|[\mathbf{P}\Psi]^{-1}\|_2$$

Sea Surface temperature dataset (Manohar et al.)



Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix (sensor locations).

Why QR factorization?

Reminder: we want to design a sparse measurement matrix $\mathbf{P}$ such that the inverse problem is as well-conditioned as possible.

$$\mathbf{y} = \mathbf{P}\Psi\mathbf{a}$$

QR factorization with pivoting, maximizes the submatrix volume and controls the condition number.

Sensing: QDEIM

Side Note: QR factorization with column pivoting

QR decomposition with column pivoting decomposes a matrix into a unitary matrix $\mathbf{Q}$, an upper triangular matrix $\mathbf{R}$, and a column permutation matrix Φ,

$$\mathbf{W}\Phi = \mathbf{Q}\mathbf{R}$$

Where, $\mathbf{Q} \in \mathbb{C}^{n \times n}$ and $\mathbf{R} \in \mathbb{C}^{n \times m}$.

Sensing: QDEIM

Side Note: QR factorization with column pivoting

Given $\mathbf{W} \in \mathbb{R}^{n \times m}$, with a column pivoted QR decomposition we have,

$$\mathbf{W}\Phi = \mathbf{Q}\mathbf{R}$$

Where, $\mathbf{Q} \in \mathbb{C}^{n \times n}$ and $\mathbf{R} \in \mathbb{C}^{n \times m}$, therefore,

$$\mathbf{W}\Phi = \mathbf{Q}[\mathbf{R}_1 \quad \mathbf{R}_2]$$

$\boldsymbol{R_1} \in \mathbb{C}^{n \times n}$ is an upper triangular matrix.

The columns of $\mathbf{W}$ are permuted such that the diagonal elements of $\boldsymbol{R_1}$ are non-increasing.

Sensing: QDEIM

Side Note: QR factorization with column pivoting

Given $\mathbf{W} \in \mathbb{R}^{n \times m}$, with a column pivoted QR decomposition we have,

$$\mathbf{W}\Phi = \mathbf{Q}\mathbf{R}$$

Where, $\mathbf{Q} \in \mathbb{C}^{n \times n}$ and $\mathbf{R} \in \mathbb{C}^{n \times m}$, therefore,

$$\mathbf{W}\Phi = \mathbf{Q}[\mathbf{R}_1 \quad \mathbf{R}_2]$$

$\boldsymbol{R_1} \in \mathbb{C}^{n \times n}$ is an upper triangular matrix.

Also,

$$\sigma_i^2 = |r_{ii}|^2 \quad ; \quad 1 \leq i \leq n$$

Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix (sensor locations).

$$\mathbf{W}\Phi = \mathbf{QR}$$

Let's set $\mathbf{W} = \Psi^T$, and

$$\mathbf{P}\Psi = [\mathbf{QR}_1]^T = \mathbf{R}_1^T\mathbf{Q}^T$$

We want to solve

$$\mathbf{P}^* = min_{\mathbf{P}}\|[\mathbf{P}\Psi]^{-1}\|_2$$

Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix (sensor locations).

Let's set $\mathbf{W} = \Psi^T$, and

$$\mathbf{P}\Psi = [\mathbf{Q}\mathbf{R}_1]^T = \mathbf{R}_1^T\mathbf{Q}^T$$

We want to solve

$$\mathbf{P}^* = min_{\mathbf{P}}\|[\mathbf{P}\Psi]^{-1}\|_2$$

Therefore,

$$\|\mathbf{P}\Psi\|_2 = \|\mathbf{R}_1^T\mathbf{Q}^T\|_2 = \sigma_{max}(\mathbf{R}_1)$$

Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix (sensor locations).

We want to solve

$$\mathbf{P}^* = min_{\mathbf{P}} \|[\mathbf{P}\Psi]^{-1}\|_2$$

Therefore,

$$\|\mathbf{P}\Psi\|_2 = \|\mathbf{R}_1^T \mathbf{Q}^T\|_2 = \sigma_{max}(\mathbf{R}_1)$$

and,

$$\|[\mathbf{P}\Psi]^{-1}\|_2 = \frac{1}{\sigma_{min}(\mathbf{R}_1)}$$

Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix (sensor locations).

We want to solve

$$\mathbf{P}^* = min_{\mathbf{P}} \|[\mathbf{P}\Psi]^{-1}\|_2$$

Therefore,

$$\|\mathbf{P}\Psi\|_2 = \|\mathbf{R}_1^T \mathbf{Q}^T\|_2 = \sigma_{max}(\mathbf{R}_1)$$

and,

$$\|[\mathbf{P}\Psi]^{-1}\|_2 = \frac{1}{\sigma_{min}(\mathbf{R}_1)}$$

We should keep $\sigma_{min}(\mathbf{R}_1)$ as large as possible.

Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix (sensor locations).

We want to minimize sampling error,

$$\|[\mathbf{P}\Psi]^{-1}\|_2 = \frac{1}{\sigma_{min}(\mathbf{R}_1)}$$

We should keep $\sigma_{min}(\mathbf{R}_1)$ as large as possible.

QR factorization with column pivoting expands the submatrix volume by enforcing a diagonal dominance structure (Manohar, 2018),

$$\sigma_i^2 = |r_{ii}|^2 \geq \sum_{j=i}^{k} |r_{jk}|^2; \quad 1 \leq i \leq k \leq m$$

Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix (sensor locations).

Let's set $\mathbf{W} = \Psi^T$ in

$$\mathbf{W}\Phi = \mathbf{Q}[\mathbf{R}_1 \quad \mathbf{R}_2]$$

QDEIM solves the column pivoted QR factorization

$$\boldsymbol{\Psi}^T\boldsymbol{\Phi} = \mathbf{Q}\mathbf{R}_1$$

and the optimal sampling matrix is,

$$\mathbf{P}^* = \boldsymbol{\Phi}^T$$

Sensing: QDEIM

The QDEIM approach uses QR factorization to obtain the sampling matrix $\mathbf{C}$.

```
if (p==r)     % QR sensor selection, p=r
    [Q,R,pivot] = qr(Psi_r','vector');
elseif (p>r) % Oversampled QR sensors, p>r
    [Q,R,pivot] = qr(Psi_r*Psi_r','vector');
end
C = zeros(p,n);
for j=1:p
    C(j,pivot(j))=1;
end
```

$$\mathbf{\Psi}_r^T \mathbf{C}^T = \mathbf{QR}$$

$$(\mathbf{\Psi}_r \mathbf{\Psi}_r^T)\mathbf{C}^T = \mathbf{QR}$$

Brunton and Kutz, 2019

p: number of sensors
r: number of basis functions $\mathbf{a} \in \mathbb{R}^r$