MOR – 2023 – MS^{2}SC

Practical work of model reduction and data compression
Session 1 : data compression (a-posteriori)

Laboratoire de Mécanique Paris-Saclay

Victor Matray

POD approximation of a matrix

600

600

$\underline{\underline{M}}$ *(gray level matrix)*

POD by $\underbrace{\text{Singular Values}}$ Decomposition (SVD)

$$\underline{\underline{M}} = \underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^T$$

left eigenvectors

right eigenvectors

Singulars values

$$\underset{m \times n}{\underline{\underline{M}}} = \underset{m \times m}{\underline{\underline{U}}} \times \underset{m \times n}{\overset{\underline{\underline{S}}}{(0)}} \times \underset{n \times n}{\underline{\underline{V}}^T}$$

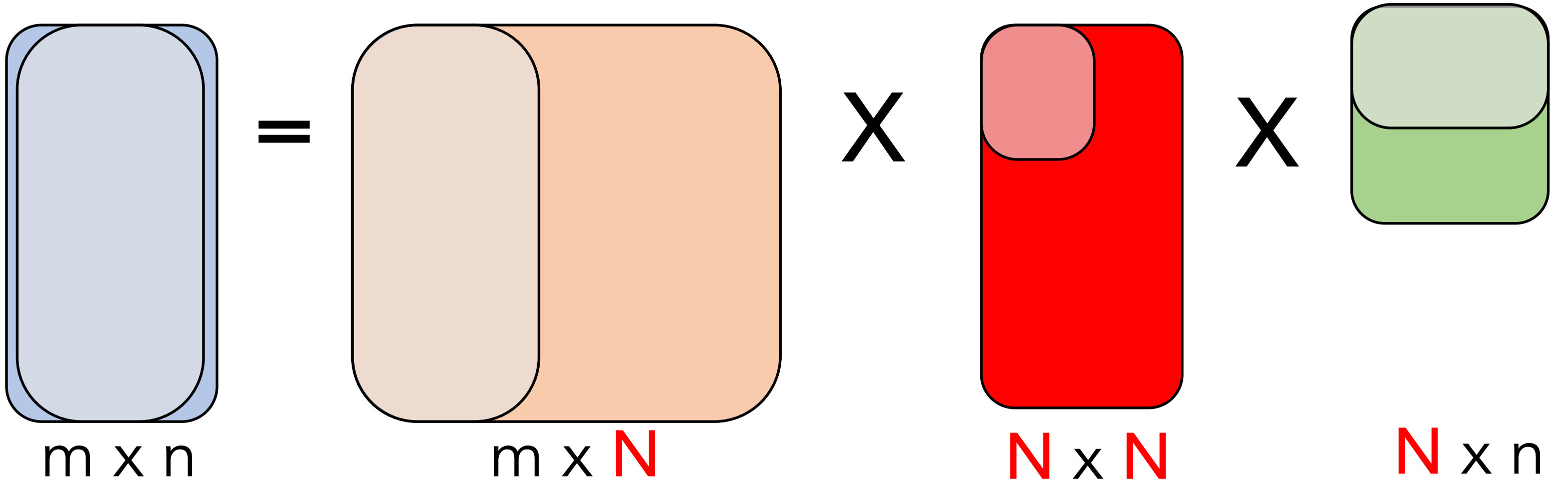
POD approximation of a matrix

On Matlab :

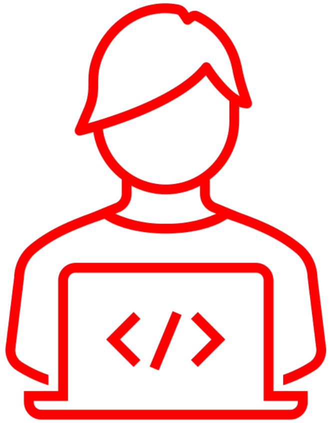
```
[U,S,V] = svd(M);
```

Choose a number of modes : N

With a few modes we can recontruct the original image : $\underline{\underline{M}}_{\text{POD}} = \sum_{i=1}^{N} S_i \underline{U}_i \underline{V}_i^\top$



