Optimal control using reduced-order models

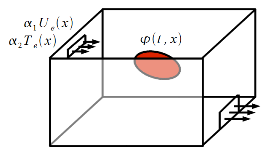
Cyrille Allery, Claudine Béghein, Antoine Dumon, Mourad Oulghelou, Alexandra Tallet

Laboratoire des Sciences de l'Inégnieur pour l'Environnement

(LaSIE, UMR CNRS 7356), La Rochelle Université

Objective

- control indoor air quality :
  - keep a velocity or temperature profile in a room
  - evacuate a pollutant



by acting on the intensity and temperature of the injected air or on external sources

- optimize comfort or heating consumption in a design phase

Strategy : formulation as an optimal control problem

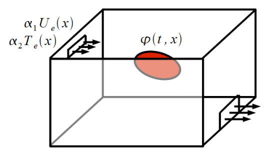
- minimize the functional $J(\varphi, \gamma) = \frac{1}{2} \int_0^T \int_\Omega (\varphi - \hat{\varphi})^2 d\Omega\, dt + \frac{\kappa}{2}\gamma^2$

where φ can be the velocity, the temperature or the pollutant concentration
$\hat{\varphi}$ is the target and γ the control parameter

under the constraints of the Navier-Stokes equ.

Objective

- control indoor air quality :
  - keep a velocity or temperature profile in a room
  - evacuate a pollutant



by acting on the intensity and temperature of the injected air or on external sources

- optimize comfort or heating consumption in a design phase

Strategy : formulation as an optimal control problem

- minimize the functional $J(\varphi,\gamma) = \frac{1}{2}\int_0^T\int_\Omega (\varphi - \hat{\varphi})^2 d\Omega\, dt + \frac{\kappa}{2}\gamma^2$

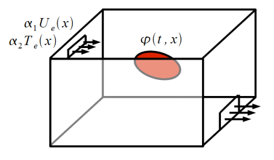
where φ can be the velocity, the temperature or the pollutant concentration

$\hat{\varphi}$ is the target and γ the control parameter

under the constraints of the Navier-Stokes equ.

Objective

- control indoor air quality :
  - keep a velocity or temperature profile in a room
  - evacuate a pollutant



by acting on the intensity and temperature of the injected air or on external sources

- optimize comfort or heating consumption in a design phase

Strategy : formulation as an optimal control problem

- minimize the functional $J(\varphi, \gamma) = \frac{1}{2}\int_0^T \int_\Omega (\varphi - \hat{\varphi})^2 d\Omega \, dt + \frac{\kappa}{2}\gamma^2$

where φ can be the velocity, the temperature or the pollutant concentration
$\hat{\varphi}$ is the target and γ the control parameter

under the constraints of the Navier-Stokes equ.

Problem statement

- incompressible and anisothermal flow subject to gravity $\boldsymbol{g} = -g\boldsymbol{e_y}$
- based on Boussinesq hypothesis, the pb can be written :

$$\begin{cases} \nabla \cdot \boldsymbol{u} = 0 \\ \dfrac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} = -\nabla p + \dfrac{1}{Re} \nabla^2 \boldsymbol{u} + Ri\, \theta\, \mathbf{e_y} \\ \dfrac{\partial \theta}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \theta = \dfrac{1}{RePr} \nabla^2 \theta \end{cases}$$

- we assume that there exists a part of the boundary, denoted Γ_u (resp. Γ_θ), where the velocity (resp. the temp.) can be modified :

$$\boldsymbol{u}|_{\Gamma_u} = \gamma_1\, \boldsymbol{u}_\Gamma(\mathbf{x}) \qquad \theta|_{\Gamma_\theta} = \gamma_2\, \theta_\Gamma(\mathbf{x})$$

- starting from an initial flow, we want to achieve a temperature $\hat{\theta}$ in $\Omega_c \subset \Omega$ associated to control parameters $\hat{\gamma}$
- the objective functional is

$$\mathcal{J}(\theta, \gamma) = \frac{1}{2} \int_0^T \int_{\Omega_c} |\theta - \hat{\theta}|^2 d\Omega dt + \frac{1}{2} \int_{\Omega_c} |\theta_T - \hat{\theta}_T|^2 d\Omega + \frac{\kappa}{2} |\gamma|^2$$

Problem statement

- incompressible and anisothermal flow subject to gravity $\boldsymbol{g} = -g\boldsymbol{e_y}$
- based on Boussinesq hypothesis, the pb can be written :

$$\begin{cases} \nabla \cdot \boldsymbol{u} = 0 \\ \dfrac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} = -\nabla p + \dfrac{1}{Re} \nabla^2 \boldsymbol{u} + Ri\, \theta\, \mathbf{e_y} \\ \dfrac{\partial \theta}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \theta = \dfrac{1}{RePr} \nabla^2 \theta \end{cases}$$

- we assume that there exists a part of the boundary, denoted Γ_u (resp. Γ_θ), where the velocity (resp. the temp.) can be modified :

$$\boldsymbol{u}|_{\Gamma_u} = \gamma_1\, \boldsymbol{u}_{\boldsymbol{\Gamma}}(\mathbf{x}) \qquad \theta|_{\Gamma_\theta} = \gamma_2\, \theta_\Gamma(\mathbf{x})$$

- starting from an initial flow, we want to achieve a temperature $\hat{\theta}$ in $\Omega_c \subset \Omega$ associated to control parameters $\hat{\gamma}$
- the objective functional is

$$\mathcal{J}(\theta, \gamma) = \frac{1}{2} \int_0^T \int_{\Omega_c} |\theta - \hat{\theta}|^2 d\Omega dt + \frac{1}{2} \int_{\Omega_c} |\theta_T - \hat{\theta}_T|^2 d\Omega + \frac{\kappa}{2} |\gamma|^2$$

Problem statement

- incompressible and anisothermal flow subject to gravity $\boldsymbol{g} = -g\boldsymbol{e_y}$
- based on Boussinesq hypothesis, the pb can be written :

$$\begin{cases} \nabla \cdot \boldsymbol{u} = 0 \\ \dfrac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} = -\nabla p + \dfrac{1}{Re} \nabla^2 \boldsymbol{u} + Ri\, \theta\, \mathbf{e_y} \\ \dfrac{\partial \theta}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \theta = \dfrac{1}{RePr} \nabla^2 \theta \end{cases}$$

- we assume that there exists a part of the boundary, denoted Γ_u (resp. Γ_θ), where the velocity (resp. the temp.) can be modified :

$$\boldsymbol{u}|_{\Gamma_u} = \gamma_1\, \boldsymbol{u_\Gamma}(\mathbf{x}) \qquad \theta|_{\Gamma_\theta} = \gamma_2\, \theta_\Gamma(\mathbf{x})$$

- starting from an initial flow, we want to achieve a temperature $\hat{\theta}$ in $\Omega_c \subset \Omega$ associated to control parameters $\hat{\gamma}$
- the objective functional is

$$\mathcal{J}(\theta, \gamma) = \frac{1}{2} \int_0^T \int_{\Omega_c} |\theta - \hat{\theta}|^2 d\Omega dt + \frac{1}{2} \int_{\Omega_c} |\theta_T - \hat{\theta}_T|^2 d\Omega + \frac{\kappa}{2} |\gamma|^2$$

Problem statement

- incompressible and anisothermal flow subject to gravity $\boldsymbol{g} = -g\boldsymbol{e_y}$
- based on Boussinesq hypothesis, the pb can be written :

$$\begin{cases} \nabla \cdot \boldsymbol{u} = 0 \\ \dfrac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} = -\nabla p + \dfrac{1}{Re} \nabla^2 \boldsymbol{u} + Ri\, \theta\, \mathbf{e_y} \\ \dfrac{\partial \theta}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \theta = \dfrac{1}{RePr} \nabla^2 \theta \end{cases}$$

- we assume that there exists a part of the boundary, denoted Γ_u (resp. Γ_θ), where the velocity (resp. the temp.) can be modified :

$$\boldsymbol{u}|_{\Gamma_u} = \gamma_1\, \boldsymbol{u_\Gamma}(\mathbf{x}) \qquad \theta|_{\Gamma_\theta} = \gamma_2\, \theta_\Gamma(\mathbf{x})$$

- starting from an initial flow, we want to achieve a temperature $\hat{\theta}$ in $\Omega_c \subset \Omega$ associated to control parameters $\hat{\gamma}$
- the objective functional is

$$\mathcal{J}(\theta, \gamma) = \frac{1}{2} \int_0^T \int_{\Omega_c} |\theta - \hat{\theta}|^2 d\Omega dt + \frac{1}{2} \int_{\Omega_c} |\theta_T - \hat{\theta}_T|^2 d\Omega + \frac{\kappa}{2} |\gamma|^2$$

Associated unconstrained optimization problem

- this pb is converted into an unconstrained optimization pb by the method of Lagrange multipliers
- we look for a local minimum to the Lagrange functional :

$$\mathcal{L}(\mathbf{u}, p, \theta, \pi, \boldsymbol{\xi}, \beta, \gamma) = \mathcal{J}(\hat{\theta}, \gamma) - \int_0^T \langle \zeta, \mathbf{N}(\mathbf{u}, p, \theta, \gamma) \rangle \, dt \quad \text{with } \zeta = (\pi, \boldsymbol{\xi}, \beta)^T$$

- assuming that all arguments are independent, a local minimum is reached if
  - $\frac{\partial \mathcal{L}}{\partial \zeta} \delta\zeta = 0 \rightarrow$ state equations $\mathbf{N}(\mathbf{u}, p, \theta, \gamma) = 0$
  - $\frac{\partial \mathcal{L}}{\partial \mathbf{u}} \delta\mathbf{u} = \frac{\partial \mathcal{L}}{\partial \theta} \delta\theta = \frac{\partial \mathcal{L}}{\partial p} \delta p = 0$

    $\rightarrow$ adjoint equations $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \gamma) = 0$
  - $\frac{\partial \mathcal{L}}{\partial \gamma} \delta\gamma = 0 \implies$ the optimality condition $\nabla_\gamma \mathcal{J}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \gamma)$

Associated unconstrained optimization problem

- this pb is converted into an unconstrained optimization pb by the method of Lagrange multipliers
- we look for a local minimum to the Lagrange functional :

$$\mathcal{L}(\mathbf{u}, p, \theta, \pi, \boldsymbol{\xi}, \beta, \boldsymbol{\gamma}) = \mathcal{J}(\hat{\theta}, \boldsymbol{\gamma}) - \int_0^T \langle \zeta, \mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}) \rangle \, dt \quad \text{with } \zeta = (\pi, \boldsymbol{\xi}, \beta)^T$$

- assuming that all arguments are independent, a local minimum is reached if
  - $\dfrac{\partial \mathcal{L}}{\partial \zeta} \delta\zeta = 0 \rightarrow$ state equations $\mathbf{N}(\mathbf{u}, p, \theta, \gamma) = \mathbf{0}$
  - $\dfrac{\partial \mathcal{L}}{\partial \mathbf{u}} \delta\mathbf{u} = \dfrac{\partial \mathcal{L}}{\partial \theta} \delta\theta = \dfrac{\partial \mathcal{L}}{\partial p} \delta p = 0$

    $\rightarrow$ adjoint equations $\mathbf{Q}(\xi, \pi, \beta, \mathbf{u}, p, \theta, \gamma) = \mathbf{0}$
  - $\dfrac{\partial \mathcal{L}}{\partial \gamma} \delta\gamma = 0 \;\rightarrow$ the optimality condition $\nabla_\gamma J(\xi, \pi, \beta, \mathbf{u}, p, \theta, \gamma)$

Associated unconstrained optimization problem

- this pb is converted into an unconstrained optimization pb by the method of Lagrange multipliers
- we look for a local minimum to the Lagrange functional :

$$\mathcal{L}(\mathbf{u}, p, \theta, \pi, \boldsymbol{\xi}, \beta, \boldsymbol{\gamma}) = \mathcal{J}(\hat{\theta}, \boldsymbol{\gamma}) - \int_0^T \langle \zeta, \mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}) \rangle \, dt \quad \text{with } \zeta = (\pi, \boldsymbol{\xi}, \beta)^T$$

- assuming that all arguments are independent, a local minimum is reached if
  - $\dfrac{\partial \mathcal{L}}{\partial \zeta} \delta\zeta = 0 \rightarrow$ state equations $\mathbf{N}(\mathbf{u}, p, \theta, \gamma) = \mathbf{0}$
  - $\dfrac{\partial \mathcal{L}}{\partial \mathbf{u}} \delta\mathbf{u} = \dfrac{\partial \mathcal{L}}{\partial \theta} \delta\theta = \dfrac{\partial \mathcal{L}}{\partial p} \delta p = 0$
    $\rightarrow$ adjoint equations $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \gamma) = \mathbf{0}$
  - $\dfrac{\partial \mathcal{L}}{\partial \gamma} \delta\gamma = 0 \;\rightarrow$ the optimality condition $\nabla_\gamma J(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \gamma)$

Associated unconstrained optimization problem

- this pb is converted into an unconstrained optimization pb by the method of Lagrange multipliers
- we look for a local minimum to the Lagrange functional :

$$\mathcal{L}(\mathbf{u}, p, \theta, \pi, \boldsymbol{\xi}, \beta, \boldsymbol{\gamma}) = \mathcal{J}(\hat{\theta}, \boldsymbol{\gamma}) - \int_0^T \langle \zeta, \mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}) \rangle \, dt \quad \text{with } \zeta = (\pi, \boldsymbol{\xi}, \beta)^T$$

- assuming that all arguments are independent, a local minimum is reached if
  - $\dfrac{\partial \mathcal{L}}{\partial \zeta} \delta\zeta = 0 \rightarrow$ state equations $\mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}) = \mathbf{0}$
  - $\dfrac{\partial \mathcal{L}}{\partial \mathbf{u}} \delta\mathbf{u} = \dfrac{\partial \mathcal{L}}{\partial \theta} \delta\theta = \dfrac{\partial \mathcal{L}}{\partial p} \delta p = 0$
    $\rightarrow$ adjoint equations $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}) = \mathbf{0}$
  - $\dfrac{\partial \mathcal{L}}{\partial \boldsymbol{\gamma}} \delta\boldsymbol{\gamma} = 0 \rightarrow$ the optimality condition $\nabla_{\boldsymbol{\gamma}} J(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma})$

Resolution by an iterative descent method

a) initialization of the algorithm : $k = 0$ and $\gamma^{(k)} = \gamma init$

b) solving the state pb $\mathbf{N}(\mathbf{u}, p, \theta, \gamma^{(k)}) = \mathbf{0}$

c) solving the adjoint pb $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \gamma^{(k)}) = \mathbf{0}$

d) assessment of the descent direction $\mathbf{d}^{(k)} = -\nabla_\gamma J(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \gamma^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

f) update the control parameter $\gamma^{(k+1)} = \gamma^{(k)} + \omega^{(k)} d^{(k)}$

g) If $\mathcal{J}(\mathbf{u}, \gamma^{(k+1)}) \geq \eta$ return to step b.

- many resolutions of the Navier-Stokes and their adjoint equations
- CPU time is very large and great storage capacity is required

Resolution by an iterative descent method

a) initialization of the algorithm : $k = 0$ and $\boldsymbol{\gamma}^{(k)} = \boldsymbol{\gamma} init$

b) solving the state pb $\mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

c) solving the adjoint pb $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

d) assessment of the descent direction $\mathbf{d}^{(k)} = -\nabla_{\gamma} J(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

f) update the control parameter $\boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + \omega^{(k)} d^{(k)}$

g) If $\mathcal{J}(\mathbf{u}, \boldsymbol{\gamma}^{(k+1)}) \geq \eta$ return to step b.

- many resolutions of the Navier-Stokes and their adjoint equations
- CPU time is very large and great storage capacity is required

Resolution by an iterative descent method

a) initialization of the algorithm : $k = 0$ and $\boldsymbol{\gamma}^{(k)} = \boldsymbol{\gamma} init$

b) solving the state pb $\mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

c) solving the adjoint pb $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

d) assessment of the descent direction $\mathbf{d}^{(k)} = -\nabla_{\boldsymbol{\gamma}} J(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

f) update the control parameter $\boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + \omega^{(k)} d^{(k)}$

g) If $\mathcal{J}(\mathbf{u}, \gamma^{(k+1)}) \geq \eta$ return to step b.

- many resolutions of the Navier-Stokes and their adjoint equations
- CPU time is very large and great storage capacity is required

Resolution by an iterative descent method

a) initialization of the algorithm : $k = 0$ and $\boldsymbol{\gamma}^{(k)} = \boldsymbol{\gamma} init$

b) solving the state pb $\mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

c) solving the adjoint pb $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

d) assessment of the descent direction $\mathbf{d}^{(k)} = -\nabla_{\boldsymbol{\gamma}} J(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

f) update the control parameter $\boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + \omega^{(k)} d^{(k)}$

g) If $\mathcal{J}(\mathbf{u}, \boldsymbol{\gamma}^{(k+1)}) \geq \eta$ return to step b.

- many resolutions of the Navier-Stokes and their adjoint equations
- CPU time is very large and great storage capacity is required

Resolution by an iterative descent method

a) initialization of the algorithm : $k = 0$ and $\gamma^{(k)} = \gamma init$

b) solving the state pb $\mathbf{N}(\mathbf{u}, p, \theta, \gamma^{(k)}) = \mathbf{0}$

c) solving the adjoint pb $\mathbf{Q}(\xi, \pi, \beta, \mathbf{u}, p, \theta, \gamma^{(k)}) = \mathbf{0}$

d) assessment of the descent direction $\mathbf{d}^{(k)} = -\nabla_\gamma J(\xi, \pi, \beta, \mathbf{u}, p, \theta, \gamma^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

f) update the control parameter $\gamma^{(k+1)} = \gamma^{(k)} + \omega^{(k)} d^{(k)}$

g) If $\mathcal{J}(\mathbf{u}, \gamma^{(k+1)}) \geq \eta$ return to step b.

- many resolutions of the Navier-Stokes and their adjoint equations
- CPU time is very large and great storage capacity is required

Resolution by an iterative descent method

a) initialization of the algorithm : $k = 0$ and $\boldsymbol{\gamma}^{(k)} = \boldsymbol{\gamma} init$

b) solving the state pb $\mathbf{N}(\mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

c) solving the adjoint pb $\mathbf{Q}(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)}) = \mathbf{0}$

d) assessment of the descent direction $\mathbf{d}^{(k)} = -\nabla_{\boldsymbol{\gamma}} J(\boldsymbol{\xi}, \pi, \beta, \mathbf{u}, p, \theta, \boldsymbol{\gamma}^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

f) update the control parameter $\boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + \omega^{(k)} d^{(k)}$

g) If $\mathcal{J}(\mathbf{u}, \boldsymbol{\gamma}^{(k+1)}) \geq \eta$ return to step b.

- many resolutions of the Navier-Stokes and their adjoint equations
- CPU time is very large and great storage capacity is required

A solution ?

- using reduced order methods

Reduced order models

- find a reduced basis Φ such that the solution w of the problem we solve, can be approximated as :

$$w(\mathbf{x}, t) \simeq w_N(\mathbf{x}, t) = \sum_{k=1}^{N} a^k(t)\phi^k(\mathbf{x})$$

  - $N <<$ nb of degrees of freedom computed with FV, FE, FD...

- the time coefficients $a^k(t)$ are the solutions of a system of N differential equations
  - obtained by projecting the equ. onto each $\phi^k(\mathbf{x})$
  - solving this system is almost instantaneous.
- Many reduction techniques have been developed :
  - POD, Balanced Truncation, DMD, SVD, PGD ...

Reduced order models

- find a reduced basis Φ such that the solution w of the problem we solve, can be approximated as :

$$w(\mathbf{x},t) \simeq w_N(\mathbf{x},t) = \sum_{k=1}^{N} a^k(t)\phi^k(\mathbf{x})$$

  - $N <<$ nb of degrees of freedom computed with FV, FE, FD...

- the time coefficients $a^k(t)$ are the solutions of a system of N differential equations
  - obtained by projecting the equ. onto each $\phi^k(\mathbf{x})$
  - solving this system is almost instantaneous.

- Many reduction techniques have been developed :
  - POD, Balanced Truncation, DMD, SVD, PGD ...

Reduced order models

- find a reduced basis Φ such that the solution w of the problem we solve, can be approximated as :

$$w(\mathbf{x}, t) \simeq w_N(\mathbf{x}, t) = \sum_{k=1}^{N} a^k(t)\phi^k(\mathbf{x})$$

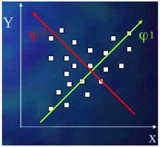
  - $N <<$ nb of degrees of freedom computed with FV, FE, FD...

- the time coefficients $a^k(t)$ are the solutions of a system of N differential equations
  - obtained by projecting the equ. onto each $\phi^k(\mathbf{x})$
  - solving this system is almost instantaneous.

- Many reduction techniques have been developed :
  - POD, Balanced Truncation, DMD, SVD, PGD ...