POD approximation of a matrix



1. Calculate for a number N of POD modes the storage gain achieved by this approach.

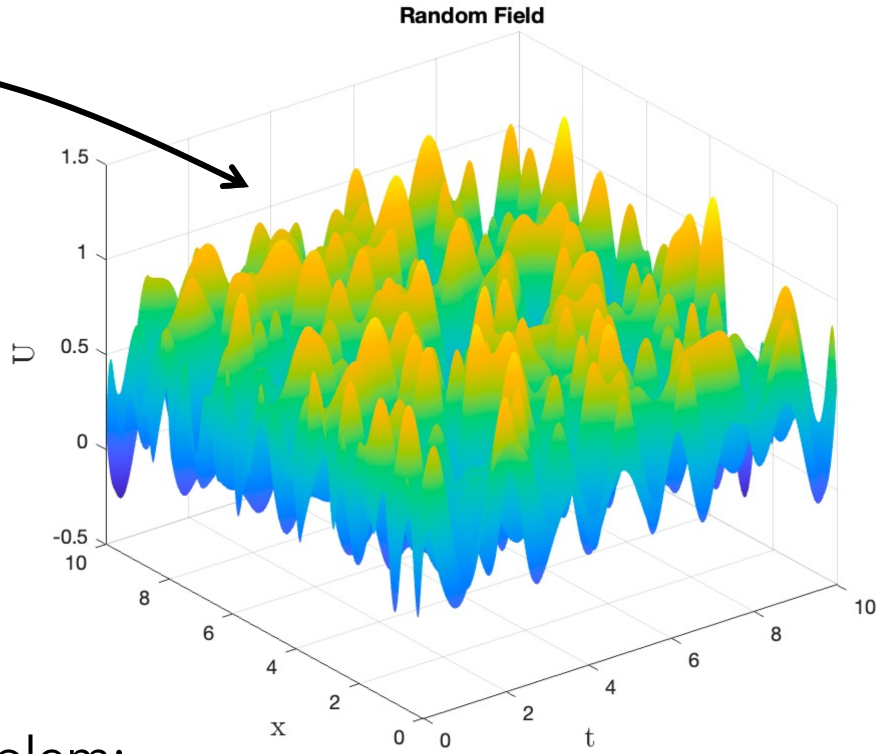
$$\frac{\|\underline{\underline{M}}^{\text{POD}} - \underline{\underline{M}}\|}{\|\underline{\underline{M}}\|}$$

2. Plot the evolution of the relative reconstruction error as a function of the number of modes.

3. Deduce an appropriate number of modes and the effective gain achieved with the previous question.

Random space-time field

$$\begin{array}{rcl} U : \mathcal{W} \times \mathcal{I} & \longrightarrow & \mathbb{R} \\ (x,t) & \mapsto & U(x,t) \end{array}$$



Matrix $N_x \times N_t$

vector $N_x \times 1$

vector $1 \times N_t$

$$\underline{\underline{U}}(x,t) \approx \underline{\underline{U}}_m(x,t) = \sum_{i=1}^{m} \underline{\Lambda_i}(x)\underline{\Gamma_i}^\top(t)$$

number of modes

We are looking for the solution of the following minimization problem:

$$\begin{cases} \underline{\Gamma_i}, \underline{\Lambda_i} = \underset{\mathcal{I}\times\mathcal{W}}{\arg\min} \left(\left\| \underline{\underline{U}} - \underline{\underline{U}}_{\mathrm{PGD}}^{i-1} - \underline{\Lambda_i}\underline{\Gamma_i}^\top \right\|^2 \right) \\ \underline{\underline{U}}_{\mathrm{PGD}}^{i-1} = \displaystyle\sum_{k=1}^{i-1} \underline{\Lambda_k}(x)\underline{\Gamma_k}^\top(t) \end{cases} \longrightarrow \begin{cases} \text{Problem on } \underline{\Lambda_i} \\ \text{Fixed point algorithm} \\ \text{Problem on } \underline{\Gamma_i} \end{cases}$$

cf. your lesson !

Calculation of integrals by the finite element method

How to compute $\displaystyle\int_0^L a(x)\cdot b(x)dx$ **?**

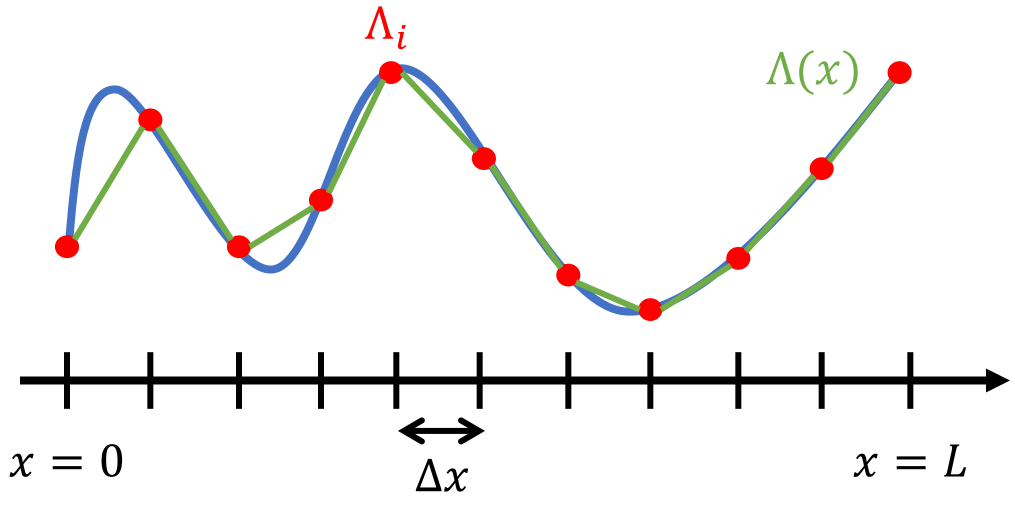
(We apply the same approach for the *temporal integrals*.)

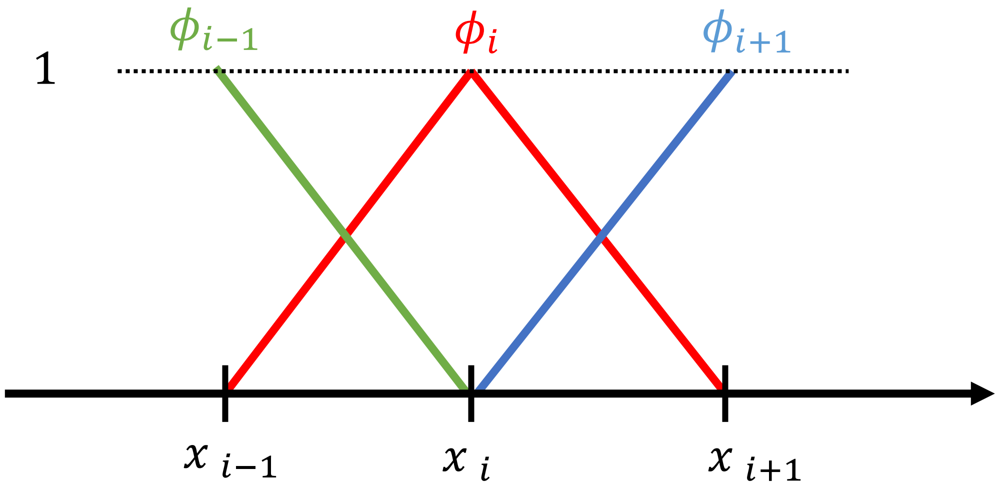
The $\underline{\Lambda}$ and $\underline{\Gamma}$ are discrete fields defined at points.