POD (Proper Orthogonal Decomposition)

- obtain snapshots $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$ representing the studied phenomenon, based on numerical simulations or experiments
- seek the determinist functions $\{\phi^j(\boldsymbol{x})\}_{j=1}^{m}$ that are the best approx. in average of a set of a large number of random data $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$



- is equivalent to solving the maximization problem :

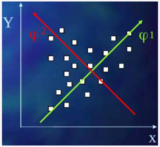
$$\max_{\phi^l \in H} < (w, \phi^l)^2 > \quad \text{avec} \quad (\phi^i, \phi^l) = \delta_{il} \quad \text{pour } 1 \le i \le l \le m$$

- this leads to solving the eigenvalue problem :

$$\Big(R(x, x'), \phi(x)\Big) = \lambda \phi(x) \text{ with } R(x, x') = < w(x, t) w(x', t) >$$

POD (Proper Orthogonal Decomposition)

- obtain snapshots $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$ representing the studied phenomenon, based on numerical simulations or experiments
- seek the determinist functions $\{\phi^j(\boldsymbol{x})\}_{j=1}^{m}$ that are the best approx. in average of a set of a large number of random data $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$



- is equivalent to solving the maximization problem :

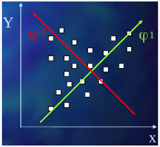
$$\max_{\phi^l \in H} < (w, \phi^l)^2 > \quad \text{avec} \quad (\phi^i, \phi^l) = \delta_{il} \quad \text{pour } 1 \leq i \leq l \leq m$$

- this leads to solving the eigenvalue problem :

$$\Big(R(\boldsymbol{x}, \boldsymbol{x}'), \phi(\boldsymbol{x}) \Big) = \lambda \phi(\boldsymbol{x}) \text{ with } R(\boldsymbol{x}, \boldsymbol{x}') = < w(\boldsymbol{x}, t) w(\boldsymbol{x}', t) >$$

POD (Proper Orthogonal Decomposition)

- obtain snapshots $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$ representing the studied phenomenon, based on numerical simulations or experiments
- seek the determinist functions $\{\phi^j(\boldsymbol{x})\}_{j=1}^{m}$ that are the best approx. in average of a set of a large number of random data $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$



- is equivalent to solving the maximization problem :

$$\max_{\phi^l \in H} < (w, \phi^l)^2 > \quad \text{avec} \quad (\phi^i, \phi^l) = \delta_{il} \quad \text{pour } 1 \leq i \leq l \leq m$$

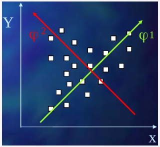
- this leads to solving the eigenvalue problem :

$$\Big(R(\boldsymbol{x}, \boldsymbol{x}'), \phi(\boldsymbol{x})\Big) = \lambda \phi(\boldsymbol{x}) \text{ with } R(\boldsymbol{x}, \boldsymbol{x}') =< w(\boldsymbol{x}, t) w(\boldsymbol{x}', t) >$$

POD (Proper Orthogonal Decomposition)

- obtain snapshots $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$ representing the studied phenomenon, based on numerical simulations or experiments

- seek the determinist functions $\{\phi^j(\boldsymbol{x})\}_{j=1}^{m}$ that are the best approx. in average of a set of a large number of random data $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$



- is equivalent to solving the maximization problem :

$$\max_{\phi^l \in H} < \left(w, \phi^l\right)^2 > \quad \text{avec} \quad \left(\phi^i, \phi^l\right) = \delta_{il} \quad \text{pour } 1 \leq i \leq l \leq m$$

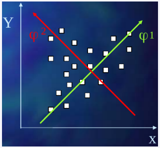
- this leads to solving the eigenvalue problem :

$$\Big(R(\boldsymbol{x}, \boldsymbol{x'}), \phi(\boldsymbol{x})\Big) = \lambda \phi(\boldsymbol{x}) \text{ with } R(\boldsymbol{x}, \boldsymbol{x'}) =< w(\boldsymbol{x}, t) w(\boldsymbol{x'}, t) >$$

POD (Proper Orthogonal Decomposition)

- obtain snapshots $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$ representing the studied phenomenon, based on numerical simulations or experiments

- seek the determinist functions $\{\phi^j(\boldsymbol{x})\}_{j=1}^{m}$ that are the best approx. in average of a set of a large number of random data $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$



- in practice, we solve the following eigenvalues pb (snapshot method)) :

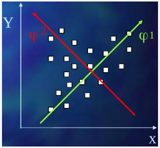
$$[C]\boldsymbol{a} = \lambda \boldsymbol{a} \text{ with } C_{ki} = \frac{1}{M}\left(w(\boldsymbol{x}', t_k), w(\boldsymbol{x}', t_i)\right) \text{ and } \boldsymbol{a} = {}^t\{a^1, \ldots, a^M\}$$

and the spatial modes are given by

$$\phi^i(\boldsymbol{x}) = \sum_{k=1}^{M} a_k^i w(\boldsymbol{x}, t_k)$$

POD (Proper Orthogonal Decomposition)

- obtain snapshots $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$ representing the studied phenomenon, based on numerical simulations or experiments
- seek the determinist functions $\{\phi^j(\boldsymbol{x})\}_{j=1}^{m}$ that are the best approx. in average of a set of a large number of random data $\{w(\boldsymbol{x}, t_i)\}_{i=1}^{M}$



- in practice, we solve the following eigenvalues pb (snapshot method)) :

$$[C]\boldsymbol{a} = \lambda \boldsymbol{a} \text{ with } C_{ki} = \frac{1}{M}\big(w(\boldsymbol{x}', t_k), w(\boldsymbol{x}', t_i)\big) \text{ and } \boldsymbol{a} = {}^t\{a^1, \ldots, a^M\}$$

and the spatial modes are given by

$$\phi^i(\boldsymbol{x}) = \sum_{k=1}^{M} a_k^i w(\boldsymbol{x}, t_k)$$

Properties of the POD basis

- the basis POD Φ is optimal in an energetic sense
  - any realization of the random field w can be approximated with :

$$w(\mathbf{x},t) \simeq w_N(\mathbf{x},t) = \sum_{k=1}^{N} a^k(t)\phi^k(\mathbf{x}) \qquad \text{with } N \text{ small}$$

- the $\phi^n(\mathbf{x})$ respect the boundary cond. and they are divergence free

Properties of the POD basis

- the basis POD Φ is optimal in an energetic sense
  - any realization of the random field w can be approximated with :

$$w(\mathbf{x}, t) \simeq w_N(\mathbf{x}, t) = \sum_{k=1}^{N} a^k(t)\phi^k(\mathbf{x}) \qquad \text{with } N \text{ small}$$

- the $\phi^n(\mathbf{x})$ respect the boundary cond. and they are divergence free

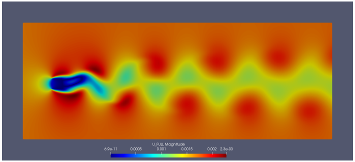
Properties of the POD basis

- the basis POD Φ is optimal in an energetic sense
  - any realization of the random field w can be approximated with :

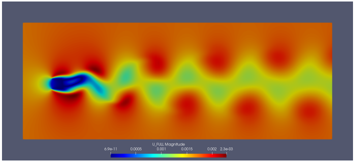
$$w(\boldsymbol{x}, t) \simeq w_N(\mathbf{x}, t) = \sum_{k=1}^{N} a^k(t)\phi^k(\mathbf{x}) \qquad \text{with } N \text{ small}$$

- the $\phi^n(\boldsymbol{x})$ respect the boundary cond. and they are divergence free

- Method illustration (flow in porous media, Re=100, Da=0.0007355)
  - Flow sampling

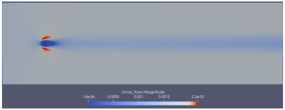


- Method illustration (flow in porous media, Re=100, Da=0.0007355)
  - Flow sampling

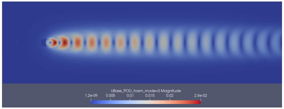


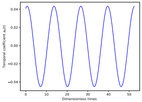
  - POD basis and temporal coeff. (eigenvectors of temporal correlation tensor)

a) Mean field

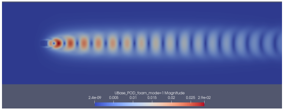


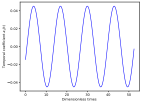
b) Mode 1



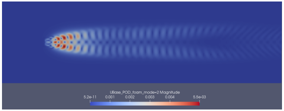


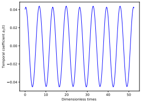
c) Mode 2



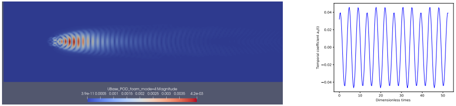


d) Mode 3

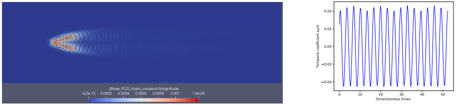




e) Mode 5



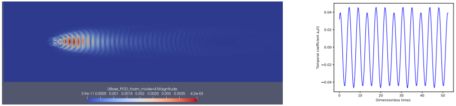
f) Mode 7



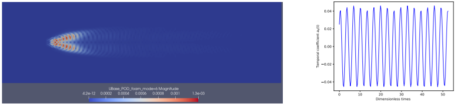
g) Mode 9

h) Mode 11

e) Mode 5



f) Mode 7



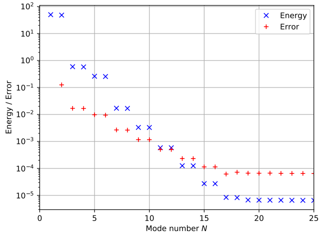
g) Mode 9

h) Mode 11

- Reconstruction error

$$err(N) = \sup_t \frac{\|u(t) - u^N_{POD}(t))\|_{L^2(\Omega)}}{\|u(t)\|_{L^2(\Omega)}}$$

$$u^N_{POD}(\mathbf{x}, t) = \sum_{k=1}^{N} a^k(t)\phi^k(\mathbf{x})$$



ROM associated to the anisothermal Navier-Stokes equations

- velocity and temperature are decomposed as :

$$\boldsymbol{u}(\mathbf{x},\alpha,t) = \bar{\boldsymbol{u}}(\mathbf{x},\alpha) + \boldsymbol{u}'(\mathbf{x},\alpha,t) \quad \text{and} \quad \theta(\mathbf{x},\alpha,t) = \bar{\theta}(\mathbf{x},\alpha) + \theta'(\mathbf{x},\alpha,t)$$

- POD decomposition of the fluctuating parts :

$$\boldsymbol{u}'(\mathbf{x},\alpha,t) \approx \sum_{i=1}^{N_u} a_i(\alpha,t)\boldsymbol{\Phi}_i^{\mathbf{u}}(\mathbf{x}) \quad \text{and} \quad \theta'(\mathbf{x},\alpha,t) \approx \sum_{i=1}^{N_\theta} b_i(\alpha,t)\Phi_i^{\theta}(\mathbf{x},t)$$

  - the sampling required to build this basis consists of snapshots corresponding to one or more values of the parameter α.

ROM associated to the anisothermal Navier-Stokes equations

- velocity and temperature are decomposed as :

$$\boldsymbol{u}(\mathbf{x},\alpha,t) = \bar{\boldsymbol{u}}(\mathbf{x},\alpha) + \boldsymbol{u}'(\mathbf{x},\alpha,t) \quad \text{and} \quad \theta(\mathbf{x},\alpha,t) = \bar{\theta}(\mathbf{x},\alpha) + \theta'(\mathbf{x},\alpha,t)$$

- POD decomposition of the fluctuating parts :

$$\boldsymbol{u}'(\mathbf{x},\alpha,t) \approx \sum_{i=1}^{N_u} a_i(\alpha,t)\boldsymbol{\Phi}_\mathbf{i}^\mathbf{u}(\mathbf{x}) \quad \text{and} \quad \theta'(\mathbf{x},\alpha,t) \approx \sum_{i=1}^{N_\theta} b_i(\alpha,t)\Phi_i^\theta(\mathbf{x},t)$$

  - the sampling required to build this basis consists of snapshots corresponding to one or more values of the parameter α.

ROM associated to the anisothermal Navier-Stokes equations

- velocity and temperature are decomposed as :

$$\boldsymbol{u}(\mathbf{x},\alpha,t) = \bar{\boldsymbol{u}}(\mathbf{x},\alpha) + \boldsymbol{u}'(\mathbf{x},\alpha,t) \quad \text{and} \quad \theta(\mathbf{x},\alpha,t) = \bar{\theta}(\mathbf{x},\alpha) + \theta'(\mathbf{x},\alpha,t)$$

- POD decomposition of the fluctuating parts :

$$\boldsymbol{u}'(\mathbf{x},\alpha,t) \approx \sum_{i=1}^{N_u} a_i(\alpha,t)\boldsymbol{\Phi}_{\mathbf{i}}^{\mathbf{u}}(\mathbf{x}) \quad \text{and} \quad \theta'(\mathbf{x},\alpha,t) \approx \sum_{i=1}^{N_\theta} b_i(\alpha,t)\Phi_i^{\theta}(\mathbf{x},t)$$

  - the sampling required to build this basis consists of snapshots corresponding to one or more values of the parameter α.

ROM associated to the anisothermal Navier-Stokes equations

- introduction of the POD decompositions in the Navier-Stokes equ.
  - conservation of momentum equation

$$\sum_{j=1}^{N_u} \mathbf{\Phi}_\mathbf{j}^\mathbf{u} \frac{da_j}{dt} + \sum_{j=1}^{N_u} a_j (\nabla \bar{\boldsymbol{u}}.\mathbf{\Phi}_\mathbf{j}^\mathbf{u} + \nabla \mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\bar{\boldsymbol{u}} - \frac{1}{Re} \nabla^2 \mathbf{\Phi}_\mathbf{j}^\mathbf{u})$$
$$+ \sum_{j=1}^{N_u} \sum_{k=1}^{N_u} a_j a_k \nabla \mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\mathbf{\Phi}_\mathbf{k}^\mathbf{u} + Ri \sum_{j=1}^{N_\theta} b_j \Phi_j^\theta \, \mathbf{e_y} = \mathbf{f}'(\bar{\boldsymbol{u}}, p, \bar{\theta}) + \mathbf{R_u}$$

  - energy conservation equation

$$\sum_{j=1}^{N_\theta} \Phi_j^\theta \frac{db_j}{dt} + \sum_{j=1}^{N_\theta} b_j (\bar{\boldsymbol{u}}.\nabla)\Phi_j^\theta - \sum_{j=1}^{N_\theta} b_j \frac{1}{RePr} \nabla^2 \Phi_j^\theta$$
$$+ \sum_{j=1}^{N_u} \sum_{k=1}^{N_\theta} a_j b_k (\mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\nabla)\Phi_k^\theta + \sum_{j=1}^{N_u} a_j (\mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\nabla)\bar{\theta} = \mathbf{g}'(\bar{\boldsymbol{u}}, \bar{\theta}) + R_\theta$$

ROM associated to the anisothermal Navier-Stokes equations

- introduction of the POD decompositions in the Navier-Stokes equ.
  - conservation of momentum equation

$$\sum_{j=1}^{N_u} \mathbf{\Phi}_\mathbf{j}^\mathbf{u} \frac{da_j}{dt} + \sum_{j=1}^{N_u} a_j (\nabla \bar{\boldsymbol{u}}.\mathbf{\Phi}_\mathbf{j}^\mathbf{u} + \nabla \mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\bar{\boldsymbol{u}} - \frac{1}{Re} \nabla^2 \mathbf{\Phi}_\mathbf{j}^\mathbf{u})$$
$$+ \sum_{j=1}^{N_u} \sum_{k=1}^{N_u} a_j a_k \nabla \mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\mathbf{\Phi}_\mathbf{k}^\mathbf{u} + Ri \sum_{j=1}^{N_\theta} b_j \Phi_j^\theta \, \mathbf{e_y} = \mathbf{f}'(\bar{\boldsymbol{u}}, p, \bar{\theta}) + \mathbf{R_u}$$

  - energy conservation equation

$$\sum_{j=1}^{N_\theta} \Phi_j^\theta \frac{db_j}{dt} + \sum_{j=1}^{N_\theta} b_j (\bar{\boldsymbol{u}}.\nabla) \Phi_j^\theta - \sum_{j=1}^{N_\theta} b_j \frac{1}{RePr} \nabla^2 \Phi_j^\theta$$
$$+ \sum_{j=1}^{N_u} \sum_{k=1}^{N_\theta} a_j b_k (\mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\nabla) \Phi_k^\theta + \sum_{j=1}^{N_u} a_j (\mathbf{\Phi}_\mathbf{j}^\mathbf{u}.\nabla) \bar{\theta} = \mathbf{g}'(\bar{\boldsymbol{u}}, \bar{\theta}) + R_\theta$$

ROM associated to the anisothermal Navier-Stokes equations

- introduction of the POD decompositions in the Navier-Stokes equ.
  - conservation of momentum equation

$$\sum_{j=1}^{N_u} \mathbf{\Phi}_{\mathbf{j}}^{\mathbf{u}} \frac{da_j}{dt} + \sum_{j=1}^{N_u} a_j (\nabla \bar{\boldsymbol{u}}.\mathbf{\Phi}_{\mathbf{j}}^{\mathbf{u}} + \nabla \mathbf{\Phi}_{\mathbf{j}}^{\mathbf{u}}.\bar{\boldsymbol{u}} - \frac{1}{Re} \nabla^2 \mathbf{\Phi}_{\mathbf{j}}^{\mathbf{u}}) + \sum_{j=1}^{N_u} \sum_{k=1}^{N_u} a_j a_k \nabla \mathbf{\Phi}_{\mathbf{j}}^{\mathbf{u}}.\mathbf{\Phi}_{\mathbf{k}}^{\mathbf{u}} + Ri \sum_{j=1}^{N_\theta} b_j \Phi_j^\theta \, \mathbf{e_y} = \mathbf{f}'(\bar{\boldsymbol{u}}, p, \bar{\theta}) + \mathbf{R_u}$$

  - energy conservation equation

$$\sum_{j=1}^{N_\theta} \Phi_j^\theta \frac{db_j}{dt} + \sum_{j=1}^{N_\theta} b_j (\bar{\boldsymbol{u}}.\nabla) \Phi_j^\theta - \sum_{j=1}^{N_\theta} b_j \frac{1}{RePr} \nabla^2 \Phi_j^\theta + \sum_{j=1}^{N_u} \sum_{k=1}^{N_\theta} a_j b_k (\mathbf{\Phi}_{\mathbf{j}}^{\mathbf{u}}.\nabla) \Phi_k^\theta + \sum_{j=1}^{N_u} a_j (\mathbf{\Phi}_{\mathbf{j}}^{\mathbf{u}}.\nabla) \bar{\theta} = \mathbf{g}'(\bar{\boldsymbol{u}}, \bar{\theta}) + R_\theta$$

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\langle \mathbf{\Phi_i^u}, \mathbf{R_u} \rangle = 0$ and $\langle \Phi_i^\theta, R_\theta \rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_u} C_{ijk}\, a_j\, a_k + \sum_{j=1}^{N_u} \left(D_{ij}(\alpha) + A_{ij}\right)\, a_j + \sum_{j=1}^{N_\theta} B_{ij}\, b_j + E_{i1}(\alpha) + E_{i2} \quad \text{for } i = 1, \ldots, N_u \\ \dfrac{db_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_\theta} C^\theta_{ijk}\, a_j\, b_k + \sum_{j=1}^{N_u} \left(D^\theta_{ij}(\alpha) + A^\theta_{ij}\right)\, a_j + \sum_{j=1}^{N_\theta} B^\theta_{ij}\, a_j + E^\theta_{i1}(\alpha) \quad \text{for } i = 1, \ldots, N_\theta \end{cases}$$

avec

$$C_{ijk} = -(\mathbf{\Phi_n^u}, \nabla\mathbf{\Phi_m^u}.\mathbf{\Phi_k^u}) \qquad D_{ij}(\alpha) = (\mathbf{\Phi_n^u}, -\nabla\bar{\boldsymbol{u}}.\mathbf{\Phi_m^u} - \nabla\mathbf{\Phi_m^u}.\bar{\boldsymbol{u}}) \qquad A_{ij} = (\mathbf{\Phi_n^u}, \frac{1}{Re}\nabla^2\mathbf{\Phi_m^u})$$

$$B_{ij} = (\mathbf{\Phi_n^u}, \Phi_i^\theta\, \mathbf{e_y}) \qquad E_{i1}(\alpha) = (\mathbf{\Phi_n^u}, -\nabla\bar{p} + \frac{1}{Re}\nabla^2\bar{\boldsymbol{u}} - \nabla\bar{\boldsymbol{u}}.\bar{\boldsymbol{u}}) \qquad E_{i2} = -\int_\Gamma p'\,\mathbf{\Phi_n^u}.\boldsymbol{n}\,d\Gamma$$

$$C^\theta_{ijk} = -(\Phi^\theta_n, (\mathbf{\Phi_i^u}.\nabla)\Phi^\theta_k) \qquad D^\theta_{ij}(\alpha) = (\Phi^\theta_n, (\bar{\boldsymbol{u}}.\nabla)\Phi^\theta_m) \qquad A^\theta_{ij} = (\Phi^\theta_n, \frac{1}{RePr}\nabla^2\Phi^\theta_m)$$

$$B^\theta_{ij} = (\Phi^\theta_n, (\mathbf{\Phi_m^u}.\nabla)\bar{\theta}) \qquad E^\theta_{i1}(\alpha) = (\Phi^\theta_n, \frac{1}{RePr}\nabla^2\bar{\theta} - (\bar{\boldsymbol{u}}.\nabla)\bar{\theta})$$

$\Rightarrow$ coefficients dependent on α (via $\bar{\theta}$) are determined using a Lagrange interpolation of $\bar{\theta}$

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\langle \mathbf{\Phi}_\mathbf{i}^\mathbf{u}, \mathbf{R_u} \rangle = 0$ and $\langle \Phi_i^\theta, R_\theta \rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_u} C_{ijk}\, a_j\, a_k \;+\; \sum_{j=1}^{N_u}\left(D_{ij}(\alpha) + A_{ij}\right)\, a_j \;+\; \sum_{j=1}^{N_\theta} B_{ij}\, b_j \;+\; E_{i1}(\alpha) + E_{i2} \quad \text{for } i = 1, \ldots, N_u \\ \\ \end{cases}$$

avec

$$C_{ijk} = -(\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, \nabla\mathbf{\Phi}_\mathbf{m}^\mathbf{u}.\mathbf{\Phi}_\mathbf{k}^\mathbf{u}) \qquad D_{ij}(\alpha) = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, -\nabla\bar{\boldsymbol{u}}.\mathbf{\Phi}_\mathbf{m}^\mathbf{u} - \nabla\mathbf{\Phi}_\mathbf{m}^\mathbf{u}.\bar{\boldsymbol{u}}) \qquad A_{ij} = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, \frac{1}{Re}\nabla^2\mathbf{\Phi}_\mathbf{m}^\mathbf{u})$$

$$B_{ij} = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, \Phi_i^\theta\, \mathbf{e_y}) \qquad E_{i1}(\alpha) = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, -\nabla\bar{p} + \frac{1}{Re}\nabla^2\bar{\boldsymbol{u}} - \nabla\bar{\boldsymbol{u}}.\bar{\boldsymbol{u}}) \qquad E_{i2} = -\int_\Gamma p'\,\mathbf{\Phi}_\mathbf{n}^\mathbf{u}.\boldsymbol{n}\,d\Gamma$$