To compute the integrals we will interpolate these discrete fields using **shape functions** (in the same way as for the finite element method). For this tutorial we will use the simplest shape functions : **piecewise linear**.

In this way, we have :

$$\underline{\Lambda}(x) = \sum_{i=1}^{N_x} \phi_i(x) \times \Lambda_i \text{ with } \begin{cases} \Lambda_i = \Lambda(x_i) \\ x_i = i \times \Delta x \end{cases}$$





In our case the shape functions are written as :

$$\begin{cases} \phi_i\left(x_i\right) = 1 \\ \phi_i(x) = \frac{x - x_{i-1}}{\Delta x} \text{ if } x \in [x_{i-1}; x_i] \\ \phi_i(x) = \frac{x_{i+1} - x}{\Delta x} \text{ if } x \in [x_i; x_{i+1}] \\ 0 \quad \text{otherwise} \end{cases}$$

Thus, with the use of these shape functions, the integral of a product of functions is written as :

$$\int_0^L a(x) \cdot b(x) dx = \sum_{i=0}^{N_x - 1} \int_{x_i}^{x_{i+1}} a(x) \cdot b(x) dx$$

With :

$$a(x) = \sum_{j=1}^{N_x} \phi_j(x) a_j$$

$$b(x) = \sum_{j=1}^{N_x} \phi_j(x) b_j$$

Taking advantage of the fact that only functions of form ϕ_i and ϕ_{i+1} are nonzero on the interval $[x_i; x_{i+1}]$ (they are identically zero everywhere outside) the previously defined integral becomes :

$$\sum_{i=0}^{N_x - 1} \int_{x_i}^{x_{i+1}} (a_i \phi_i + a_{i+1}\phi_{i+1}) \times (b_i \phi_i + b_{i+1}\phi_{i+1}) \, dx$$

in algebraic form :

$$\begin{pmatrix} a_i \\ a_{i+1} \end{pmatrix}^T \cdot \begin{pmatrix} \phi_i^2 & \phi_i \phi_{i+1} \\ \phi_i \phi_{i+1} & \phi_{i+1}^2 \end{pmatrix} \cdot \begin{pmatrix} b_i \\ b_{i+1} \end{pmatrix}$$

constant

$$\int_{x_i}^{x_{i+1}} \bullet \, dx = \begin{pmatrix} a_i \\ a_{i+1} \end{pmatrix}^T \cdot \underbrace{\begin{pmatrix} \int_{x_i}^{x_{i+1}} \phi_i^2 dx & \int_{x_i}^{x_{i+1}} \phi_i \phi_{i+1} dx \\ \int_{x_i}^{x_{i+1}} \phi_i \phi_{i+1} dx & \int_{x_i}^{x_{i+1}} \phi_{i+1}^2 dx \end{pmatrix}}_{\underline{\underline{m}}} \cdot \begin{pmatrix} b_i \\ b_{i+1} \end{pmatrix}$$

constant

(mass) elementary matrix

Let us note :

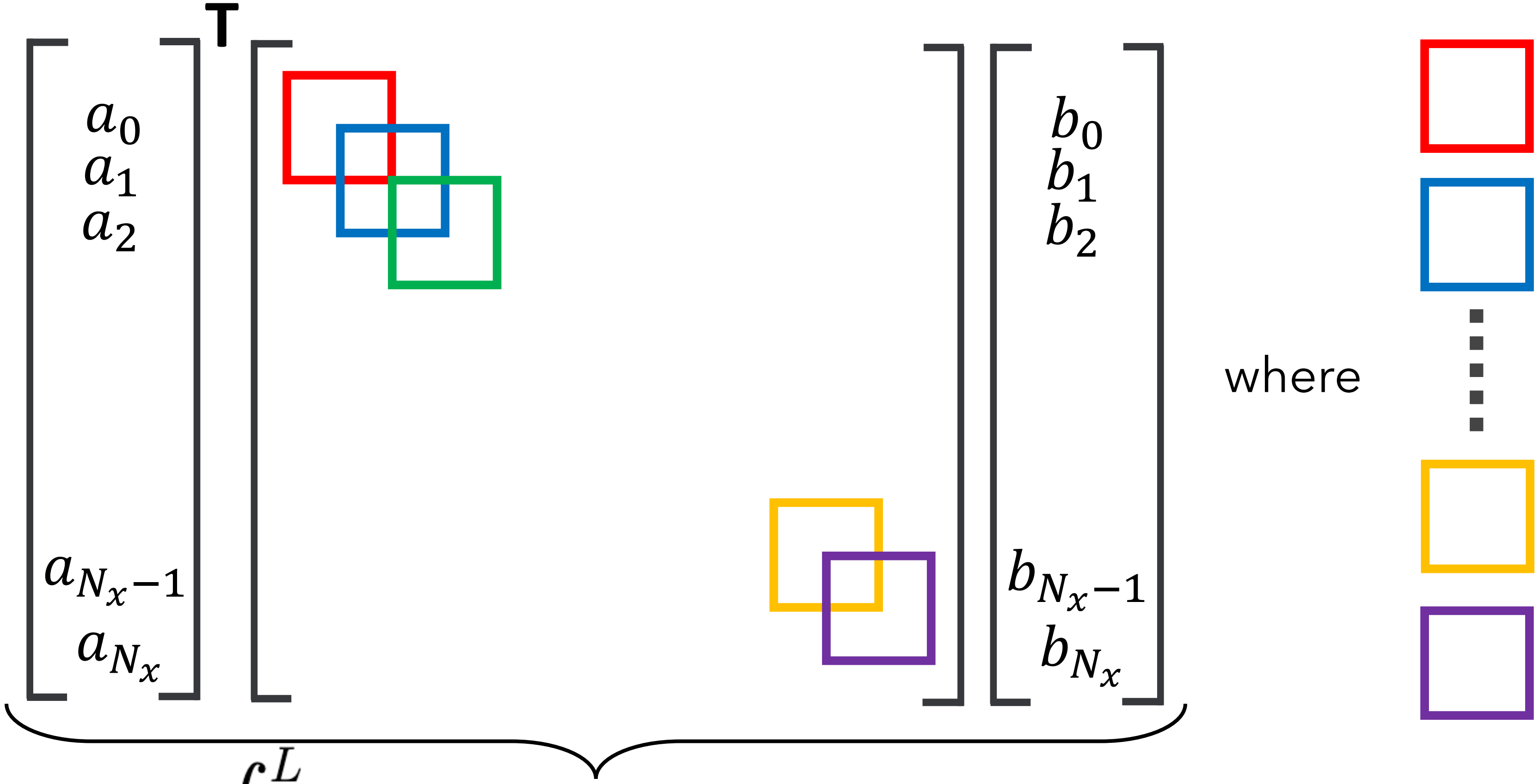
$$\int_{x_i}^{x_{i+1}} \phi_i^2 dx = \int_{x_i}^{x_{i+1}} \phi_{i+1}^2 dx$$

Show that :

$$\underline{\underline{m}} = \Delta x \cdot \begin{pmatrix} 1/3 & 1/6 \\ 1/6 & 1/3 \end{pmatrix}$$

Calculation of integrals by the finite element method

(Here, we have $N_x + 1$ nodes and N_x elements…)



$$= \frac{\Delta x}{6}\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\int_0^L a(x)\cdot b(x)dx = \underline{a}^\top \underline{\underline{I}}_x \underline{b}$$

<u>In this case we have two strategies:</u>

- either we assemble the elementary matrices (cf. FEM)
- or we build directly the operator in one shot (faster and easier by taking advantage of the matlab functions : *diag* & *ones*)

Fixed point algorithm

Pseudo algorithm for PGD approximation (Fixed Point) :