- coefficients dependent on α (via $\bar{\boldsymbol{u}}$) are determined using a Lagrange interpolation of $\bar{\boldsymbol{u}}$

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\left\langle \mathbf{\Phi}_{\mathbf{i}}^{\mathbf{u}}, \mathbf{R}_{\mathbf{u}} \right\rangle = 0$ and $\left\langle \Phi_i^{\theta}, R_{\theta} \right\rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_u} C_{ijk}\; a_j\; a_k \;+\; \sum_{j=1}^{N_u} \left(D_{ij}(\alpha) + A_{ij}\right)\; a_j \;+\; \sum_{j=1}^{N_\theta} B_{ij}\; b_j \;+\; E_{i1}(\alpha) + E_{i2} \quad \text{for } i = 1, \ldots, N_u \\ \dfrac{db_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_\theta} C_{ijk}^{\theta}\; a_j\; b_k \;+\; \sum_{j=1}^{N_u} \left(D_{ij}^{\theta}(\alpha) + A_{ij}^{\theta}\right)\; a_j \;+\; \sum_{j=1}^{N_\theta} B_{ij}^{\theta}\; a_j \;+\; E_{i1}^{\theta}(\alpha) \quad \text{for } i = 1, \ldots, N_\theta \end{cases}$$

avec

$$C_{ijk} = -(\mathbf{\Phi}_{\mathbf{n}}^{\mathbf{u}}, \nabla\mathbf{\Phi}_{\mathbf{m}}^{\mathbf{u}}.\mathbf{\Phi}_{\mathbf{k}}^{\mathbf{u}}) \qquad D_{ij}(\alpha) = (\mathbf{\Phi}_{\mathbf{n}}^{\mathbf{u}}, -\nabla\bar{\boldsymbol{u}}.\mathbf{\Phi}_{\mathbf{m}}^{\mathbf{u}} - \nabla\mathbf{\Phi}_{\mathbf{m}}^{\mathbf{u}}.\bar{\boldsymbol{u}}) \qquad A_{ij} = (\mathbf{\Phi}_{\mathbf{n}}^{\mathbf{u}}, \frac{1}{Re}\nabla^2\mathbf{\Phi}_{\mathbf{m}}^{\mathbf{u}})$$

$$B_{ij} = (\mathbf{\Phi}_{\mathbf{n}}^{\mathbf{u}}, \Phi_i^{\theta}\; \mathbf{e}_{\mathbf{y}}) \qquad E_{i1}(\alpha) = (\mathbf{\Phi}_{\mathbf{n}}^{\mathbf{u}}, -\nabla\bar{p} + \frac{1}{Re}\nabla^2\bar{\boldsymbol{u}} - \nabla\bar{\boldsymbol{u}}.\bar{\boldsymbol{u}}) \qquad E_{i2} = -\int_{\Gamma} p'\mathbf{\Phi}_{\mathbf{n}}^{\mathbf{u}}.\boldsymbol{n}d\Gamma$$

$$C_{ijk}^{\theta} = -(\Phi_n^{\theta}, (\mathbf{\Phi}_{\mathbf{i}}^{\mathbf{u}}.\nabla)\Phi_k^{\theta}) \qquad D_{ij}^{\theta}(\alpha) = (\Phi_n^{\theta}, (\bar{\boldsymbol{u}}.\nabla)\Phi_m^{\theta}) \qquad A_{ij}^{\theta} = (\Phi_n^{\theta}, \frac{1}{RePr}\nabla^2\Phi_m^{\theta})$$

$$B_{ij}^{\theta} = (\Phi_n^{\theta}, (\mathbf{\Phi}_{\mathbf{m}}^{\mathbf{u}}.\nabla)\bar{\theta}) \qquad E_{i1}^{\theta}(\alpha) = (\Phi_n^{\theta}, \frac{1}{RePr}\nabla^2\bar{\theta} - (\bar{\boldsymbol{u}}.\nabla)\bar{\theta})$$

- coefficients dependent on α (via $\bar{\boldsymbol{u}}$) are determined using a Lagrange interpolation of $\bar{\boldsymbol{u}}$

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\langle \mathbf{\Phi}_\mathbf{i}^\mathbf{u}, \mathbf{R}_\mathbf{u} \rangle = 0$ and $\langle \Phi_i^\theta, R_\theta \rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_u} C_{ijk}\, a_j\, a_k + \sum_{j=1}^{N_u} \left(D_{ij}(\alpha) + A_{ij}\right) a_j + \sum_{j=1}^{N_\theta} B_{ij}\, b_j + E_{i1}(\alpha) + E_{i2} \quad \text{for } i = 1, \ldots, N_u \\ \dfrac{db_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_\theta} C^\theta_{ijk}\, a_j\, b_k + \sum_{j=1}^{N_u} \left(D^\theta_{ij}(\alpha) + A^\theta_{ij}\right) a_j + \sum_{j=1}^{N_\theta} B^\theta_{ij}\, a_j + E^\theta_{i1}(\alpha) \quad \text{for } i = 1, \ldots, N_\theta \end{cases}$$

avec

$$C_{ijk} = -(\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, \nabla\mathbf{\Phi}_\mathbf{m}^\mathbf{u}.\mathbf{\Phi}_\mathbf{k}^\mathbf{u}) \qquad D_{ij}(\alpha) = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, -\nabla\bar{\boldsymbol{u}}.\mathbf{\Phi}_\mathbf{m}^\mathbf{u} - \nabla\mathbf{\Phi}_\mathbf{m}^\mathbf{u}.\bar{\boldsymbol{u}}) \qquad A_{ij} = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, \frac{1}{Re}\nabla^2\mathbf{\Phi}_\mathbf{m}^\mathbf{u})$$

$$B_{ij} = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, \Phi_i^\theta\, \mathbf{e_y}) \qquad E_{i1}(\alpha) = (\mathbf{\Phi}_\mathbf{n}^\mathbf{u}, -\nabla\bar{p} + \frac{1}{Re}\nabla^2\bar{\boldsymbol{u}} - \nabla\bar{\boldsymbol{u}}.\bar{\boldsymbol{u}}) \qquad E_{i2} = -\int_\Gamma p'\,\mathbf{\Phi}_\mathbf{n}^\mathbf{u}.\boldsymbol{n}\,d\Gamma$$

$$C^\theta_{ijk} = -(\Phi_n^\theta, (\mathbf{\Phi}_\mathbf{i}^\mathbf{u}.\nabla)\Phi_k^\theta) \qquad D^\theta_{ij}(\alpha) = (\Phi_n^\theta, (\bar{\boldsymbol{u}}.\nabla)\Phi_m^\theta) \qquad A^\theta_{ij} = (\Phi_n^\theta, \frac{1}{RePr}\nabla^2\Phi_m^\theta)$$

$$B^\theta_{ij} = (\Phi_n^\theta, (\mathbf{\Phi}_\mathbf{m}^\mathbf{u}.\nabla)\bar{\theta}) \qquad E^\theta_{i1}(\alpha) = (\Phi_n^\theta, \frac{1}{RePr}\nabla^2\bar{\theta} - (\bar{\boldsymbol{u}}.\nabla)\bar{\theta})$$

- coefficients dependent on α (via $\bar{\boldsymbol{u}}$) are determined using a Lagrange interpolation of $\bar{\boldsymbol{u}}$

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\langle \mathbf{\Phi_i^u}, \mathbf{R_u} \rangle = 0$ and $\langle \Phi_i^\theta, R_\theta \rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_u} C_{ijk}\, a_j\, a_k + \sum_{j=1}^{N_u} \left(D_{ij}(\alpha) + A_{ij}\right) a_j + \sum_{j=1}^{N_\theta} B_{ij}\, b_j + E_{i1}(\alpha) + E_{i2} \quad \text{for } i = 1, \ldots, N_u \\ \dfrac{db_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_\theta} C^\theta_{ijk}\, a_j\, b_k + \sum_{j=1}^{N_u} \left(D^\theta_{ij}(\alpha) + A^\theta_{ij}\right) a_j + \sum_{j=1}^{N_\theta} B^\theta_{ij}\, a_j + E^\theta_{i1}(\alpha) \quad \text{for } i = 1, \ldots, N_\theta \end{cases}$$

- system of ODE of low order $N_u + N_\theta \Rightarrow$ fast resolution
- since the divergence of the POD modes is null, if the POD modes are null on the boundaries, the pressure disappears of the ROM
- if it is not the case POD is also applied to the pressure and a ROM that gives the temporal evol. of p and u is construct

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\langle \mathbf{\Phi_i^u}, \mathbf{R_u} \rangle = 0$ and $\langle \Phi_i^\theta, R_\theta \rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_u} C_{ijk}\, a_j\, a_k + \sum_{j=1}^{N_u} \left(D_{ij}(\alpha) + A_{ij}\right) a_j + \sum_{j=1}^{N_\theta} B_{ij}\, b_j + E_{i1}(\alpha) + E_{i2} \quad \text{for } i = 1, \dots, N_u \\ \dfrac{db_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_\theta} C_{ijk}^\theta\, a_j\, b_k + \sum_{j=1}^{N_u} \left(D_{ij}^\theta(\alpha) + A_{ij}^\theta\right) a_j + \sum_{j=1}^{N_\theta} B_{ij}^\theta\, a_j + E_{i1}^\theta(\alpha) \quad \text{for } i = 1, \dots, N_\theta \end{cases}$$

  - system of ODE of low order $N_u + N_\theta \Rightarrow$ fast resolution
  - since the divergence of the POD modes is null, if the POD modes are null on the boundaries, the pressure disappears of the ROM
  - if it is not the case POD is also applied to the pressure and a ROM that gives the temporal evol. of p and $\mathbf{u}$ is construct

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\langle \mathbf{\Phi_i^u}, \mathbf{R_u} \rangle = 0$ and $\langle \Phi_i^\theta, R_\theta \rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_u} C_{ijk}\, a_j\, a_k + \sum_{j=1}^{N_u} \left(D_{ij}(\alpha) + A_{ij}\right) a_j + \sum_{j=1}^{N_\theta} B_{ij}\, b_j + E_{i1}(\alpha) + E_{i2} & \text{for } i = 1, \ldots, N_u \\ \dfrac{db_i}{dt} = \displaystyle\sum_{j=1}^{N_u}\sum_{k=1}^{N_\theta} C^\theta_{ijk}\, a_j\, b_k + \sum_{j=1}^{N_u} \left(D^\theta_{ij}(\alpha) + A^\theta_{ij}\right) a_j + \sum_{j=1}^{N_\theta} B^\theta_{ij}\, a_j + E^\theta_{i1}(\alpha) & \text{for } i = 1, \ldots, N_\theta \end{cases}$$

  - system of ODE of low order $N_u + N_\theta \Rightarrow$ fast resolution
  - since the divergence of the POD modes is null, if the POD modes are null on the boundaries, the pressure disappears of the ROM
  - if it is not the case POD is also applied to the pressure and a ROM that gives the temporal evol. of p and $\boldsymbol{u}$ is construct

ROM associated to the anisothermal Navier-Stokes equations

- Galerkin projection : $\langle \mathbf{\Phi_i^u}, \mathbf{R_u} \rangle = 0$ and $\langle \Phi_i^\theta, R_\theta \rangle = 0$
- the reduced order model is written as

$$\begin{cases} \dfrac{da_i}{dt} = \displaystyle\sum_{j=1}^{N_u} \sum_{k=1}^{N_u} C_{ijk}\, a_j\, a_k + \sum_{j=1}^{N_u} \left(D_{ij}(\alpha) + A_{ij}\right)\, a_j + \sum_{j=1}^{N_\theta} B_{ij}\, b_j + E_{i1}(\alpha) + E_{i2} \quad \text{for } i = 1, \ldots, N_u \\ \dfrac{db_i}{dt} = \displaystyle\sum_{j=1}^{N_u} \sum_{k=1}^{N_\theta} C^\theta_{ijk}\, a_j\, b_k + \sum_{j=1}^{N_u} \left(D^\theta_{ij}(\alpha) + A^\theta_{ij}\right)\, a_j + \sum_{j=1}^{N_\theta} B^\theta_{ij}\, a_j + E^\theta_{i1}(\alpha) \quad \text{for } i = 1, \ldots, N_\theta \end{cases}$$

- system of ODE of low order $N_u + N_\theta \Rightarrow$ fast resolution
- since the divergence of the POD modes is null, if the POD modes are null on the boundaries, the pressure disappears of the ROM
- if it is not the case POD is also applied to the pressure and a ROM that gives the temporal evol. of p and $\boldsymbol{u}$ is construct

Tallet et al, A minimum residual projection to build coupled velocity-pressure POD-ROM for incomp. NS equations, *Comm. in Nonlinear Science and Num. Simulation*, vol 22, 2015.

Formulation of the reduced optimal control problem

- POD is applied to the target temperature $\hat{\theta}$

$$\hat{\theta}(\mathbf{x},t,\hat{\gamma}) \approx \overline{\theta}(\mathbf{x}) + \sum_{j=1}^{N_{\hat{\theta}}} \hat{b}_j(t)\mathbf{\Phi}_j^{\hat{\theta}}(\mathbf{x})$$

- the reduced optimal control is written :

Search the control parameter γ and the state variables $\mathbf{a} = (a_1,\ldots,a_{N_u})$ and $\mathbf{b} = (b_1,\ldots,b_{N_\theta})$ such that the functional

$$\mathcal{J}_{red}(\mathbf{b},\gamma) = \frac{1}{2}\int_0^T \left(\sum_{k=1}^{N_\theta} b_k^2 + \sum_{l=1}^{N_{\hat{u}}} \hat{b}_l^2 - \sum_{k=1}^{N_u}\sum_{l=1}^{N_{\hat{u}}} C_{kl} b_k \hat{b}_l\right) dt$$
$$+\sum_{k=1}^{N_u} b_k^2(T) + \sum_{l=1}^{N_{\hat{u}}} \hat{b}_l^2(T) + \sum_{k=1}^{N_u}\sum_{l=1}^{N_{\hat{u}}} C_{kl} b_k(T)\hat{b}_l(T) + \frac{\kappa}{2}|\gamma|^2.$$

with $C_{kl} = \left\langle \mathbf{\Phi}_k^{\theta}, \mathbf{\Phi}_l^{\hat{\theta}} \right\rangle$

is minimized under the constraints of the previous ROMs denoted $\mathcal{M}_j(\mathbf{a},\mathbf{b},\gamma)$ and $\mathcal{N}_j(\mathbf{a},\mathbf{b},\gamma)$

- this pb is converted into an unconstrained optimization pb by the method of Lagrange multipliers

Formulation of the reduced optimal control problem

- POD is applied to the target temperature $\hat{\theta}$

$$\hat{\theta}(\mathbf{x},t,\hat{\gamma}) \approx \overline{\theta}(\mathbf{x}) + \sum_{j=1}^{N_{\hat{\theta}}} \hat{b}_j(t)\boldsymbol{\Phi}_j^{\hat{\theta}}(\mathbf{x})$$

- the reduced optimal control is written :

Search the control parameter γ and the state variables

$\mathbf{a} = (a_1, \ldots, a_{N_u})$ and $\mathbf{b} = (b_1, \ldots, b_{N_\theta})$ such that the functional

$$\mathcal{J}_{red}(\mathbf{b},\boldsymbol{\gamma}) = \frac{1}{2}\int_0^T \left(\sum_{k=1}^{N_\theta} b_k^2 + \sum_{l=1}^{N_{\hat{u}}} \hat{b}_l^2 - \sum_{k=1}^{N_u}\sum_{l=1}^{N_{\hat{u}}} C_{kl}\, b_k\, \hat{b}_l \right) dt$$
$$+ \sum_{k=1}^{N_u} b_k^2(T) + \sum_{l=1}^{N_{\hat{u}}} \hat{b}_l^2(T) + \sum_{k=1}^{N_u}\sum_{l=1}^{N_{\hat{u}}} C_{kl}\, b_k(T)\hat{b}_l(T) + \frac{\kappa}{2}|\boldsymbol{\gamma}|^2.$$

$$\text{with } C_{kl} = \left\langle \boldsymbol{\Phi}_k^{\theta}, \boldsymbol{\Phi}_l^{\hat{\theta}} \right\rangle$$

is minimized under the constraints of the previous ROMs denoted

$\mathcal{M}_j(\mathbf{a},\mathbf{b},\boldsymbol{\gamma})$ and $\mathcal{N}_j(\mathbf{a},\mathbf{b},\boldsymbol{\gamma})$

- this pb is converted into an unconstrained optimization pb by the method of Lagrange multipliers

Formulation of the reduced optimal control problem

- POD is applied to the target temperature $\hat{\theta}$

$$\hat{\theta}(\mathbf{x}, t, \hat{\gamma}) \approx \overline{\theta}(\mathbf{x}) + \sum_{j=1}^{N_{\hat{\theta}}} \hat{b}_j(t)\boldsymbol{\Phi}_j^{\hat{\theta}}(\mathbf{x})$$

- the reduced optimal control is written :

Search the control parameter γ and the state variables $\mathbf{a} = (a_1, \ldots, a_{N_u})$ and $\mathbf{b} = (b_1, \ldots, b_{N_\theta})$ such that the functional

$$\mathcal{J}_{red}(\mathbf{b}, \boldsymbol{\gamma}) = \frac{1}{2}\int_0^T \left(\sum_{k=1}^{N_\theta} b_k^2 + \sum_{l=1}^{N_{\hat{u}}} \hat{b}_l^2 - \sum_{k=1}^{N_u}\sum_{l=1}^{N_{\hat{u}}} C_{kl}\, b_k\, \hat{b}_l \right) dt$$
$$+ \sum_{k=1}^{N_u} b_k^2(T) + \sum_{l=1}^{N_{\hat{u}}} \hat{b}_l^2(T) + \sum_{k=1}^{N_u}\sum_{l=1}^{N_{\hat{u}}} C_{kl}\, b_k(T)\hat{b}_l(T) + \frac{\kappa}{2}|\boldsymbol{\gamma}|^2.$$

with $C_{kl} = \left\langle \boldsymbol{\Phi}_k^{\theta}, \boldsymbol{\Phi}_l^{\hat{\theta}} \right\rangle$

is minimized under the constraints of the previous ROMs denoted $\mathcal{M}_j(\mathbf{a}, \mathbf{b}, \boldsymbol{\gamma})$ and $\mathcal{N}_j(\mathbf{a}, \mathbf{b}, \boldsymbol{\gamma})$

- this pb is converted into an unconstrained optimization pb by the method of Lagrange multipliers

Resolution by an iterative descent method

a) initialization of the algorithm :$k = 0$ et $\gamma^{(k)} = \gamma_{init}$

b) solving the state ROM

$$\mathcal{M}(\boldsymbol{a}, \boldsymbol{b}, \gamma^{(k)}) = \mathbf{0} \text{ and } \mathcal{N}(\boldsymbol{a}, \boldsymbol{b}, \gamma^{(k)}) = \mathbf{0} \rightarrow \mathbf{a}^{(k)} \text{ and } \mathbf{b}^{(k)}$$

c) solving the adjoint ROM

$$\mathcal{P}(\boldsymbol{a}, \boldsymbol{b}, \zeta, \xi, \alpha^{(k)}) = \mathbf{0},\ \mathcal{Q}(\boldsymbol{a}, \boldsymbol{b}, \zeta, \xi, \alpha^{(k)}) = \mathbf{0} \rightarrow \zeta^{(k)} \text{ and } \xi^{(k)}$$

d) assessment of the descent dir. $\rightarrow \mathbf{d}^{(k)} = -\nabla_\gamma J_{red}(\mathbf{a}, \mathbf{b}, , \zeta, \xi, \gamma^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

f) update the control parameter $\rightarrow \gamma^{(k+1)} = \gamma^{(k)} + \omega^{(k)} d^{(k)}$

g) convergence criterion : if $\|J_{red}(\mathbf{a}, \gamma^{(k+1)})\| > \varepsilon$, return to step b)

- fast to solve and small storage capacity is required

Resolution by an iterative descent method

a) initialization of the algorithm : $k = 0$ et $\gamma^{(k)} = \gamma_{init}$

b) solving the state ROM

$$\mathcal{M}(\boldsymbol{a}, \boldsymbol{b}, \gamma^{(k)}) = \mathbf{0} \text{ and } \mathcal{N}(\boldsymbol{a}, \boldsymbol{b}, \gamma^{(k)}) = \mathbf{0} \rightarrow \mathbf{a}^{(k)} \text{ and } \mathbf{b}^{(k)}$$

c) solving the adjoint ROM

$$\mathcal{P}(\boldsymbol{a}, \boldsymbol{b}, \zeta, \boldsymbol{\xi}, \alpha^{(k)}) = \mathbf{0},\ \mathcal{Q}(\boldsymbol{a}, \boldsymbol{b}, \zeta, \boldsymbol{\xi}, \alpha^{(k)}) = \mathbf{0} \rightarrow \zeta^{(k)} \text{ and } \boldsymbol{\xi}^{(k)}$$

d) assessment of the descent dir. $\rightarrow \mathbf{d}^{(k)} = -\nabla_\gamma J_{red}(\mathbf{a}, \mathbf{b}, , \boldsymbol{\zeta}, \boldsymbol{\xi}, \boldsymbol{\gamma}^{(k)})$

e) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

(linear search algorithm of Armijo)

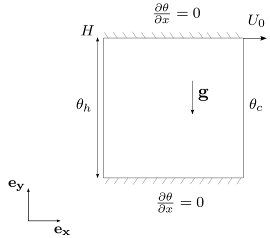
f) update the control parameter $\rightarrow \boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + \omega^{(k)} d^{(k)}$

g) convergence criterion : if $\|J_{red}(\mathbf{a}, \boldsymbol{\gamma}^{(k+1)})\| > \varepsilon$, return to step b)

- fast to solve and small storage capacity is required

Application : 2D lid driven heated square cavity

- square cavity of side H
- $10^6 \leq Gr \leq 5 \times 10^6$
- $158 \leq Re \leq 474$
- uniform grid with 100^2 cells
- transient regime
- EDF finite-volume code : Saturne



Tallet, Allery, Leblond Optimal flow control using a POD based Reduced-Order Model, *Numerical Heat Transfer, Part B*, vol 70, 2016.

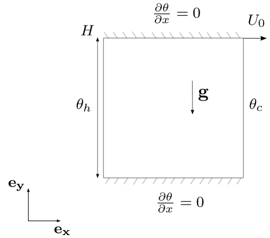
Objective

- controlling the temperature inside the cavity by varying the control parameters γ_1 and γ_2 defined by

$$\mathbf{u}_{|\Gamma_{top}} = \gamma_1 \, U_0 \, \mathbf{e_x} \qquad \text{and} \qquad \theta_{|\Gamma_{left}} = \gamma_2 \, \theta_c$$

Application : 2D lid driven heated square cavity

- square cavity of side H
- $10^6 \leq Gr \leq 5 \times 10^6$
- $158 \leq Re \leq 474$
- uniform grid with 100^2 cells
- transient regime
- EDF finite-volume code : Saturne



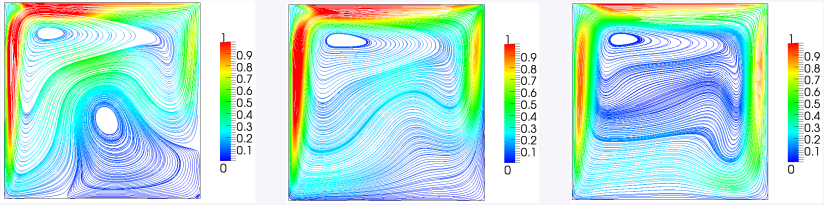
Tallet, Allery, Leblond Optimal flow control using a POD based Reduced-Order Model, *Numerical Heat Transfer, Part B*, vol 70, 2016.

Objective

- controlling the temperature inside the cavity by varying the control parameters γ_1 and γ_2 defined by

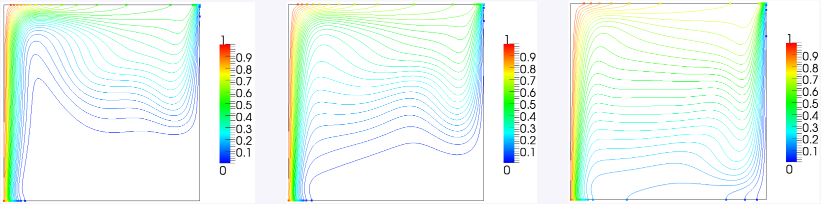
$$\mathbf{u}_{|\Gamma_{top}} = \gamma_1 \, U_0 \, \mathbf{e_x} \qquad \text{and} \qquad \theta_{|\Gamma_{left}} = \gamma_2 \, \theta_c$$

Streamlines for $Re = 316$ and $Gr = 10^6$



a) $t = 3$ b) $t = 5$ c) $t = 15$

Isovalues of temperature for $Re = 316$ and $Gr = 10^6$



a) $t = 3$ b) $t = 5$ c) $t = 15$

Construction of the POD basis

- the snapshots are obtained from 6 simulations at various Reynolds numbers ($158 \leq Re \leq 474$) and various Grashof numbers $10^6 \leq Re \leq 5.10^6$:

Reynolds number Re	Grashof Gr
158	1.10^6 5.10^6
316	1.10^6 5.10^6
474	1.10^6 5.10^6

- for each couple Re-Gr, 150 snapshots evenly distributed on the transient regime are considered

Construction of the POD basis

- the snapshots are obtained from 6 simulations at various Reynolds numbers ($158 \leq Re \leq 474$) and various Grashof numbers $10^6 \leq Re \leq 5.10^6$:

Reynolds number Re	Grashof Gr
158	1.10^6 5.10^6
316	1.10^6 5.10^6
474	1.10^6 5.10^6

- for each couple Re-Gr, 150 snapshots evenly distributed on the transient regime are considered

Control

- we want to achieve a target temp. $\hat{\theta}$ corresponding to a couple Re_{targ}-Gr_{targ} by starting from a temp. θ_{init} corresponding to Re_{init}-Gr_{init}
- four target pairs of Re-Gr that do not belong to the sampling are considered

1) $Re = 221\,;\ Gr = 2.10^6$
2) $Re = 221\,;\ Gr = 4.10^6$
3) $Re = 379\,;\ Gr = 2.10^6$
4) $Re = 379\,;\ Gr = 4.10^6$

Control

- we want to achieve a target temp. $\hat{\theta}$ corresponding to a couple Re_{targ}-Gr_{targ} by starting from a temp. θ_{init} corresponding to Re_{init}-Gr_{init}
- four target pairs of Re-Gr that do not belong to the sampling are considered

1) $Re = 221\,;\ Gr = 2.10^6$
2) $Re = 221\,;\ Gr = 4.10^6$
3) $Re = 379\,;\ Gr = 2.10^6$
4) $Re = 379\,;\ Gr = 4.10^6$

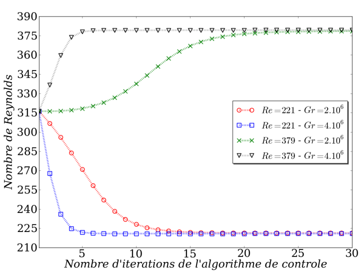
Control

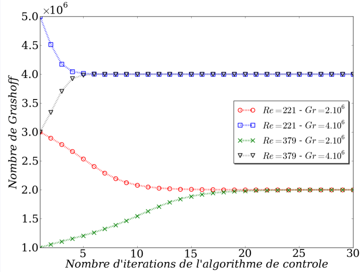
- we want to achieve a target temp. $\hat{\theta}$ corresponding to a couple Re_{targ}-Gr_{targ} by starting from a temp. θ_{init} corresponding to Re_{init}-Gr_{init}
- four target pairs of Re-Gr that do not belong to the sampling are considered

1) $Re = 221\,;\; Gr = 2.10^6$ 2) $Re = 221\,;\; Gr = 4.10^6$

3) $Re = 379\,;\; Gr = 2.10^6$ 4) $Re = 379\,;\; Gr = 4.10^6$

- algorithm convergence





Control

- error (averaged in time) between the results of the full model and those obtained with the reduced control algorithm

Reynolds number	Grashof number	Temperature error	Velocity error
221	2.10^6	5,15 %	11,7 %
221	4.10^6	4,74 %	13,8 %
379	2.10^6	5,78 %	10,8 %
379	4.10^6	5,47 %	13,6 %

  - acceptable error : about 5% for the temperature and 11-13% for the velocity, whatever the considered target
  - the optimization algorithm performs quite well

Control

- error (averaged in time) between the results of the full model and those obtained with the reduced control algorithm

Reynolds number	Grashof number	Temperature error	Velocity error
221	2.10^6	5,15 %	11,7 %
221	4.10^6	4,74 %	13,8 %
379	2.10^6	5,78 %	10,8 %
379	4.10^6	5,47 %	13,6 %

  - acceptable error : about 5% for the temperature and 11-13% for the velocity, whatever the considered target
  - the optimization algorithm performs quite well

Control

- error (averaged in time) between the results of the full model and those obtained with the reduced control algorithm

Reynolds number	Grashof number	Temperature error	Velocity error
221	2.10^6	5,15 %	11,7 %
221	4.10^6	4,74 %	13,8 %
379	2.10^6	5,78 %	10,8 %
379	4.10^6	5,47 %	13,6 %

  - acceptable error : about 5% for the temperature and 11-13% for the velocity, whatever the considered target
  - the optimization algorithm performs quite well

Control

- error (averaged in time) between the results of the full model and those obtained with the reduced control algorithm

Reynolds number	Grashof number	Temperature error	Velocity error
221	2.10^6	5,15 %	11,7 %
221	4.10^6	4,74 %	13,8 %
379	2.10^6	5,78 %	10,8 %
379	4.10^6	5,47 %	13,6 %

  - acceptable error : about 5% for the temperature and 11-13% for the velocity, whatever the considered target
  - the optimization algorithm performs quite well

Computing time necessary for the control procedure

- reduced model : about 2-3 minute with 1 proc.
- full model : estimated to be several days with 12 proc.

Summary

- control algorithm is fast (about one minute) and accurate
- however, reduced order models are again too expensive in storage requirements

Nb of kept POD modes	ROMs coefficients	POD modes	Mean fields	In all
10	11,7 Ko	9,54 Mo	10,7 Mo	**20,2 Mo**
20	23,4 Ko	19,1 Mo	21,4 Mo	**40,5 Mo**
30	35,2 Ko	28,6 Mo	32,1 Mo	**60,7 Mo**

- controllers : limited in storage and in computing power
- development of another control strategy
  - requiring less storage and less computation
  - in return, no "temporal dynamic" $\Rightarrow$ mean fields

Summary

- control algorithm is fast (about one minute) and accurate
- however, reduced order models are again too expensive in storage requirements

Nb of kept POD modes	ROMs coefficients	POD modes	Mean fields	In all
10	11,7 Ko	9,54 Mo	10,7 Mo	**20,2 Mo**
20	23,4 Ko	19,1 Mo	21,4 Mo	**40,5 Mo**
30	35,2 Ko	28,6 Mo	32,1 Mo	**60,7 Mo**

- controllers : limited in storage and in computing power
- development of another control strategy
  - requiring less storage and less computation
  - in return, no "temporal dynamic" $\Rightarrow$ mean fields

Summary

- control algorithm is fast (about one minute) and accurate
- however, reduced order models are again too expensive in storage requirements

Nb of kept POD modes	ROMs coefficients	POD modes	Mean fields	In all
10	11,7 Ko	9,54 Mo	10,7 Mo	**20,2 Mo**
20	23,4 Ko	19,1 Mo	21,4 Mo	**40,5 Mo**
30	35,2 Ko	28,6 Mo	32,1 Mo	**60,7 Mo**

- controllers : limited in storage and in computing power
- development of another control strategy
  - requiring less storage and less computation
  - in return, no "temporal dynamic" $\Rightarrow$ mean fields

Summary

- control algorithm is fast (about one minute) and accurate
- however, reduced order models are again too expensive in storage requirements

Nb of kept POD modes	ROMs coefficients	POD modes	Mean fields	In all
10	11,7 Ko	9,54 Mo	10,7 Mo	**20,2 Mo**
20	23,4 Ko	19,1 Mo	21,4 Mo	**40,5 Mo**
30	35,2 Ko	28,6 Mo	32,1 Mo	**60,7 Mo**

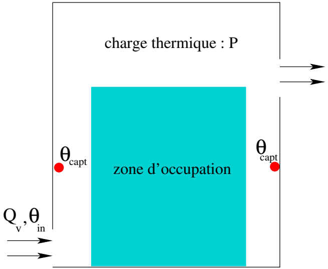
- controllers : limited in storage and in computing power
- development of another control strategy
  - requiring less storage and less computation
  - in return, no "temporal dynamic" $\Rightarrow$ mean fields

Goal

- Control the temperature θ_{zone} in the occupied zone, which depends on the building thermal load P (solar gains, occupant gains. . .) in the room, by modifying the injected air flow rate Q_v

Principle of actual controllers

- Temperature measurement with sensors located usually close to the walls
- While the temperature measured by the sensors, θ_{sensor}, is different from the desired temperature θ_{target}, the injected air flow rate Q_v is modified
- but, the temperature θ_{zone} in the occupied zone is **unknown** and **different**

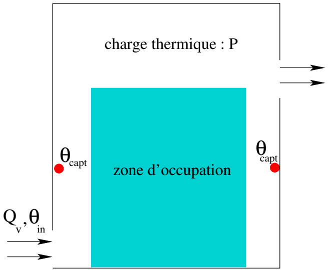


Goal

- Control the temperature θ_{zone} in the occupied zone, which depends on the building thermal load P (solar gains, occupant gains. . .) in the room, by modifying the injected air flow rate Q_v

Principle of actual controllers

- Temperature measurement with sensors located usually close to the walls
- While the temperature measured by the sensors, θ_{sensor}, is different from the desired temperature θ_{target}, the injected air flow rate Q_v is modified
- but, the temperature θ_{zone} in the occupied zone is **unknown** and **different**



Idea

- Obtain the temperature (and even the velocity) in the occupied zone with POD
- Add two more steps in the controller program

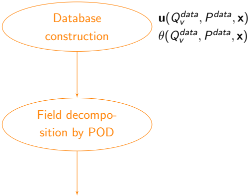
OFFLINE PROCEDURE

Database construction

- The database is built :

  ✓ with flow simulations obtained for several Q_v^{data} and several thermal loads P^{data}

Offline procedure



- The database is built :

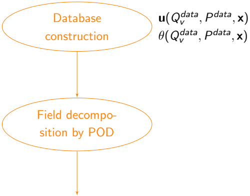
  ✓ with flow simulations obtained for several Q_v^{data} and several thermal loads P^{data}

- POD decomposition applied to the velocity and temperature fields :

$$\mathbf{u}(Q_v^{data}, P^{data}, \mathbf{x}) = \sum_{i=1}^{N_u} a_i^u(Q_v^{data}, P^{data})\, \boldsymbol{\Phi}_i^u(\mathbf{x})$$

$$\theta(Q_v^{data}, P^{data}, \mathbf{x}) = \sum_{i=1}^{N_\theta} a_i^\theta(Q_v^{data}, P^{data})\, \boldsymbol{\Phi}_i^\theta(\mathbf{x})$$

Offline procedure



- The database is built :

  ✓ with flow simulations obtained for several Q_v^{data} and several thermal loads P^{data}

- POD decomposition applied to the velocity and temperature fields :

$$\mathbf{u}(Q_v^{data}, P^{data}, \mathbf{x}) = \sum_{i=1}^{N_u} a_i^u(Q_v^{data}, P^{data})\ \mathbf{\Phi}_i^u(\mathbf{x})$$

$$\theta(Q_v^{data}, P^{data}, \mathbf{x}) = \sum_{i=1}^{N_\theta} a_i^\theta(Q_v^{data}, P^{data})\ \Phi_i^\theta(\mathbf{x})$$

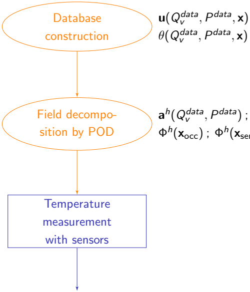
$\Rightarrow$ Computationaly expensive step, but it is done *before* the control loop

$\Rightarrow$ The following variables only are embedded in the sensor :

- $\mathbf{a}^u(Q_v^{data}, P^{data})$; $\mathbf{\Phi}^u(\mathbf{x}_{\text{occ}})$; $\mathbf{\Phi}^u(\mathbf{x}_{\text{sensor}})$
- $\mathbf{a}^\theta(Q_v^{data}, P^{data})$; $\Phi^\theta(\mathbf{x}_{\text{occ}})$; $\Phi^\theta(\mathbf{x}_{\text{sensor}})$

Online procedure

- Temperature measurement with the sensors (close to the walls) : $\theta_{\text{sensor}}()$



ONLINE PROCEDURE

- Temperature measurement with the sensors (close to the walls) : $\theta_{\text{sensor}}()$

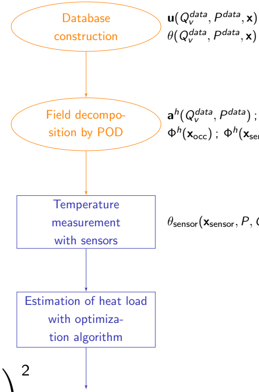
- The heat load P is estimated by solving the optimization problem :

$$\min_{P} \mathcal{J}(P, Q_v, \mathbf{x}_{\text{sensor}})$$

where the cost functional $\mathcal{J}$ is :

$$\mathcal{J} = \frac{1}{2}\sum_{j=1}^{M}\left(\theta_{\text{sensor}}(\mathbf{x}_{\text{sensor},j}) - \underbrace{\sum_{i=1}^{N_\theta} a_i^\theta(Q_v, P)\,\Phi_i^\theta(\mathbf{x}_{\text{sensor},j})}_{=\theta^{POD}(Q_v, P, \mathbf{x}_{\text{sensor},j})}\right)^2$$

M : number of sensors



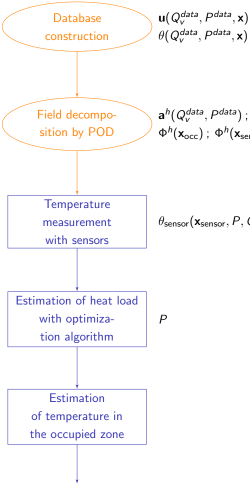
Online procedure

- Temperature assessment in the control zone θ_{zone} :

$$\theta_{\text{zone}}(\mathbf{x}_{\text{occ}}) = \sum_{i=1}^{N_\theta} a_i^\theta(Q_v, P)\,\Phi_i^\theta(\mathbf{x}_{\text{occ}})$$

<u>Remark</u> : The velocity in the control zone $\mathbf{u}_{\text{occ}}$ can be calculated :

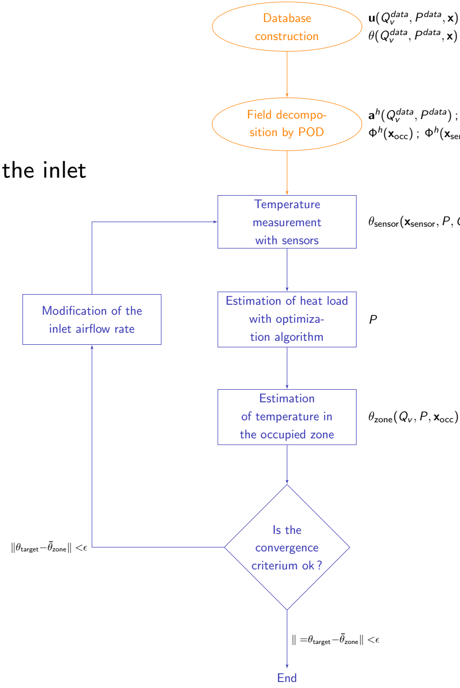
$$\mathbf{u}_{\text{occ}}(\mathbf{x}_{\text{occ}}) = \sum_{i=1}^{N_\theta} a_i^u(Q_v, P)\,\Phi_i^u(\mathbf{x}_{\text{occ}})$$



Online procedure

- Convergence criterium :

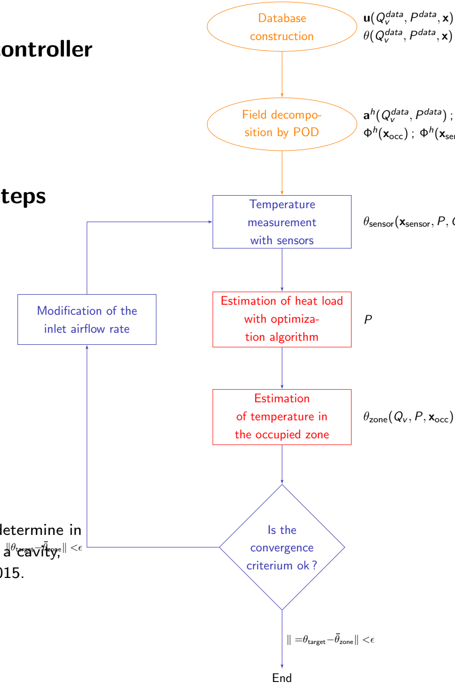
While $\|\theta_{\text{target}} - \bar{\theta}_{\text{zone}}\| < \epsilon$, the inlet airflow rate Q_v is modified



Two steps are added in the controller program

$\Downarrow$

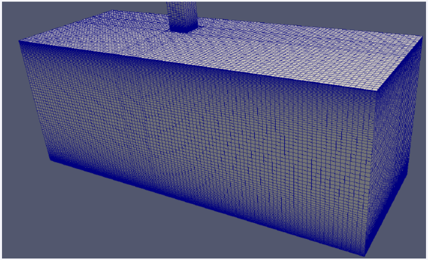
Validation of these two steps

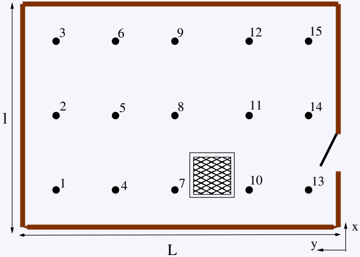


Tallet, Allery, Allard, POD approach to determine in real-time the temperature distribution in a cavity, *Building and Environment*, vol. 93(2), 2015.