1. we compute : $\underline{\underline{u}}^* \leftarrow \underline{\underline{u}}_f - \underline{\underline{u}}^{i-1}_{PGD}$
2. we initialize : $\underline{\Gamma}_i^{\text{new}} \leftarrow \text{linspace}(0, T, N_T)^T$
3. we compute : $\underline{\Lambda}_i^{\text{new}} \leftarrow \frac{\int_I \underline{\underline{u}}^*(x,t)\underline{\Gamma}_i^{\text{new}}dt}{\int_I (\underline{\Gamma}_i^{\text{new}})^2 dt}$
4. we normalize (for uniqueness) : $\underline{\Lambda}_i^{\text{new}} \leftarrow \frac{\underline{\Lambda}_i^{\text{new}}}{\sqrt{\int_{\mathcal{W}} (\underline{\Lambda}_i^{\text{new}})^2 dx}}$
5. $\underline{\Gamma}_i^{\text{old}} \leftarrow \underline{\Gamma}_i^{\text{new}}$
6. we compute : $\underline{\Gamma}_i^{\text{new}} \leftarrow \frac{\int_{\mathcal{W}} \underline{\underline{u}}^*(x,t)\underline{\Lambda}_i^{\text{new}}dx}{\int_{\mathcal{W}} (\underline{\Lambda}_i^{\text{new}})^2 dx}$
7. calculation of the stagnation criterion : $s = \frac{\int_I \left(\underline{\Gamma}_i{}^{\text{new}} - \underline{\Gamma}_i{}^{\text{old}}\right)^2 dt}{\int_I \left(\underline{\Gamma}_i{}^{\text{old}}\right)^2 dt}$
8. If $s < threshold$:
   - We save the modes : $\underline{\Lambda}_i \leftarrow \underline{\Lambda}_i^{\text{new}}$ et $\underline{\Gamma}_i \leftarrow \underline{\Gamma}_i^{\text{new}}$
   - $\underline{\underline{U}}_{\text{PGD}} = \underline{\underline{U}}_{\text{PGD}} + \underline{\Gamma}_i \times \underline{\Lambda}_i^T$
   - we update the field to approximate : $\underline{\underline{u}}^* \leftarrow \underline{\underline{u}}^* - \underline{\Gamma}_i \times \underline{\Lambda}_i^T$
   - we are looking for the next mode $(i+1)$

   If **NOT**

   - we return to step 3 until convergence of the fixed point !

field to approximate

It's a choice !

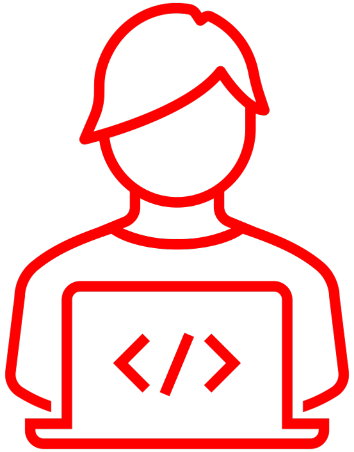
N_x integrations on time

space interval

N_t integrations on space

time interval

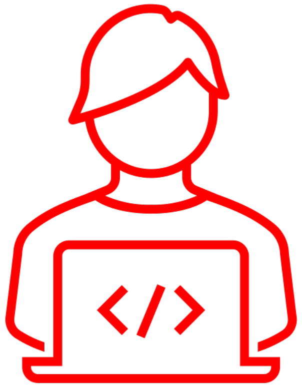
It's a choice, begin with : 10^{-3}



Fixed point

Computation of the space-time error

$$\text{Error} = \frac{\|\underline{\underline{U}}^{\text{ref}} - \underline{\underline{U}}^{\text{PGD}}\|_{\mathcal{U}}}{\|\underline{\underline{U}}^{\text{ref}}\|_{\mathcal{U}}}$$



As a reminder : $\mathcal{U} = \mathcal{W} \times \mathcal{I}$.