3D flow in an office

- Geometry (l=2.7m, H=2.85m, L=7.2m)

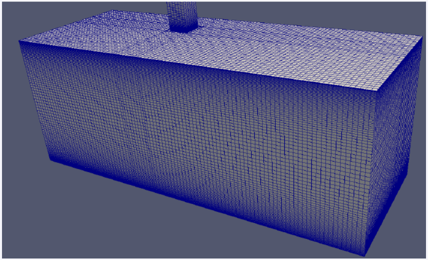


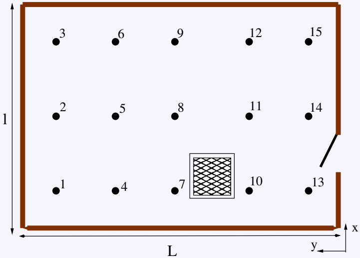


- Hypotheses :
  - incompressible and non isothermal flow
  - Boussinesq hypothesis
  - uniform heat load in the domain

3D flow in an office

- Geometry (l=2.7m, H=2.85m, L=7.2m)

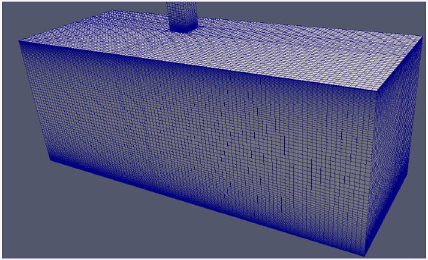


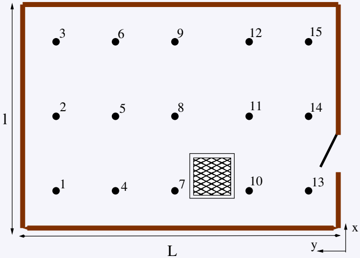


- Initial and boundary conditions :
  - walls : constant temperature 26^oC
  - outlet : homogeneous Neuman with zero heat flux
  - inlet : imposed temperature and velocity
  - initial temperature in the whole room 26^oC

3D flow in an office

- Geometry (l=2.7m, H=2.85m, L=7.2m)

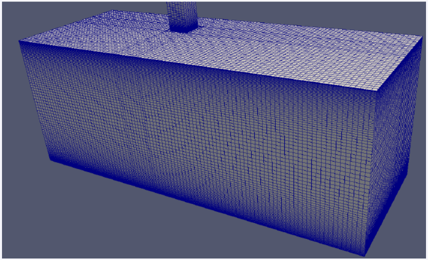


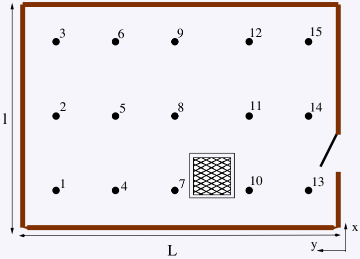


- Models and grids :
  - Saturne, 500000 nodes
  - Steady turbulence $k - epsilon$ model

3D flow in an office

- Geometry (l=2.7m, H=2.85m, L=7.2m)

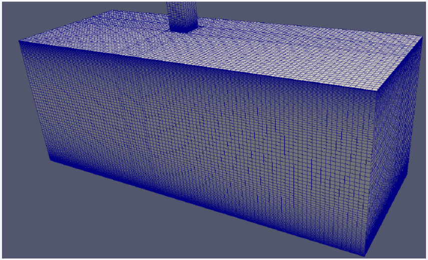


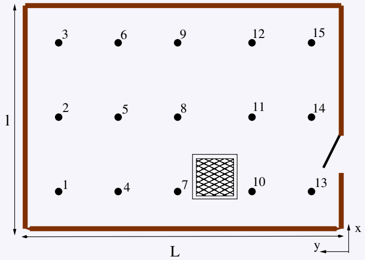


- position of the sensors :
  - one close to the switch (x/l=0.23, y/L=0.004, z/h=0.49)
  - one at the outlet

3D flow in an office

- Geometry (l=2.7m, H=2.85m, L=7.2m)



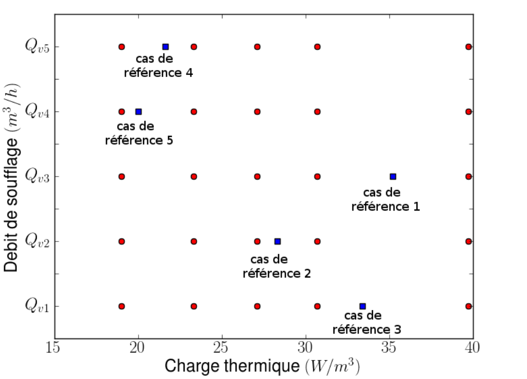


- occupied zone :

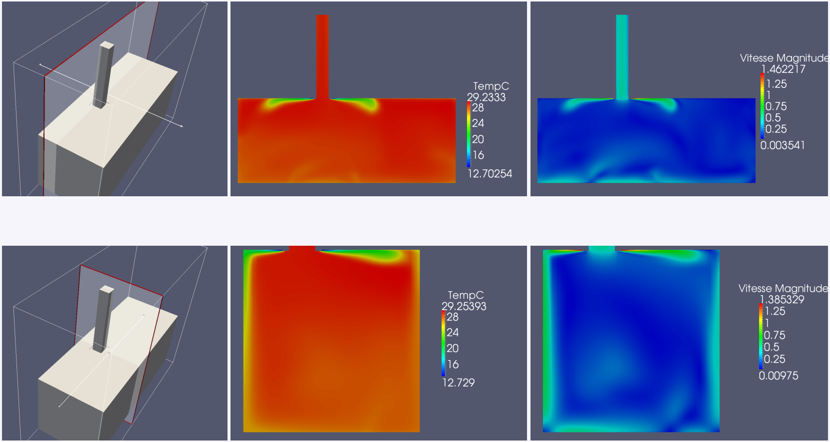
$$\begin{cases} 0.37 \leq \dfrac{x_{occ}}{l} \leq 0.74 \\ 0.41 \leq \dfrac{y_{occ}}{L} \leq 0.55 \\ 0.53 \leq \dfrac{z_{occ}}{h} \leq 0.70 \end{cases}$$

Construction of the database

- 5 airflow rates $(215m^3/h \leq Q_v \leq 590m^3/h)$, 5 thermal loads $(19W/m^3 \leq P \leq 39.7W/m^3\ )$



Isovalues of temperature and velocity along the inlet plane (case 1 – cold air injection, $P = 39.7 W/m^3$)



Results

- 2 POD modes are kept :

$$\mathbf{u}(Q_v, P, \mathbf{x}) = \sum_{i=1}^{2} a_i^u(Q_v, P)\, \mathbf{\Phi}_i^u(\mathbf{x}) \quad \text{et} \quad \theta(Q_v, P, \mathbf{x}) = \sum_{i=1}^{2} a_i^\theta(Q_v, P)\, \Phi_i^\theta(\mathbf{x})$$

Results

- 2 POD modes are kept :

$$\mathbf{u}(Q_v, P, \mathbf{x}) = \sum_{i=1}^{2} a_i^u(Q_v, P)\, \mathbf{\Phi}_i^u(\mathbf{x}) \quad \text{et} \quad \theta(Q_v, P, \mathbf{x}) = \sum_{i=1}^{2} a_i^\theta(Q_v, P)\, \Phi_i^\theta(\mathbf{x})$$

- average temperature in the occupied zone

	reference $\bar{\theta}_{\text{zone}}$	computed $\bar{\theta}_{\text{zone}}$
Case 1 $(Q_{v_3},\ P = 35.2)$	27.8	27.7
Case 2 $(Q_{v_2},\ P = 28.3)$	27.5	27.2
Case 3 $(Q_{v_1},\ P = 33.4)$	29	29.1
Case 4 $(Q_{v_5},\ P = 21.6)$	25.2	25.2
Case 5 $(Q_{v_4},\ P = 20)$	25.7	25.7

Results

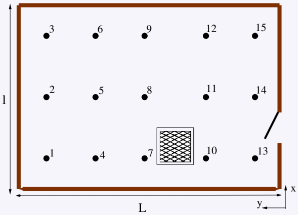
- 2 POD modes are kept :

$$\mathbf{u}(Q_v, P, \mathbf{x}) = \sum_{i=1}^{2} a_i^u(Q_v, P)\, \mathbf{\Phi}_i^u(\mathbf{x}) \quad \text{et} \quad \theta(Q_v, P, \mathbf{x}) = \sum_{i=1}^{2} a_i^\theta(Q_v, P)\, \Phi_i^\theta(\mathbf{x})$$

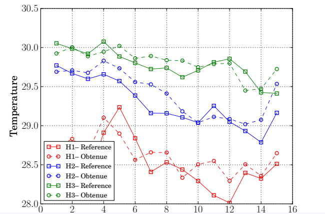
- average velocity in the occupied zone

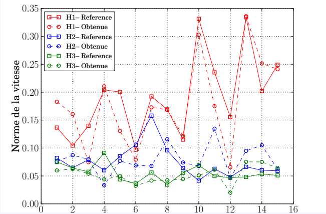
	reference $\|u\|_{\text{zone}}$	computed $\|u\|_{\text{zone}}$
Case 1 (Q_{v_3}, $P = 35.2$)	0.44	0.39
Case 2 (Q_{v_2}, $P = 28.3$)	0.40	0.35
Case 3 (Q_{v_1}, $P = 33.4$)	0.38	0.34
Case 4 (Q_{v_5}, $P = 21.6$)	0.57	0.55
Case 5 (Q_{v_4}, $P = 20$)	0.45	0.43

Position of the reference points for 3 heights H_i

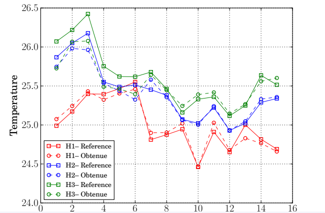


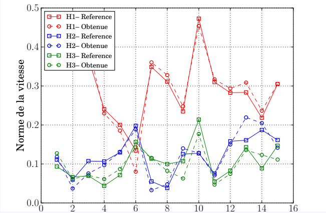
Temperature and velocity at the reference points – Case 3



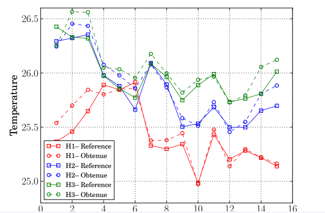


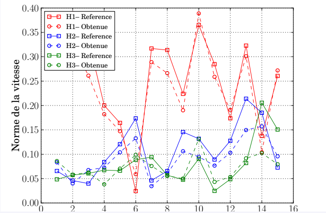
Temperature and velocity at the reference points – Case 4





Temperature and velocity at the reference points – Case 5





Conclusions

- development of another control strategy that can be embedded in the actual controllers
  - requiring less storage and less computation but no "temporal dynamic" (mean fields)
  - prediction of temperature and velocity in the occupied zone with a good accuracy
  - algorithm very fast $< 5s$

Solution 1 : Multiple Parametrized Snapshots method (MPS)

- a POD basis is generated from snapshots associated to n different values of control parameters (see first part of the talk)

Solution 2 : Interpolation on the Tangent Subspace of the Grassmann manifold (ITSGM)

- to construct a bunch of POD basis corresponding to different control parameters $(\tilde{\gamma}_1, \ldots, \tilde{\gamma}_n)$
- to interpolate them to obtain a basis valid for the parameter γ^{algo} imposed by the control algorithm

Solution 1 : Multiple Parametrized Snapshots method (MPS)

- a POD basis is generated from snapshots associated to n different values of control parameters (see first part of the talk)

Solution 2 : Interpolation on the Tangent Subspace of the Grassmann manifold (ITSGM)

- to construct a bunch of POD basis corresponding to different control parameters $(\tilde{\gamma}_1, \ldots, \tilde{\gamma}_n)$
- to interpolate them to obtain a basis valid for the parameter γ^{algo} imposed by the control algorithm
  - Lagrange or Radial Basis Function (RBF) interpolations do not necessarily produce a basis

  to ensure this property we will use a powerful interpolation method based on the calculation of geodesics in the Grassmann manifold

Solution 1 : Multiple Parametrized Snapshots method (MPS)

- a POD basis is generated from snapshots associated to n different values of control parameters (see first part of the talk)

Solution 2 : Interpolation on the Tangent Subspace of the Grassmann manifold (ITSGM)

- to construct a bunch of POD basis corresponding to different control parameters $(\tilde{\gamma}_1, \ldots, \tilde{\gamma}_n)$
- to interpolate them to obtain a basis valid for the parameter γ^{algo} imposed by the control algorithm
  - Lagrange or Radial Basis Function (RBF) interpolations do not necessarily produce a basis
  - to ensure this property we will use a powerful interpolation method based on the calculation of geodesics in the Grassmann manifold

Solution 1 : Multiple Parametrized Snapshots method (MPS)

- a POD basis is generated from snapshots associated to n different values of control parameters (see first part of the talk)

Solution 2 : Interpolation on the Tangent Subspace of the Grassmann manifold (ITSGM)

- to construct a bunch of POD basis corresponding to different control parameters $(\tilde{\gamma}_1, \ldots, \tilde{\gamma}_n)$
- to interpolate them to obtain a basis valid for the parameter γ^{algo} imposed by the control algorithm
  - Lagrange or Radial Basis Function (RBF) interpolations do not necessarily produce a basis
  - to ensure this property we will use a powerful interpolation method based on the calculation of geodesics in the Grassmann manifold

Solution 1 : Multiple Parametrized Snapshots method (MPS)

- a POD basis is generated from snapshots associated to n different values of control parameters (see first part of the talk)

Solution 2 : Interpolation on the Tangent Subspace of the Grassmann manifold (ITSGM)

- to construct a bunch of POD basis corresponding to different control parameters $(\tilde{\gamma}_1, \ldots, \tilde{\gamma}_n)$
- to interpolate them to obtain a basis valid for the parameter γ^{algo} imposed by the control algorithm
  - Lagrange or Radial Basis Function (RBF) interpolations do not necessarily produce a basis
  - to ensure this property we will use a powerful interpolation method based on the calculation of geodesics in the Grassmann manifold

Solution 1 : Multiple Parametrized Snapshots method (MPS)

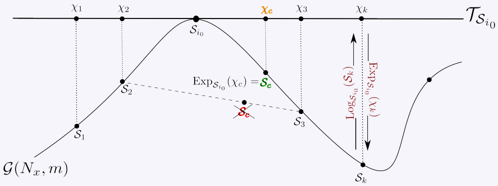
- a POD basis is generated from snapshots associated to n different values of control parameters (see first part of the talk)

Solution 2 : Interpolation on the Tangent Subspace of the Grassmann manifold (ITSGM)

- to construct a bunch of POD basis corresponding to different control parameters $(\tilde{\gamma}_1, \dots, \tilde{\gamma}_n)$
- to interpolate them to obtain a basis valid for the parameter γ^{algo} imposed by the control algorithm
  - Lagrange or Radial Basis Function (RBF) interpolations do not necessarily produce a basis
  - to ensure this property we will use a powerful interpolation method based on the calculation of geodesics in the Grassmann manifold

Principle of ITSGM

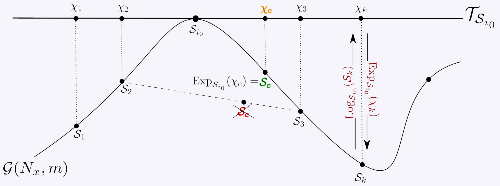
- let $\psi \in \mathbb{R}^{N_x \times m}$ denote the full-rank column matrix, whose columns provide a POD basis of a dimension m of the subspace $\mathcal{S}$ of $\mathbb{R}^{N_x}$



- the set of all these m dimensional subspaces $\mathcal{S}$ form what we call a Grassmann manifold $\mathcal{G}(m, N_x)$
  - at each point $\mathcal{S}$ of the Grassmann manifold $\mathcal{G}$ there exists a tangent space $\mathcal{T}_\mathcal{S}$ of the same dimension, with origin the point of tangency
  - the tangent space $\mathcal{T}_\mathcal{S}$ is a vector space
  - $\mathcal{T}_\mathcal{S}$ is a flat space in which interpolations can be performed as usual

Principle of ITSGM

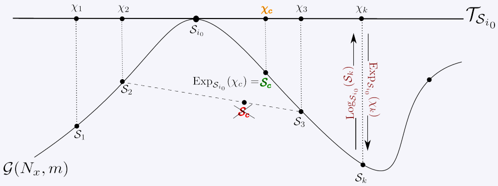
- let $\psi \in \mathbb{R}^{N_x \times m}$ denote the full-rank column matrix, whose columns provide a POD basis of a dimension m of the subspace $\mathcal{S}$ of $\mathbb{R}^{N_x}$



- the set of all these m dimensional subspaces $\mathcal{S}$ form what we call a Grassmann manifold $\mathcal{G}(m, N_x)$
  - at each point $\mathcal{S}$ of the Grassmann manifold $\mathcal{G}$ there exists a tangent space $\mathcal{T}_{\mathcal{S}}$ of the same dimension, with origin the point of tangency
  - the tangent space $\mathcal{T}_{\mathcal{S}}$ is a vector space
  - $\mathcal{T}_{\mathcal{S}}$ is a flat space in which interpolations can be performed as usual

Principle of ITSGM

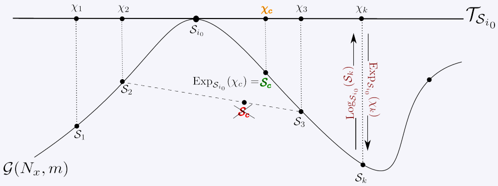
- let $\psi \in \mathbb{R}^{N_x \times m}$ denote the full-rank column matrix, whose columns provide a POD basis of a dimension m of the subspace $\mathcal{S}$ of $\mathbb{R}^{N_x}$



- the set of all these m dimensional subspaces $\mathcal{S}$ form what we call a Grassmann manifold $\mathcal{G}(m, N_x)$
  - at each point $\mathcal{S}$ of the Grassmann manifold $\mathcal{G}$ there exists a tangent space $\mathcal{T}_{\mathcal{S}}$ of the same dimension, with origin the point of tangency
  - the tangent space $\mathcal{T}_{\mathcal{S}}$ is a vector space
  - $\mathcal{T}_{\mathcal{S}}$ is a flat space in which interpolations can be performed as usual

Principle of ITSGM

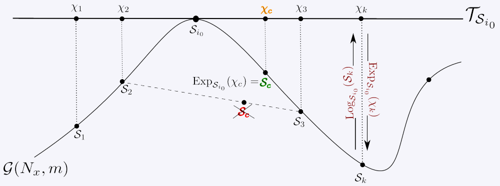
- let $\psi \in \mathbb{R}^{N_x \times m}$ denote the full-rank column matrix, whose columns provide a POD basis of a dimension m of the subspace $\mathcal{S}$ of $\mathbb{R}^{N_x}$



- the set of all these m dimensional subspaces $\mathcal{S}$ form what we call a Grassmann manifold $\mathcal{G}(m, N_x)$
  - at each point $\mathcal{S}$ of the Grassmann manifold $\mathcal{G}$ there exists a tangent space $\mathcal{T}_{\mathcal{S}}$ of the same dimension, with origin the point of tangency
  - the tangent space $\mathcal{T}_{\mathcal{S}}$ is a vector space
  - $\mathcal{T}_{\mathcal{S}}$ is a flat space in which interpolations can be performed as usual

Principle of ITSGM

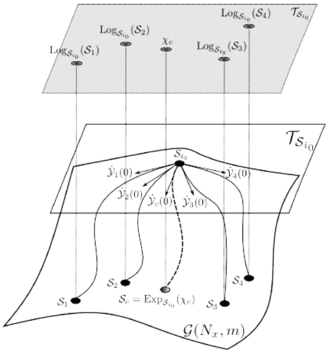
- let $\psi \in \mathbb{R}^{N_x \times m}$ denote the full-rank column matrix, whose columns provide a POD basis of a dimension m of the subspace $\mathcal{S}$ of $\mathbb{R}^{N_x}$



- the set of all these m dimensional subspaces $\mathcal{S}$ form what we call a Grassmann manifold $\mathcal{G}(m, N_x)$
  - at each point $\mathcal{S}$ of the Grassmann manifold $\mathcal{G}$ there exists a tangent space $\mathcal{T}_{\mathcal{S}}$ of the same dimension, with origin the point of tangency
  - the tangent space $\mathcal{T}_{\mathcal{S}}$ is a vector space
  - $\mathcal{T}_{\mathcal{S}}$ is a flat space in which interpolations can be performed as usual

Practical algorithm of basis adaptation on the Grassmann manifold

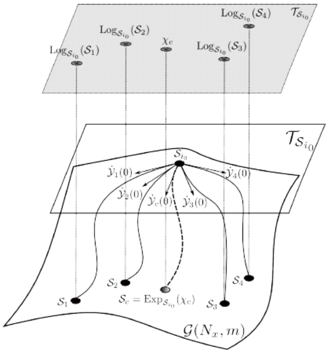
Amsallem and Farhat, An Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity, *AIAA Journal*, 2008.



Practical algorithm of basis adaptation on the Grassmann manifold

Amsallem and Farhat, An Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity, *AIAA Journal*, 2008.

1) choose a reference point $\mathcal{S}_{i_0}$ to be the origin point of the interp.



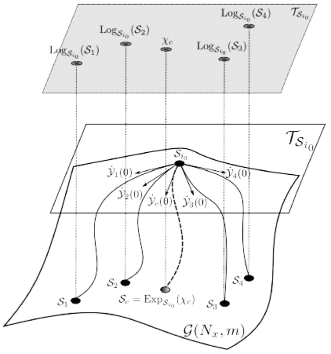
Practical algorithm of basis adaptation on the Grassmann manifold

Amsallem and Farhat, An Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity, *AIAA Journal*, 2008.

2) map each $\mathcal{S}_i$ to a matrix Γ_i representing a point χ_i of $\mathcal{T}_{\mathcal{S}_{i_0}}$ with logarithm application $\mathrm{Log}_{\mathcal{S}_{i_0}}$

$$(I - \psi_{i_0}\psi_{i_0}^T)\psi_i(\psi_{i_0}^T\psi_i)^{-1} = U_i\Sigma_i V_i^T$$

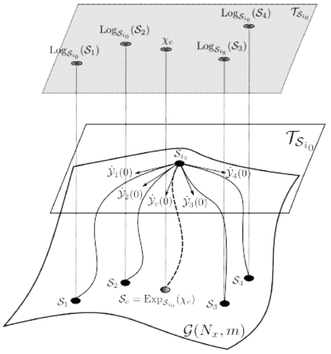
$$\text{and} \quad \Gamma_i = U_i \tan^{-1}(\Sigma_i) V_i^T$$



Practical algorithm of basis adaptation on the Grassmann manifold

Amsallem and Farhat, An Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity, *AIAA Journal*, 2008.

3) compute Γ_c associated to the control parameter γ_c by using usual interpolation method



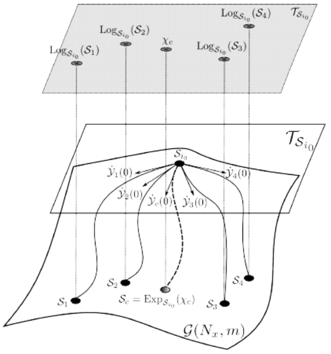
Practical algorithm of basis adaptation on the Grassmann manifold

Amsallem and Farhat, An Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity, *AIAA Journal*, 2008.