We also define :

$$\| \bullet \|_{\mathcal{U}} = \left(\int_{\mathcal{W}\times\mathcal{I}} \bullet^2 \, \mathrm{d}\Omega \, \mathrm{d}t \right)^{1/2}$$

So :

$$\text{Error} = \frac{\int_{\mathcal{W}} \int_{\mathcal{I}} \left(\underline{\underline{U}}^{\text{ref}} - \underline{\underline{U}}^{\text{PGD}} \right)^2 \, \mathrm{d}t \, \mathrm{d}\Omega}{\int_{\mathcal{W}} \int_{\mathcal{I}} \left(\underline{\underline{U}}^{\text{ref}} \right)^2 \, \mathrm{d}t \, \mathrm{d}\Omega}$$

How to calculate this error numerically?

1. First we integrate on $\mathcal{I}$, we obtain a vector of size N_x
2. Then we integrate on $\mathcal{W}$, we get a **scalar**.

Plot the evolution of the relative reconstruction error as a function of the number of modes.

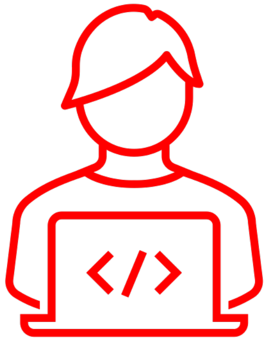
Adding the orthogonalization of spatial modes

WHY ORTHOGONALIZE THE MODES ?

- to avoid redundancy of information between modes
- this will be useful for TP 2 for the "update phase"

HOW TO ORTHOGONALIZE THE MODES ?

- Gram-Schmidt algorithm



Scalar product based on space operator here

We want : $\left(\underline{\Lambda}_i, \underline{\Lambda}_j\right) = \delta_{ij}$.

So, for : $\underline{\Lambda}_{m+1}^{\text{new}}$, we write :

$$\underline{\Lambda}_{m+1} = \underline{\Lambda}_{m+1}^{\text{new}} - \sum_{i=1}^{m} \underbrace{\left(\underline{\Lambda}_{m+1}^{\text{new}}, \underline{\Lambda}_i\right)}_{=\sqrt{\int_{\mathcal{W}} \dots dx}} \cdot \underline{\Lambda}_i$$

This step consists in removing from $\underline{\Lambda}_{m+1}^{\text{new}}$ its different projections on the $\underline{\Lambda}_i, \forall i \in [1, m]$.

In this way, we obtain : $\left(\underline{\Lambda}_{m+1}, \underline{\Lambda}_i\right) = 0, \forall i \in [1, m]$.

However, as we "remove" information at the level of $\underline{\Lambda}_{m+1}$, we have to "add" it in the $\underline{\Gamma}$. It then comes :

$$\underline{\Gamma}_i \leftarrow \underline{\Gamma}_i + \underline{\Gamma}_{m+1} \cdot \left(\underline{\Lambda}_{m+1}^{new}, \underline{\Lambda}_i\right), \forall i \in [1, m]$$

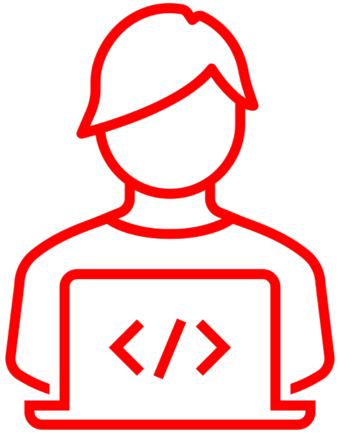
We must finally normalize :

$$\underline{\Lambda}_i^{\text{new}} \leftarrow \frac{\underline{\Lambda}_{m+1}^{\text{new}}}{\sqrt{\int_{\mathcal{W}} (\underline{\Lambda}_{m+1})^2 dx}}$$

$$\underline{\Gamma}_i^{\text{new}} \leftarrow \underline{\Gamma}_{m+1}^{\text{new}} \times \sqrt{\int_{\mathcal{W}} (\underline{\Lambda}_{m+1})^2 dx}$$

Back to the image

Implement the PGD algorithm for the approximation of the Maria Callas image.



<u>To go further:</u> apply these different methods to color images of your choice