4) map Γ_c to a subspace $\mathcal{S}_c$ spanned by a matrix ψ_c with exponential application $\mathrm{Exp}_{\mathcal{S}_{i_0}}$:

$$\Gamma_c = U_c \Sigma_c V_c^T$$

and $$\psi_c = \psi_{i_0} V_c \cos(\Sigma_c) + U_c \sin(\Sigma_c)$$



Solution 3 : Proper Generalized Decomposition (PGD)

- the PGD is used here like a space-time enrichment approach
- consider a velocity and a pressure POD bases associated to a value γ_1 of the control parameter :

$$\mathbf{u}_{\gamma_1}(x,t) \simeq \sum_{j=1}^{m_u} a_j(t)\boldsymbol{\Phi}_j^u(x) \quad \text{and} \quad p_{\gamma_1}(t,x) \simeq \sum_{j=1}^{m_p} b_j(t)\Phi_{p_j}(x)$$

  - these approx. are not valid for another value γ_2
- the previous bases are enriched in the following way

$$\mathbf{u}_{\gamma_2} \simeq \mathbf{u}_{\gamma_1} + a(t)\boldsymbol{\Phi}^u(x) \quad \text{and} \quad p_{\gamma_2} \simeq p_{\gamma_1} + b(t)\Phi_p(x)$$

Solution 3 : Proper Generalized Decomposition (PGD)

- the PGD is used here like a space-time enrichment approach
- consider a velocity and a pressure POD bases associated to a value γ_1 of the control parameter :

$$\mathbf{u}_{\gamma_1}(\boldsymbol{x}, t) \simeq \sum_{j=1}^{m_u} a_j(t)\boldsymbol{\Phi}_j^u(\boldsymbol{x}) \quad \text{and} \quad p_{\gamma_1}(t, \boldsymbol{x}) \simeq \sum_{j=1}^{m_p} b_j(t)\Phi_{p_j}(\boldsymbol{x})$$

  - these approx. are not valid for another value γ_2
- the previous bases are enriched in the following way

$$\mathbf{u}_{\gamma_2} \simeq \mathbf{u}_{\gamma_1} + a(t)\boldsymbol{\Phi}^u(\boldsymbol{x}) \quad \text{and} \quad p_{\gamma_2} \simeq p_{\gamma_1} + b(t)\Phi_p(\boldsymbol{x})$$

Solution 3 : Proper Generalized Decomposition (PGD)

- the PGD is used here like a space-time enrichment approach
- consider a velocity and a pressure POD bases associated to a value γ_1 of the control parameter :

$$\mathbf{u}_{\gamma_1}(\boldsymbol{x}, t) \simeq \sum_{j=1}^{m_u} a_j(t)\boldsymbol{\Phi}_j^u(\boldsymbol{x}) \quad \text{and} \quad p_{\gamma_1}(t, \boldsymbol{x}) \simeq \sum_{j=1}^{m_p} b_j(t)\Phi_{p_j}(\boldsymbol{x})$$

  - these approx. are not valid for another value γ_2
- the previous bases are enriched in the following way

$$\mathbf{u}_{\gamma_2} \simeq \mathbf{u}_{\gamma_1} + a(t)\boldsymbol{\Phi}^u(x) \quad \text{and} \quad p_{\gamma_2} \simeq p_{\gamma_1} + b(t)\Phi_p(x)$$

  - introduction into the Navier-Stokes equations :

$$\begin{cases} \nabla \cdot \boldsymbol{\Phi}^u = 0 \\ \boldsymbol{\Phi}^u \dfrac{da}{dt} + a^2 \boldsymbol{\Phi}^u \cdot \nabla \boldsymbol{\Phi}^u + a\left(\mathbf{u}_{\gamma_1} \cdot \nabla \boldsymbol{\Phi}^u + \boldsymbol{\Phi}^u \cdot \nabla \mathbf{u}_{\gamma_1} - \dfrac{\Delta \boldsymbol{\Phi}^u}{Re}\right) + b \nabla \Phi_p = G(\mathbf{u}_{\gamma_1}, p_{\gamma_1}) + R_u \\ \boldsymbol{\Phi}^u = 0 \quad \text{sur } \Gamma \times \mathcal{I} \text{ et } \mathbf{u}_{\gamma_1} + a\boldsymbol{\Phi}^u = u_0 \quad \text{in } \Omega. \end{cases}$$

Solution 3 : Proper Generalized Decomposition (PGD)

- the PGD is used here like a space-time enrichment approach
- consider a velocity and a pressure POD bases associated to a value γ_1 of the control parameter :

$$\mathbf{u}_{\gamma_1}(\boldsymbol{x},t) \simeq \sum_{j=1}^{m_u} a_j(t)\boldsymbol{\Phi}_j^u(\boldsymbol{x}) \quad \text{and} \quad p_{\gamma_1}(t,\boldsymbol{x}) \simeq \sum_{j=1}^{m_p} b_j(t)\Phi_{p_j}(\boldsymbol{x})$$

  - these approx. are not valid for another value γ_2
- the previous bases are enriched in the following way

$$\mathbf{u}_{\gamma_2} \simeq \mathbf{u}_{\gamma_1} + a(t)\boldsymbol{\Phi}^u(\boldsymbol{x}) \quad \text{and} \quad p_{\gamma_2} \simeq p_{\gamma_1} + b(t)\Phi_p(\boldsymbol{x})$$

  - introduction into the Navier-Stokes equations :

$$\begin{cases} \nabla \cdot \boldsymbol{\Phi}^u = 0 \\ \boldsymbol{\Phi}^u \dfrac{da}{dt} + a^2\boldsymbol{\Phi}^u \cdot \nabla\boldsymbol{\Phi}^u + a\left(\mathbf{u}_{\gamma_1} \cdot \nabla\boldsymbol{\Phi}^u + \boldsymbol{\Phi}^u \cdot \nabla\mathbf{u}_{\gamma_1} - \dfrac{\Delta\boldsymbol{\Phi}^u}{Re}\right) + b\nabla\Phi_p = G(\mathbf{u}_{\gamma_1}, p_{\gamma_1}) + \mathbf{R_u} \\ \boldsymbol{\Phi}^u = 0 \quad \text{sur } \Gamma \times \mathcal{I} \text{ et } \mathbf{u}_{\gamma_1} + a\boldsymbol{\Phi}^u = u_0 \quad \text{in } \Omega. \end{cases}$$

Solution 3 : Proper Generalized Decomposition (PGD)

- the PGD is used here like a space-time enrichment approach
- consider a velocity and a pressure POD bases associated to a value γ_1 of the control parameter :

$$\mathbf{u}_{\gamma_1}(\boldsymbol{x},t) \simeq \sum_{j=1}^{m_u} a_j(t)\boldsymbol{\Phi}_j^u(\boldsymbol{x}) \quad \text{and} \quad p_{\gamma_1}(t,\boldsymbol{x}) \simeq \sum_{j=1}^{m_p} b_j(t)\Phi_{p_j}(\boldsymbol{x})$$

  - these approx. are not valid for another value γ_2
- the previous bases are enriched in the following way

$$\mathbf{u}_{\gamma_2} \simeq \mathbf{u}_{\gamma_1} + a(t)\boldsymbol{\Phi}^u(\boldsymbol{x}) \quad \text{and} \quad p_{\gamma_2} \simeq p_{\gamma_1} + b(t)\Phi_p(\boldsymbol{x})$$

  - introduction into the Navier-Stokes equations :

$$\left\{\begin{array}{l} \nabla\cdot\boldsymbol{\Phi}^u = 0 \\ \boldsymbol{\Phi}^u\dfrac{da}{dt} + a^2\boldsymbol{\Phi}^u\cdot\nabla\boldsymbol{\Phi}^u + a\left(\mathbf{u}_{\gamma_1}\cdot\nabla\boldsymbol{\Phi}^u + \boldsymbol{\Phi}^u\cdot\nabla\mathbf{u}_{\gamma_1} - \dfrac{\Delta\boldsymbol{\Phi}^u}{Re}\right) + b\nabla\Phi_p = G(\mathbf{u}_{\gamma_1}, p_{\gamma_1}) + \mathbf{R}_\mathbf{u} \\ \boldsymbol{\Phi}^u = 0 \quad \text{sur } \Gamma\times\mathcal{I} \text{ et } \mathbf{u}_{\gamma_1} + a\boldsymbol{\Phi}^u = u_0 \quad \text{in } \Omega. \end{array}\right.$$

Brief description of the Proper Generalized Decomposition (PGD)

- double Galerkin orthogonality
  - if $\{a, b\}$ are known and fixed, we search

$$\{\Phi^u, \Phi_p\} = \mathcal{S}(a, b)$$

$\mathcal{S}$ corresponds to the Galerkin projection of N.S equations onto temporal coefficients a and b

  - if $\{\Phi^u, \Phi_p\}$ are known and fixed, we seek

$$\{a, b\} = \mathcal{T}(\Phi^u, \Phi_p)$$

$\mathcal{T}$ corresponds to the Galerkin projection of N.S equations onto Φ^u and $\nabla\Phi_p$

  - $\{a, \Phi^u\}$ and $\{b, \Phi_p\}$ are optimal if they satisfy simultaneously the previous equations
  - theses equations are solved with a classical fixed point algorithm

Brief description of the Proper Generalized Decomposition (PGD)

- double Galerkin orthogonality
  - if $\{a, b\}$ are known and fixed, we search

$$\{\mathbf{\Phi}^u, \Phi_p\} = \mathcal{S}(a, b)$$

$\mathcal{S}$ corresponds to the Galerkin projection of N.S equations onto temporal coefficients a and b

  - if $\{\mathbf{\Phi}^u, \Phi_p\}$ are known and fixed, we seek

$$\{a, b\} = \mathcal{T}(\mathbf{\Phi}^u, \Phi_p)$$

$\mathcal{T}$ corresponds to the Galerkin projection of N.S equations onto $\mathbf{\Phi}^u$ and $\nabla\Phi_p$

  - $\{a, \mathbf{\Phi}^u\}$ and $\{b, \Phi_p\}$ are optimal if they satisfy simultaneously the previous equations
  - theses equations are solved with a classical fixed point algorithm

Brief description of the Proper Generalized Decomposition (PGD)

- double Galerkin orthogonality
  - if $\{a, b\}$ are known and fixed, we search

$$\{\mathbf{\Phi}^u, \Phi_p\} = \mathcal{S}(a, b)$$

$\mathcal{S}$ corresponds to the Galerkin projection of N.S equations onto temporal coefficients a and b

  - if $\{\mathbf{\Phi}^u, \Phi_p\}$ are known and fixed, we seek

$$\{a, b\} = \mathcal{T}(\mathbf{\Phi}^u, \Phi_p)$$

$\mathcal{T}$ corresponds to the Galerkin projection of N.S equations onto $\mathbf{\Phi}^u$ and $\nabla\Phi_p$

  - $\{a, \mathbf{\Phi}^u\}$ and $\{b, \Phi_p\}$ are optimal if they satisfy simultaneously the previous equations
  - theses equations are solved with a classical fixed point algorithm

Brief description of the Proper Generalized Decomposition (PGD)

- double Galerkin orthogonality
  - if $\{a, b\}$ are known and fixed, we search

$$\{\mathbf{\Phi}^u, \Phi_p\} = \mathcal{S}(a, b)$$

$\mathcal{S}$ corresponds to the Galerkin projection of N.S equations onto temporal coefficients a and b

  - if $\{\mathbf{\Phi}^u, \Phi_p\}$ are known and fixed, we seek

$$\{a, b\} = \mathcal{T}(\mathbf{\Phi}^u, \Phi_p)$$

$\mathcal{T}$ corresponds to the Galerkin projection of N.S equations onto $\mathbf{\Phi}^u$ and $\nabla\Phi_p$

  - $\{a, \mathbf{\Phi}^u\}$ and $\{b, \Phi_p\}$ are optimal if they satisfy simultaneously the previous equations
  - theses equations are solved with a classical fixed point algorithm

Brief description of the Proper Generalized Decomposition (PGD)

- double Galerkin orthogonality
  - if $\{a, b\}$ are known and fixed, we search

    $$\{\mathbf{\Phi}^u, \Phi_p\} = \mathcal{S}(a, b)$$

    $\mathcal{S}$ corresponds to the Galerkin projection of N.S equations onto temporal coefficients a and b

  - if $\{\mathbf{\Phi}^u, \Phi_p\}$ are known and fixed, we seek

    $$\{a, b\} = \mathcal{T}(\mathbf{\Phi}^u, \Phi_p)$$

    $\mathcal{T}$ corresponds to the Galerkin projection of N.S equations onto $\mathbf{\Phi}^u$ and $\nabla\Phi_p$

  - $\{a, \mathbf{\Phi}^u\}$ and $\{b, \Phi_p\}$ are optimal if they satisfy simultaneously the previous equations
  - theses equations are solved with a classical fixed point algorithm

Reduced optimal control with POD bases update

a) initialization of the algorithm :$k = 0$ et $\boldsymbol{\gamma}^{(k)} = \boldsymbol{\gamma}_{init}$

b) update the bases by using PGD or ITSGM

c) update the spatial coeff. of the state and adjoint ROMs

d) solving the state ROM $\boldsymbol{\mathcal{M}}(\boldsymbol{a}, \gamma^{(k)}) = \mathbf{0} \rightarrow \mathbf{a}$

e) solving the adjoint ROM $\boldsymbol{\mathcal{P}}(\boldsymbol{a}, \boldsymbol{\xi}, \gamma^{(k)}) = \mathbf{0} \rightarrow \beta$.

f) assessment of the descent direction $\rightarrow \mathbf{d}^{(k)} = -\nabla_\gamma J_{red}(\mathbf{a}, \boldsymbol{\beta}, \boldsymbol{\gamma}^{(k)})$

g) assessment of the step $\omega^{(k)}$ in the descent direction $d^{(k)}$

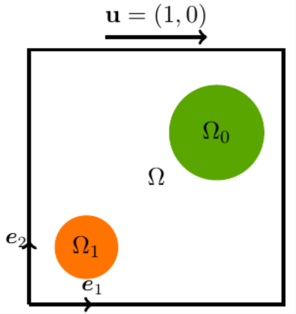
(linear search algorithm of Armijo)

f) update the control parameter $\rightarrow \boldsymbol{\gamma}^{(k+1)} = \boldsymbol{\gamma}^{(k)} + \omega^{(k)} d^{(k)}$

h) convergence criterion : if $\|J_{red}(\mathbf{a}, \boldsymbol{\gamma}^{(k+1)})\| > \varepsilon$, return to step b)

Application : 2D lid driven submitted to body forces

- two external forces $\mathbf{f_0}$ and $\mathbf{f_1}$: $\mathbf{f_0} = \gamma^0 exp(-t)\chi_{\Omega_0}(\mathbf{e_1}+\mathbf{e_2})$ and $\mathbf{f_1} = \gamma^1 \chi_{\Omega_1}(\mathbf{e_1}+\mathbf{e_2})$
- temporal domain $\mathcal{I} = [0, 1]$
- at t=0, the fluid is at rest
- finite element code : Fenics
- Taylor Hood $\mathbb{P}_2/\mathbb{P}_1$, non uniform grid, 17728 triangles



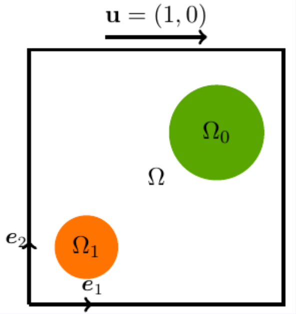
Oulghelou, Allery : A fast and robust sub-optimal control approach using reduced order model adaptation techniques, *Applied Mathematics and Computation*, vol 333, 2018.

Objective

- $\gamma = (\gamma^0, \gamma^1)$ are the control parameters
- starting from the flow associated to $\gamma_{init} = (\gamma^0_{init}, \gamma^1_{init})$, we want to obtain the param. $\hat{\gamma} = (\hat{\gamma}^0, \hat{\gamma}^1)$ corresponding to the target flow $\hat{u}$.

Application : 2D lid driven submitted to body forces

- two external forces $\mathbf{f_0}$ and $\mathbf{f_1}$: $\mathbf{f_0} = \gamma^0 exp(-t)\chi_{\Omega_0}(\mathbf{e_1}+\mathbf{e_2})$ and $\mathbf{f_1} = \gamma^1\chi_{\Omega_1}(\mathbf{e_1}+\mathbf{e_2})$
- temporal domain $\mathcal{I} = [0, 1]$
- at t=0, the fluid is at rest
- finite element code : Fenics
- Taylor Hood $\mathbb{P}_2/\mathbb{P}_1$, non uniform grid, 17728 triangles



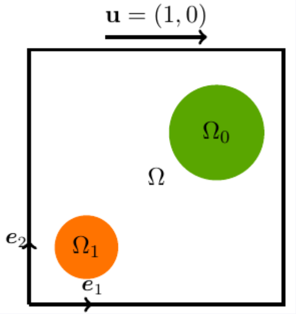
Oulghelou, Allery : A fast and robust sub-optimal control approach using reduced order model adaptation techniques, *Applied Mathematics and Computation*, vol 333, 2018.

Objective

- $\gamma = (\gamma^0, \gamma^1)$ are the control parameters
- starting from the flow associated to $\gamma_{init} = (\gamma^0_{init}, \gamma^1_{init})$, we want to obtain the param. $\hat{\gamma} = (\hat{\gamma}^0, \hat{\gamma}^1)$ corresponding to the target flow $\hat{\mathbf{u}}$.

Application : 2D lid driven submitted to body forces

- two external forces $\mathbf{f_0}$ and $\mathbf{f_1}$: $\mathbf{f_0} = \gamma^0 exp(-t)\chi_{\Omega_0}(\mathbf{e_1}+\mathbf{e_2})$ and $\mathbf{f_1} = \gamma^1\chi_{\Omega_1}(\mathbf{e_1}+\mathbf{e_2})$
- temporal domain $\mathcal{I} = [0,1]$
- at t=0, the fluid is at rest
- finite element code : Fenics
- Taylor Hood $\mathbb{P}_2/\mathbb{P}_1$, non uniform grid, 17728 triangles

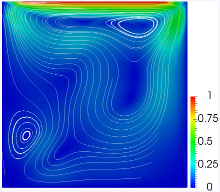


Oulghelou, Allery : A fast and robust sub-optimal control approach using reduced order model adaptation techniques, *Applied Mathematics and Computation*, vol 333, 2018.

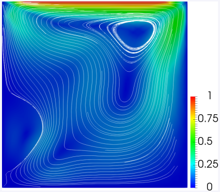
Objective

- $\gamma = (\gamma^0, \gamma^1)$ are the control parameters
- starting from the flow associated to $\gamma_{init} = (\gamma^0_{init}, \gamma^1_{init})$, we want to obtain the param. $\hat{\gamma} = (\hat{\gamma}^0, \hat{\gamma}^1)$ corresponding to the target flow $\hat{\mathbf{u}}$.

Streamlines for the initial flow $\gamma_{init} = (\gamma^0_{init}, \gamma^1_{init}) = (1, -1)$

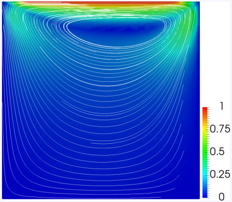


a) at t=T/2

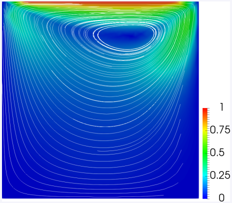


b) at t=T

Streamlines for the target flow $\hat{\gamma} = (\hat{\gamma}^0, \hat{\gamma}^1) = (0, 0)$



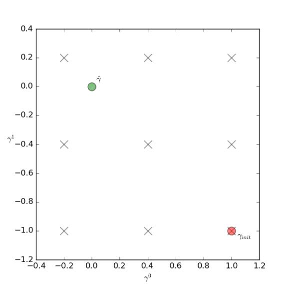
a) at t=T/2



b) at t=T

Construction of the POD bases

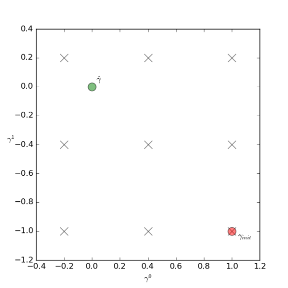
- parameters $\gamma = (\gamma^0, \gamma^1)$ used to build the sampling POD bases



  - this sampling is not necessary for the PGD approach
- POD basis is built with 400 snap. evenly distributed on the time
- the dimension of each POD basis is 10
- for MPS method, all snapshots associated at all operating points are used to generate the POD basis

Construction of the POD bases

- parameters $\gamma = (\gamma^0, \gamma^1)$ used to build the sampling POD bases

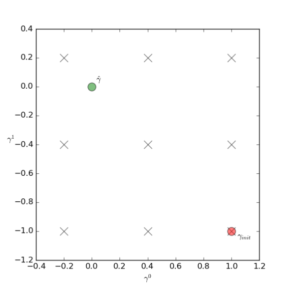


  - this sampling is not necessary for the PGD approach

- POD basis is built with 400 snap. evenly distributed on the time
- the dimension of each POD basis is 10
- for MPS method, all snapshots associated at all operating points are used to generate the POD basis

Construction of the POD bases

- parameters $\gamma = (\gamma^0, \gamma^1)$ used to build the sampling POD bases

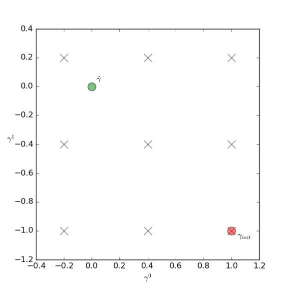


  - this sampling is not necessary for the PGD approach

- POD basis is built with 400 snap. evenly distributed on the time
- the dimension of each POD basis is 10
- for MPS method, all snapshots associated at all operating points are used to generate the POD basis

Construction of the POD bases

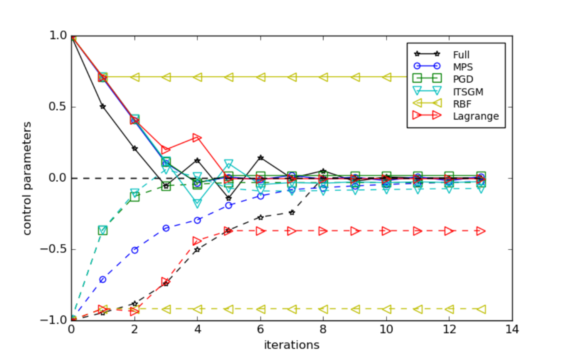
- parameters $\gamma = (\gamma^0, \gamma^1)$ used to build the sampling POD bases



  - this sampling is not necessary for the PGD approach

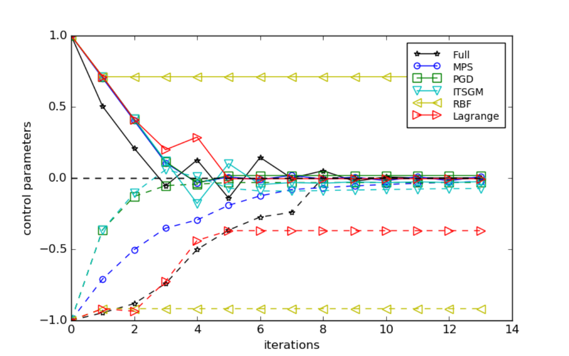
- POD basis is built with 400 snap. evenly distributed on the time
- the dimension of each POD basis is 10
- for MPS method, all snapshots associated at all operating points are used to generate the POD basis

Evolution of control parameters



- convergence towards the target values for ITSGM, PGD and MPS methods
- non convergence for Lagrange and RBF methods

Evolution of control parameters



- convergence towards the target values for ITSGM, PGD and MPS methods
- non convergence for Lagrange and RBF methods

CPU time and % error at the end of the control algorithm

Method	$\mathcal{R}_{online}$	$\mathcal{R}_{offline+online}$	% error
full	1	1	0.01%
MPS	2972	20.3	3.69%
PGD	146	146	3.81%
ITSGM	1400	20.1	2.12%

- percentage error : $\varepsilon = \int_0^T \hat{\varepsilon} dt \quad \text{where} \quad \hat{\varepsilon} = 100 \times \frac{||\hat{u} - u||_{L^2(\Omega)}}{||\hat{u}||_{L^2(\Omega)}}$

- CPU time ratio : $\mathcal{R} = T_{full} / T_{method}$

- ITSGM gives more accurate results
- the online gain is important for MPS and ITSGM
- if the offline time (associated to the construction of POD basis) is considered, the PGD becomes more advantageous

CPU time and % error at the end of the control algorithm

Method	$\mathcal{R}_{online}$	$\mathcal{R}_{offline+online}$	% error
full	1	1	0.01%
MPS	2972	20.3	3.69%
PGD	146	146	3.81%
ITSGM	1400	20.1	2.12%

- percentage error : $\varepsilon = \int_0^T \hat{\varepsilon} dt \quad \text{where} \quad \hat{\varepsilon} = 100 \times \frac{||\hat{u} - u||_{L^2(\Omega)}}{||\hat{u}||_{L^2(\Omega)}}$

- CPU time ratio : $\mathcal{R} = T_{full} / T_{method}$

- ITSGM gives more accurate results
- the online gain is important for MPS and ITSGM
- if the offline time (associated to the construction of POD basis) is considered, the PGD becomes more advantageous

CPU time and % error at the end of the control algorithm

Method	$\mathcal{R}_{online}$	$\mathcal{R}_{offline+online}$	% error
full	1	1	0.01%
MPS	2972	20.3	3.69%
PGD	146	146	3.81%
ITSGM	1400	20.1	2.12%

- percentage error : $\varepsilon = \int_0^T \hat{\varepsilon} dt \quad \text{where} \quad \hat{\varepsilon} = 100 \times \frac{||\hat{u} - u||_{L^2(\Omega)}}{||\hat{u}||_{L^2(\Omega)}}$

- CPU time ratio : $\mathcal{R} = T_{full} / T_{method}$

- ITSGM gives more accurate results
- the online gain is important for MPS and ITSGM
- if the offline time (associated to the construction of POD basis) is considered, the PGD becomes more advantageous

CPU time and % error at the end of the control algorithm

Method	$\mathcal{R}_{online}$	$\mathcal{R}_{offline+online}$	% error
full	1	1	0.01%
MPS	2972	20.3	3.69%
PGD	146	146	3.81%
ITSGM	1400	20.1	2.12%

- percentage error : $\varepsilon = \int_0^T \hat{\varepsilon} dt \quad \text{where} \quad \hat{\varepsilon} = 100 \times \frac{||\hat{u} - u||_{L^2(\Omega)}}{||\hat{u}||_{L^2(\Omega)}}$

- CPU time ratio : $\mathcal{R} = T_{full} / T_{method}$

- ITSGM gives more accurate results
- the online gain is important for MPS and ITSGM
- if the offline time (associated to the construction of POD basis) is considered, the PGD becomes more advantageous

Conclusion of this part

- simulations in a few minutes with proper accuracy
- fast simulations, but not in real time (due to ROM construction for each new parameter value)
- we propose a faster method based directly on the interpolation of solutions previously compressed by POD

Conclusion of this part

- simulations in a few minutes with proper accuracy
- fast simulations, but not in real time (due to ROM construction for each new parameter value)
- we propose a faster method based directly on the interpolation of solutions previously compressed by POD

Conclusion of this part

- simulations in a few minutes with proper accuracy
- fast simulations, but not in real time (due to ROM construction for each new parameter value)
- we propose a faster method based directly on the interpolation of solutions previously compressed by POD

Bi-CITSGM Method

- consider the set of parameterized snapshot matrices :

$$\boldsymbol{S}(\theta_i) = \{y_1(\theta_i), \ldots, y_{N_s}(\theta_i)\} \in \mathbb{R}^{N_x \times N_s} \qquad \text{where} \quad i = 1, \ldots, N_p$$

  - $y_j(\theta_i)$ is the sol. at time t_j of a parameterized physical pb
- goal : *approximate $\boldsymbol{S}(\widetilde{\theta})$ for $\widetilde{\theta} \neq \theta_i$, without resorting to the full model*
- standard polynomial interpolation methods
- proposed approach : Bi-CITSGM (Hyper Bi-Calibrated Interpolation on the Tangent Space of the Grassmann Manifold)
  - ITSGM interpolation
  - solving a constrained optimization problem

Bi-CITSGM Method

- consider the set of parameterized snapshot matrices :

$$\boldsymbol{S}(\theta_i) = \{y_1(\theta_i), \ldots, y_{N_s}(\theta_i)\} \in \mathbb{R}^{N_x \times N_s} \qquad \text{where} \quad i = 1, \ldots, N_p$$

  - $y_j(\theta_i)$ is the sol. at time t_j of a parameterized physical pb

- goal : *approximate $\boldsymbol{S}(\widetilde{\theta})$ for $\widetilde{\theta} \neq \theta_i$, without resorting to the full model*

- standard polynomial interpolation methods

  - effective for pbs with a linear behaviour
  - ineffective for pbs with a non-linear behaviour

- proposed approach : Bi-CITSGM (Hyper Bi-Calibrated Interpolation on the Tangent Space of the Grassmann Manifold)

  - ITSGM interpolation
  - solving a constrained optimization problem

Bi-CITSGM Method

- consider the set of parameterized snapshot matrices :

$$\boldsymbol{S}(\theta_i) = \{y_1(\theta_i), \ldots, y_{N_s}(\theta_i)\} \in \mathbb{R}^{N_x \times N_s} \qquad \text{where} \quad i = 1, \ldots, N_p$$

  - $y_j(\theta_i)$ is the sol. at time t_j of a parameterized physical pb
- goal : *approximate $\boldsymbol{S}(\widetilde{\theta})$ for $\widetilde{\theta} \neq \theta_i$, without resorting to the full model*
- standard polynomial interpolation methods
  - effective for pbs with a linear behaviour
  - ineffective for pbs with a non-linear behaviour
- proposed approach : Bi-CITSGM (Hyper Bi-Calibrated Interpolation on the Tangent Space of the Grassmann Manifold)
  - ITSGM interpolation
  - solving a constrained optimization problem

Bi-CITSGM Method

- consider the set of parameterized snapshot matrices :

$$\boldsymbol{S}(\theta_i) = \{y_1(\theta_i), \ldots, y_{N_s}(\theta_i)\} \in \mathbb{R}^{N_x \times N_s} \qquad \text{where} \quad i = 1, \ldots, N_p$$

  - $y_j(\theta_i)$ is the sol. at time t_j of a parameterized physical pb
- goal : *approximate $\boldsymbol{S}(\widetilde{\theta})$ for $\widetilde{\theta} \neq \theta_i$, without resorting to the full model*
- standard polynomial interpolation methods
  - effective for pbs with a linear behaviour
  - ineffective for pbs with a non-linear behaviour
- proposed approach : Bi-CITSGM (Hyper Bi-Calibrated Interpolation on the Tangent Space of the Grassmann Manifold)
  - ITSGM interpolation
  - solving a constrained optimization problem

Bi-CITSGM Method

- consider the set of parameterized snapshot matrices :

$$\boldsymbol{S}(\theta_i) = \{y_1(\theta_i), \ldots, y_{N_s}(\theta_i)\} \in \mathbb{R}^{N_x \times N_s} \qquad \text{where} \quad i = 1, \ldots, N_p$$

  - $y_j(\theta_i)$ is the sol. at time t_j of a parameterized physical pb

- goal : *approximate $\boldsymbol{S}(\widetilde{\theta})$ for $\widetilde{\theta} \neq \theta_i$, without resorting to the full model*

- standard polynomial interpolation methods
  - effective for pbs with a linear behaviour
  - ineffective for pbs with a non-linear behaviour

- proposed approach : Bi-CITSGM (Hyper Bi-Calibrated Interpolation on the Tangent Space of the Grassmann Manifold)
  - ITSGM interpolation
  - solving a constrained optimization problem

Bi-CITSGM Method

- consider the set of parameterized snapshot matrices :

$$\boldsymbol{S}(\theta_i) = \{y_1(\theta_i), \ldots, y_{N_s}(\theta_i)\} \in \mathbb{R}^{N_x \times N_s} \qquad \text{where} \quad i = 1, \ldots, N_p$$

  - $y_j(\theta_i)$ is the sol. at time t_j of a parameterized physical pb

- goal : *approximate $\boldsymbol{S}(\widetilde{\theta})$ for $\widetilde{\theta} \neq \theta_i$, without resorting to the full model*

- standard polynomial interpolation methods
  - effective for pbs with a linear behaviour
  - ineffective for pbs with a non-linear behaviour

- proposed approach : Bi-CITSGM (Hyper Bi-Calibrated Interpolation on the Tangent Space of the Grassmann Manifold)
  - ITSGM interpolation
  - solving a constrained optimization problem

Bi-CITSGM Method

- consider the set of parameterized snapshot matrices :

$$\boldsymbol{S}(\theta_i) = \{y_1(\theta_i), \ldots, y_{N_s}(\theta_i)\} \in \mathbb{R}^{N_x \times N_s} \qquad \text{where} \quad i = 1, \ldots, N_p$$

  - $y_j(\theta_i)$ is the sol. at time t_j of a parameterized physical pb

- goal : *approximate $\boldsymbol{S}(\widetilde{\theta})$ for $\widetilde{\theta} \neq \theta_i$, without resorting to the full model*

- standard polynomial interpolation methods
  - effective for pbs with a linear behaviour
  - ineffective for pbs with a non-linear behaviour

- proposed approach : Bi-CITSGM (Hyper Bi-Calibrated Interpolation on the Tangent Space of the Grassmann Manifold)
  - ITSGM interpolation
  - solving a constrained optimization problem

Bi-CITSGM Method – "Offline" Steps

- For each param. θ_i, the snapshot matrices are decomposed using POD

$$\boldsymbol{S}(\theta_i) \approx \underbrace{\boldsymbol{U}_i}_{\substack{\text{spatial basis} \\ \text{size } N_x \times q}} \underbrace{\boldsymbol{\Sigma}_i}_{\substack{\text{singular values} \\ \text{size } q \times q}} \underbrace{\boldsymbol{V}_i^T}_{\substack{\text{temporal basis} \\ \text{size } N_s \times q}}$$

Bi-CITSGM Method – "Online" Steps

- interpolate the singular values (RBF, Lagrange, spline, etc.) and obtain $\tilde{\Sigma}$
- interpolate the singular vector matrices by ITSGM $[U_1], \ldots, [U_{N_\theta}]$ and obtain $[\tilde{U}]$

Bi-CITSGM Method – "Offline" Steps

- For each param. θ_i, the snapshot matrices are decomposed using POD

$$\boldsymbol{S}(\theta_i) \approx \boldsymbol{U}_i \boldsymbol{\Sigma}_i \boldsymbol{V}_i^T$$

spatial basis	singular values	temporal basis
size $N_x \times q$	size $q \times q$	size $N_s \times q$

Bi-CITSGM Method – "Online" Steps

- interpolate the singular values (RBF, Lagrange, spline, etc.) and obtain $\widetilde{\Sigma}$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{U}_1], \ldots, [\boldsymbol{U}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{U}}]$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{V}_1], \ldots, [\boldsymbol{V}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{V}}]$

Bi-CITSGM Method – "Offline" Steps

- For each param. θ_i, the snapshot matrices are decomposed using POD

$$\boldsymbol{S}(\theta_i) \approx \underbrace{\boldsymbol{U}_i}_{\substack{\text{spatial basis} \\ \text{size } N_x \times q}} \underbrace{\boldsymbol{\Sigma}_i}_{\substack{\text{singular values} \\ \text{size } q \times q}} \underbrace{\boldsymbol{V}_i^T}_{\substack{\text{temporal basis} \\ \text{size } N_s \times q}}$$

Bi-CITSGM Method – "Online" Steps

- interpolate the singular values (RBF, Lagrange, spline, etc.) and obtain $\widetilde{\Sigma}$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{U}_1], \ldots, [\boldsymbol{U}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{U}}]$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{V}_1], \ldots, [\boldsymbol{V}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{V}}]$

Bi-CITSGM Method – "Offline" Steps

- For each param. θ_i, the snapshot matrices are decomposed using POD

$$\boldsymbol{S}(\theta_i) \approx \boldsymbol{U}_i \, \boldsymbol{\Sigma}_i \, \boldsymbol{V}_i^T$$

spatial basis	singular values	temporal basis
size $N_x \times q$	size $q \times q$	size $N_s \times q$

Bi-CITSGM Method – "Online" Steps

- interpolate the singular values (RBF, Lagrange, spline, etc.) and obtain $\widetilde{\Sigma}$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{U}_1], \ldots, [\boldsymbol{U}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{U}}]$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{V}_1], \ldots, [\boldsymbol{V}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{V}}]$

Bi-CITSGM Method – "Offline" Steps

- For each param. θ_i, the snapshot matrices are decomposed using POD

$$\boldsymbol{S}(\theta_i) \approx \underbrace{\boldsymbol{U}_i}_{\substack{\text{spatial basis} \\ \text{size } N_x \times q}} \underbrace{\boldsymbol{\Sigma}_i}_{\substack{\text{singular values} \\ \text{size } q \times q}} \underbrace{\boldsymbol{V}_i^T}_{\substack{\text{temporal basis} \\ \text{size } N_s \times q}}$$

Bi-CITSGM Method – "Online" Steps

- interpolate the singular values (RBF, Lagrange, spline, etc.) and obtain $\widetilde{\Sigma}$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{U}_1], \ldots, [\boldsymbol{U}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{U}}]$
- interpolate the singular vector matrices by ITSGM $[\boldsymbol{V}_1], \ldots, [\boldsymbol{V}_{N_p}]$ and obtain $[\widetilde{\boldsymbol{V}}]$

Bi-CITSGM Method – "Online" Steps

- the expression $\boldsymbol{S}(\widetilde{\theta}) \stackrel{!}{=} \widetilde{\boldsymbol{U}}\widetilde{\Sigma}\widetilde{\boldsymbol{V}}^T$ is **incorrect !!**
  - the modes of $\widetilde{\boldsymbol{U}}$ and $\widetilde{\boldsymbol{V}}$ do not follow the order of the singular values $\widetilde{\Sigma}$
  - necessity to introduce orthogonal matrices K and Q such that $\boldsymbol{S}(\widetilde{\theta}) = \widetilde{\boldsymbol{U}}K\widetilde{\Sigma}Q^T\widetilde{\boldsymbol{V}}^T$
  - K and Q are solutions to constrained optimization problems
  - analytical expression of the orthogonal matrices K and Q

Bi-CITSGM Method – "Online" Steps

- the expression $\boldsymbol{S}(\widetilde{\theta}) \stackrel{!}{=} \widetilde{\boldsymbol{U}}\widetilde{\Sigma}\widetilde{\boldsymbol{V}}^T$ is **incorrect !!**
  - the modes of $\widetilde{\boldsymbol{U}}$ and $\widetilde{\boldsymbol{V}}$ do not follow the order of the singular values $\widetilde{\Sigma}$
  - necessity to introduce orthogonal matrices K and Q such that $\boldsymbol{S}(\widetilde{\theta}) = \widetilde{\boldsymbol{U}}K\widetilde{\Sigma}Q^T\widetilde{\boldsymbol{V}}^T$
  - K and Q are solutions to constrained optimization problems
  - analytical expression of the orthogonal matrices K and Q

Bi-CITSGM Method – "Online" Steps

- the expression $\boldsymbol{S}(\widetilde{\theta}) \stackrel{!}{=} \widetilde{\boldsymbol{U}}\widetilde{\Sigma}\widetilde{\boldsymbol{V}}^T$ is **incorrect !!**
  - the modes of $\widetilde{\boldsymbol{U}}$ and $\widetilde{\boldsymbol{V}}$ do not follow the order of the singular values $\widetilde{\Sigma}$
  - necessity to introduce orthogonal matrices K and Q such that $\boldsymbol{S}(\widetilde{\theta}) = \widetilde{\boldsymbol{U}}K\widetilde{\Sigma}Q^T\widetilde{\boldsymbol{V}}^T$
  - K and Q are solutions to constrained optimization problems
  - analytical expression of the orthogonal matrices K and Q

Bi-CITSGM Method – "Online" Steps

- the expression $\boldsymbol{S}(\widetilde{\theta}) \stackrel{!}{=} \widetilde{\boldsymbol{U}}\widetilde{\Sigma}\widetilde{\boldsymbol{V}}^T$ is **incorrect !!**
  - the modes of $\widetilde{\boldsymbol{U}}$ and $\widetilde{\boldsymbol{V}}$ do not follow the order of the singular values $\widetilde{\Sigma}$
  - necessity to introduce orthogonal matrices K and Q such that $\boldsymbol{S}(\widetilde{\theta}) = \widetilde{\boldsymbol{U}}K\widetilde{\Sigma}Q^T\widetilde{\boldsymbol{V}}^T$
  - K and Q are solutions to constrained optimization problems
  - analytical expression of the orthogonal matrices K and Q

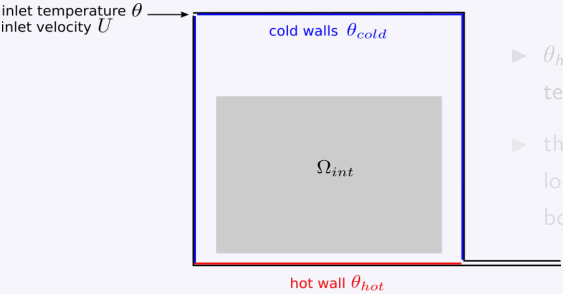
Bi-CITSGM Method – "Online" Steps

- the expression $\boldsymbol{S}(\widetilde{\theta}) \stackrel{!}{=} \widetilde{\boldsymbol{U}}\widetilde{\Sigma}\widetilde{\boldsymbol{V}}^T$ is **incorrect !!**
    - the modes of $\widetilde{\boldsymbol{U}}$ and $\widetilde{\boldsymbol{V}}$ do not follow the order of the singular values $\widetilde{\Sigma}$
    - necessity to introduce orthogonal matrices K and Q such that $\boldsymbol{S}(\widetilde{\theta}) = \widetilde{\boldsymbol{U}}K\widetilde{\Sigma}Q^T\widetilde{\boldsymbol{V}}^T$
    - K and Q are solutions to constrained optimization problems
    - analytical expression of the orthogonal matrices K and Q

M. Oulghelou and C. Allery, Non-intrusive method for parametric model order reduction using a bi-calibrated interpolation on the Grassmann manifold, *Journal of Computational Physics*, vol 426, 2021.

Problem settings

- this study focuses on the inverse problem of temperature distribution in a 2D ventilated cavity



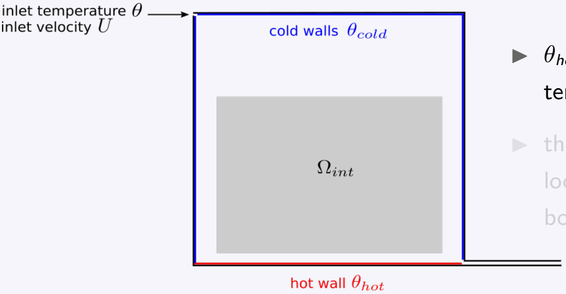
- θ_{hot} is higher than the temperature θ_{cold}
- the inlet (resp. outlet) is located at the top left (resp. bottom right) corner

M. Oulghelou, C. Beghein and C. Allery, A surrogate optimization approach for inverse problems : Application to turbulent mixed-convection flows, *Computers and Fluids*, vol 241, 2022.

- the inlet velocity U is set to a constant
- the inlet temperature θ is variable

Problem settings

- this study focuses on the inverse problem of temperature distribution in a 2D ventilated cavity



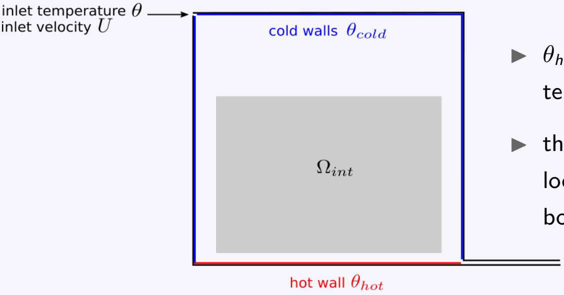
- θ_{hot} is higher than the temperature θ_{cold}
- the inlet (resp. outlet) is located at the top left (resp. bottom right) corner

M. Oulghelou, C. Beghein and C. Allery, A surrogate optimization approach for inverse problems : Application to turbulent mixed-convection flows, *Computers and Fluids*, vol 241, 2022.

- the inlet velocity U is set to a constant
- the inlet temperature θ is variable

Problem settings

- this study focuses on the inverse problem of temperature distribution in a 2D ventilated cavity



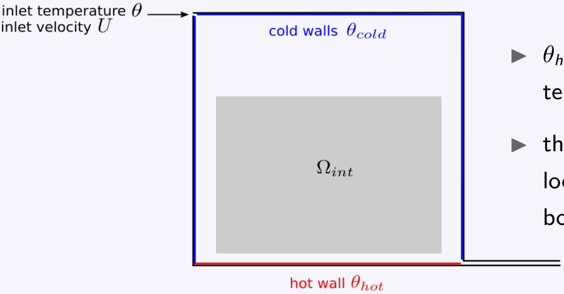
- θ_{hot} is higher than the temperature θ_{cold}
- the inlet (resp. outlet) is located at the top left (resp. bottom right) corner

M. Oulghelou, C. Beghein and C. Allery, A surrogate optimization approach for inverse problems : Application to turbulent mixed-convection flows, *Computers and Fluids*, vol 241, 2022.

- the inlet velocity U is set to a constant
- the inlet temperature θ is variable

Problem settings

- this study focuses on the inverse problem of temperature distribution in a 2D ventilated cavity



- θ_{hot} is higher than the temperature θ_{cold}
- the inlet (resp. outlet) is located at the top left (resp. bottom right) corner

M. Oulghelou, C. Beghein and C. Allery, A surrogate optimization approach for inverse problems : Application to turbulent mixed-convection flows, *Computers and Fluids*, vol 241, 2022.

- the inlet velocity U is set to a constant
- the inlet temperature θ is variable

Problem settings

- the turbulent flow is governed by Navier-Stokes equations of an incompressible Newtonian fluid with Boussinesq's assumption :

$$\begin{cases} \nabla \cdot \boldsymbol{v} = 0 \\ \rho \partial_t \boldsymbol{v} + \rho \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla p + \mu \Delta \boldsymbol{v} + \rho \, \mathbf{g} \, \beta(\Theta - \Theta_0) \boldsymbol{e}_y + \nabla \boldsymbol{\sigma_t} \\ \rho \, c_p \, \partial_t \Theta + \rho \, c_p \, \boldsymbol{v} \cdot \nabla \Theta = \lambda \, \Delta \Theta + \nabla \mathbf{q_t} \end{cases}$$

  - $\boldsymbol{v}$, Θ, p are velocity, temperature and pressure (obtained with an (URANS) turbulence model)
  - ρ, μ, ,β, C_p, λ are the density, dynamic viscosity, thermal expansion coefficient, heat capacity and heat conductivity of the fluid at the reference temperature Θ_0
  - $\mathbf{g}$ is the gravitational acceleration
  - $\boldsymbol{\sigma}_t$ and $\mathbf{q}_t$ are the turbulent Reynolds stress and the turbulent heat flux

Problem settings

- the turbulent flow is governed by Navier-Stokes equations of an incompressible Newtonian fluid with Boussinesq's assumption :

$$\begin{cases} \nabla \cdot \boldsymbol{v} = 0 \\ \rho \partial_t \boldsymbol{v} + \rho \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla p + \mu \Delta \boldsymbol{v} + \rho\, \mathbf{g}\, \beta(\Theta - \Theta_0) \boldsymbol{e}_y + \nabla \boldsymbol{\sigma_t} \\ \rho\, c_p\, \partial_t \Theta + \rho\, c_p\, \boldsymbol{v} \cdot \nabla \Theta = \lambda\, \Delta \Theta + \nabla \mathbf{q_t} \end{cases}$$

  - $\boldsymbol{v}$, Θ, p are velocity, temperature and pressure (obtained with an (URANS) turbulence model)
  - ρ, μ, ,β, C_p, λ are the density, dynamic viscosity, thermal expansion coefficient, heat capacity and heat conductivity of the fluid at the reference temperature Θ_0
  - $\mathbf{g}$ is the gravitational acceleration
  - σ_t and $\mathbf{q_t}$ are the turbulent Reynolds stress and the turbulent heat flux

Problem settings

- the turbulent flow is governed by Navier-Stokes equations of an incompressible Newtonian fluid with Boussinesq's assumption :

$$\begin{cases} \nabla \cdot \boldsymbol{v} = 0 \\ \rho \partial_t \boldsymbol{v} + \rho \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla p + \mu \Delta \boldsymbol{v} + \rho\, \mathbf{g}\, \beta(\Theta - \Theta_0) \boldsymbol{e}_y + \nabla \boldsymbol{\sigma_t} \\ \rho\, c_p\, \partial_t \Theta + \rho\, c_p\, \boldsymbol{v} \cdot \nabla \Theta = \lambda\, \Delta \Theta + \nabla \mathbf{q_t} \end{cases}$$

  - $\boldsymbol{v}$, Θ, p are velocity, temperature and pressure (obtained with an (URANS) turbulence model)
  - ρ, μ, ,β, C_p, λ are the density, dynamic viscosity, thermal expansion coefficient, heat capacity and heat conductivity of the fluid at the reference temperature Θ_0
  - $\mathbf{g}$ is the gravitational acceleration
  - $\boldsymbol{\sigma_t}$ and $\mathbf{q_t}$ are the turbulent Reynolds stress and the turbulent heat flux

Problem settings

- the turbulent flow is governed by Navier-Stokes equations of an incompressible Newtonian fluid with Boussinesq's assumption :

$$\begin{cases} \nabla \cdot \boldsymbol{v} = 0 \\ \rho \partial_t \boldsymbol{v} + \rho \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla p + \mu \Delta \boldsymbol{v} + \rho\, \mathbf{g}\, \beta(\Theta - \Theta_0) \boldsymbol{e}_y + \nabla \boldsymbol{\sigma_t} \\ \rho\, c_p\, \partial_t \Theta + \rho\, c_p\, \boldsymbol{v} \cdot \nabla \Theta = \lambda\, \Delta \Theta + \nabla \mathbf{q_t} \end{cases}$$

  - $\boldsymbol{v}$, Θ, p are velocity, temperature and pressure (obtained with an (URANS) turbulence model)
  - ρ, μ, ,β, C_p, λ are the density, dynamic viscosity, thermal expansion coefficient, heat capacity and heat conductivity of the fluid at the reference temperature Θ_0
  - $\mathbf{g}$ is the gravitational acceleration
  - $\boldsymbol{\sigma_t}$ and $\mathbf{q_t}$ are the turbulent Reynolds stress and the turbulent heat flux

Optimization problem setting

- the aim is to solve the constrained nonlinear optimization problem

$$\min_{\theta} \ \mathcal{J}(\Theta) = \int_0^{t_f} \int_{\Omega_{int}} (\Theta - \hat{\Theta})^2 \, dx \, dt \quad \text{subject to} \quad \mathcal{N}(\Theta, \boldsymbol{v}, \theta) = 0$$

  - $\mathcal{N}$ denotes the Navier-Stokes equations
  - $\boldsymbol{v}$ is the velocity field and Θ the temperature
  - the optimization variable θ is the inlet temperature
  - $\hat{\Theta}$ is a given temperature distribution
  - $[0, t_f]$ is the time frame of simulation
  - Ω_{int} is the restricted occupied zone of the spatial domain

Optimization problem setting

- the aim is to solve the constrained nonlinear optimization problem

$$\min_{\theta} \; \mathcal{J}(\Theta) = \int_0^{t_f} \int_{\Omega_{int}} (\Theta - \hat{\Theta})^2 \, dx \, dt \quad \text{subject to} \quad \mathcal{N}(\Theta, \boldsymbol{v}, \theta) = 0$$

  - $\mathcal{N}$ denotes the Navier-Stokes equations
  - $\boldsymbol{v}$ is the velocity field and Θ the temperature
  - the optimization variable θ is the inlet temperature
  - $\hat{\Theta}$ is a given temperature distribution
  - $[0, t_f]$ is the time frame of simulation
  - Ω_{int} is the restricted occupied zone of the spatial domain

Optimization problem setting

- the aim is to solve the constrained nonlinear optimization problem

$$\min_{\theta} \; \mathcal{J}(\Theta) = \int_0^{t_f} \int_{\Omega_{int}} (\Theta - \hat{\Theta})^2 \, dx \, dt \quad \text{subject to} \quad \mathcal{N}(\Theta, \boldsymbol{v}, \theta) = 0$$

  - $\mathcal{N}$ denotes the Navier-Stokes equations
  - $\boldsymbol{v}$ is the velocity field and Θ the temperature
  - the optimization variable θ is the inlet temperature
  - $\hat{\Theta}$ is a given temperature distribution
  - $[0, t_f]$ is the time frame of simulation
  - Ω_{int} is the restricted occupied zone of the spatial domain

Optimization problem setting

- the aim is to solve the constrained nonlinear optimization problem

$$\min_{\theta} \ \mathcal{J}(\Theta) = \int_0^{t_f} \int_{\Omega_{int}} (\Theta - \hat{\Theta})^2 \, dx \, dt \quad \text{subject to} \quad \mathcal{N}(\Theta, \boldsymbol{v}, \theta) = 0$$

  - $\mathcal{N}$ denotes the Navier-Stokes equations
  - $\boldsymbol{v}$ is the velocity field and Θ the temperature
  - the optimization variable θ is the inlet temperature
  - $\hat{\Theta}$ is a given temperature distribution
  - $[0, t_f]$ is the time frame of simulation
  - Ω_{int} is the restricted occupied zone of the spatial domain

Optimization problem setting

- the aim is to solve the constrained nonlinear optimization problem

$$\min_{\theta} \ \mathcal{J}(\Theta) = \int_0^{t_f} \int_{\Omega_{int}} (\Theta - \hat{\Theta})^2 \, dx \, dt \quad \text{subject to} \quad \mathcal{N}(\Theta, \boldsymbol{v}, \theta) = 0$$

  - $\mathcal{N}$ denotes the Navier-Stokes equations
  - $\boldsymbol{v}$ is the velocity field and Θ the temperature
  - the optimization variable θ is the inlet temperature
  - $\hat{\Theta}$ is a given temperature distribution
  - $[0, t_f]$ is the time frame of simulation
  - Ω_{int} is the restricted occupied zone of the spatial domain

Optimization problem setting

- the aim is to solve the constrained nonlinear optimization problem

$$\min_{\theta} \ \mathcal{J}(\Theta) = \int_0^{t_f} \int_{\Omega_{int}} (\Theta - \hat{\Theta})^2 \, dx \, dt \quad \text{subject to} \quad \mathcal{N}(\Theta, \boldsymbol{v}, \theta) = 0$$

  - $\mathcal{N}$ denotes the Navier-Stokes equations
  - $\boldsymbol{v}$ is the velocity field and Θ the temperature
  - the optimization variable θ is the inlet temperature
  - $\hat{\Theta}$ is a given temperature distribution
  - $[0, t_f]$ is the time frame of simulation
  - Ω_{int} is the restricted occupied zone of the spatial domain

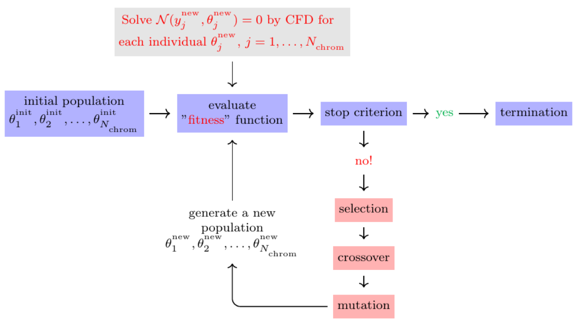
Optimization problem setting

- the aim is to solve the constrained nonlinear optimization problem

$$\min_{\theta} \; \mathcal{J}(\Theta) = \int_0^{t_f} \int_{\Omega_{int}} (\Theta - \hat{\Theta})^2 \, dx \, dt \quad \text{subject to} \quad \mathcal{N}(\Theta, \boldsymbol{v}, \theta) = 0$$

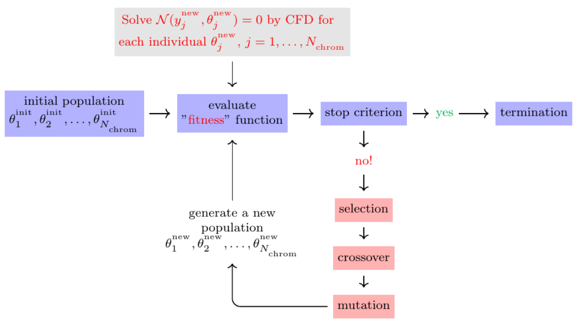
  - $\mathcal{N}$ denotes the Navier-Stokes equations
  - $\boldsymbol{v}$ is the velocity field and Θ the temperature
  - the optimization variable θ is the inlet temperature
  - $\hat{\Theta}$ is a given temperature distribution
  - $[0, t_f]$ is the time frame of simulation
  - Ω_{int} is the restricted occupied zone of the spatial domain

Standard GA approach



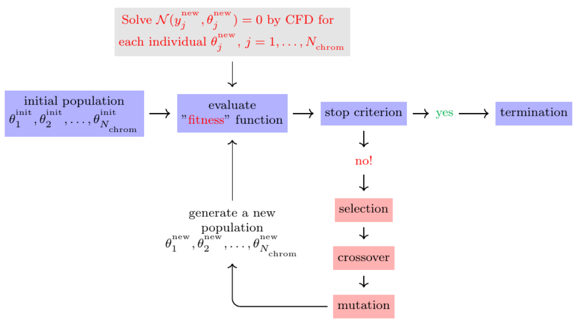
- GA consists in starting from a randomly generated set of chromosomes $\theta_1, \theta_2, \ldots, \theta_{N_{chrom}}$, forming a population
- in each population, a fitness value is assigned to each chromosome θ_j (the fitness function f is chosen as the inverse of the objective function)
- in order to evolve populations, 3 genetic operators, modeled on the Darwinian concepts of natural selection and evolution are used

Standard GA approach



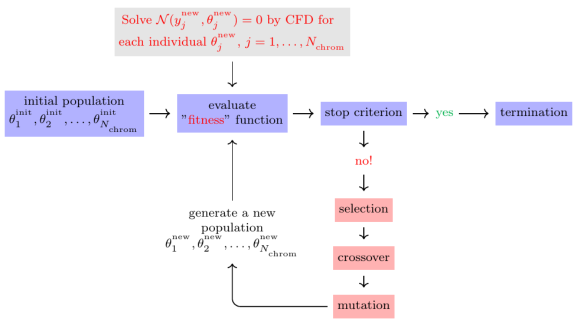
- GA consists in starting from a randomly generated set of chromosomes $\theta_1, \theta_2, \ldots, \theta_{N_{\text{chrom}}}$, forming a population
- in each population, a fitness value is assigned to each chromosome θ_j (the fitness function f is chosen as the inverse of the objective function)
- in order to evolve populations, 3 genetic operators, modeled on the Darwinian concepts of natural selection and evolution are used

Standard GA approach



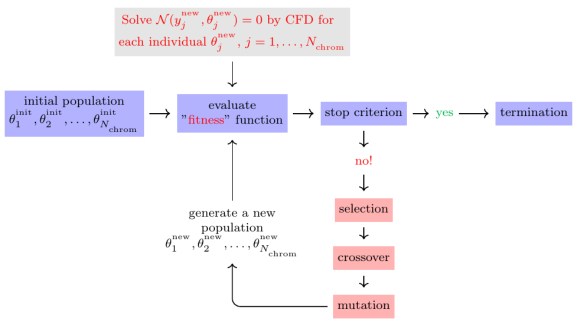
- GA consists in starting from a randomly generated set of chromosomes $\theta_1, \theta_2, \dots, \theta_{N_{chrom}}$, forming a population
- in each population, a fitness value is assigned to each chromosome θ_j (the fitness function f is chosen as the inverse of the objective function)
- in order to evolve populations, 3 genetic operators, modeled on the Darwinian concepts of natural selection and evolution are used

Standard GA approach



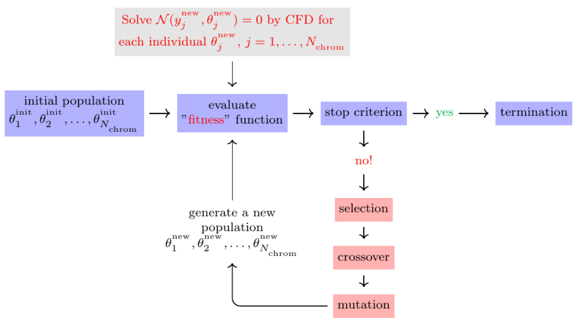
- the genetic operations are repeated for a predetermined nb of generations
- the best chromosome of the final generation is the global optimized solution

Standard GA approach



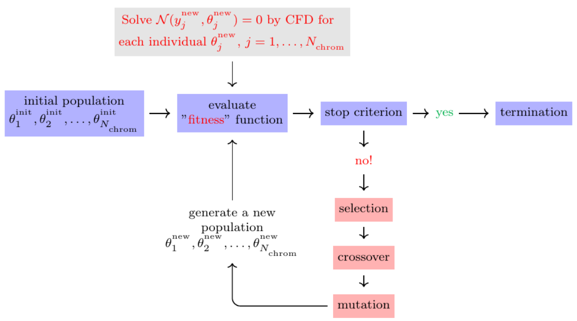
- the genetic operations are repeated for a predetermined nb of generations
- the best chromosome of the final generation is the global optimized solution

Standard GA approach



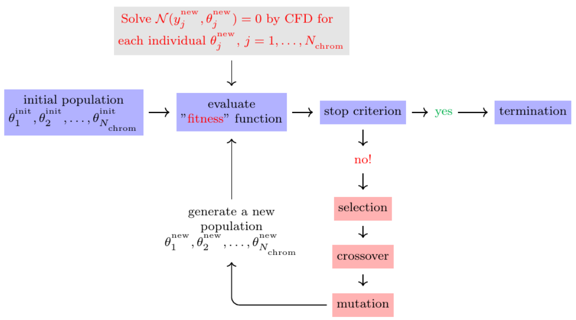
- GA needs to perform high fidelity simulations many times for each evolved population
- the time complexity of GA makes unfeasible their application in near-real time
- to tackle this issue, a reduced interpolation strategy similar to the Bi-CITSGM Method is used (barycentric interpolation on the quotient manifold, avoid the calibration phase)

Standard GA approach



- GA needs to perform high fidelity simulations many times for each evolved population
- the time complexity of GA makes unfeasible their application in near-real time
- to tackle this issue, a reduced interpolation strategy similar to the Bi-CITSGM Method is used (barycentric interpolation on the quotient manifold, avoid the calibration phase)

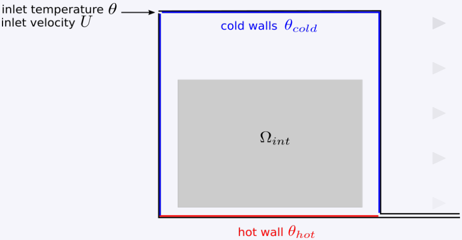
Standard GA approach



- GA needs to perform high fidelity simulations many times for each evolved population
- the time complexity of GA makes unfeasible their application in near-real time
- to tackle this issue, a reduced interpolation strategy similar to the Bi-CITSGM Method is used (barycentric interpolation on the quotient manifold, avoid the calibration phase)

Validation of the high fidelity computations

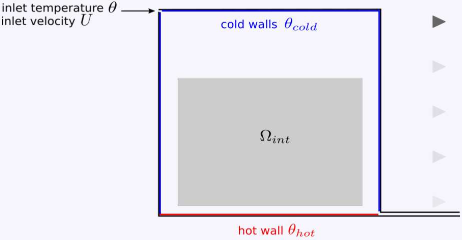
- the flow, and all input data necessary, are generate with OpenFOAM



- RNG k-epsilon model
- cavity dimensions 1.04×1.04 m
- $U_{inlet} = 0.57$ m/s and $v_{inlet} = 0$ m/s
- $\Theta_{hot} = 35.5^o C$ on the floor
- $\Theta_{cold} = 15^o C$ on the other walls
- Ra$= 2.13 \times 10^9$ and $Re = 654$
- inlet temp. varying between 2 and $26^o C$

Validation of the high fidelity computations

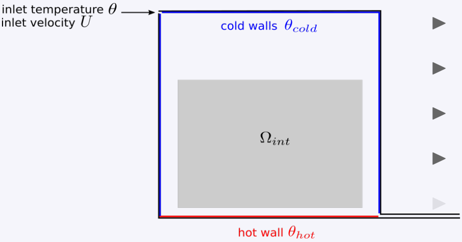
- the flow, and all input data necessary, are generate with OpenFOAM



- RNG k-epsilon model
- cavity dimensions 1.04×1.04 m
- $U_{inlet} = 0.57$ m/s and $v_{inlet} = 0$ m/s
- $\Theta_{hot} = 35.5^o C$ on the floor
- $\Theta_{cold} = 15^o C$ on the other walls
- Ra$= 2.13 \times 10^9$ and $Re = 654$
- inlet temp. varying between 2 and $26^o C$

Validation of the high fidelity computations

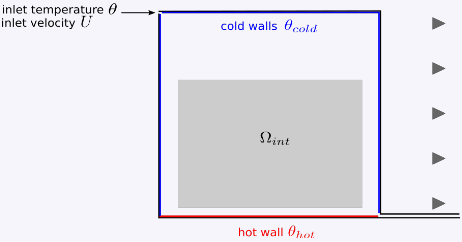
- the flow, and all input data necessary, are generate with OpenFOAM



- RNG k-epsilon model
- cavity dimensions 1.04×1.04 m
- $U_{inlet} = 0.57$ m/s and $v_{inlet} = 0$ m/s
- $\Theta_{hot} = 35.5^o C$ on the floor
- $\Theta_{cold} = 15^o C$ on the other walls
- Ra$= 2.13 \times 10^9$ and $Re = 654$
- inlet temp. varying between 2 and $26^o C$

Validation of the high fidelity computations

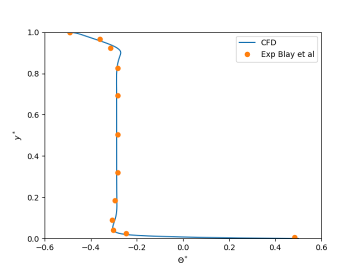
- the flow, and all input data necessary, are generate with OpenFOAM



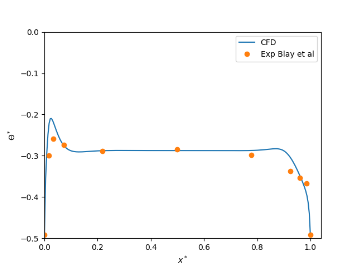
- RNG k-epsilon model
- cavity dimensions 1.04×1.04 m
- $U_{inlet} = 0.57$ m/s and $v_{inlet} = 0$ m/s
- $\Theta_{hot} = 35.5^o C$ on the floor
- $\Theta_{cold} = 15^o C$ on the other walls
- Ra$= 2.13 \times 10^9$ and $Re = 654$
- inlet temp. varying between 2 and $26^o C$

Validation of the high fidelity computations

- CFD computations were validated with the experiments of Blay et al. (for an inlet temperature equal to $15^o C$)



Θ^* at $x^* = 0.5$

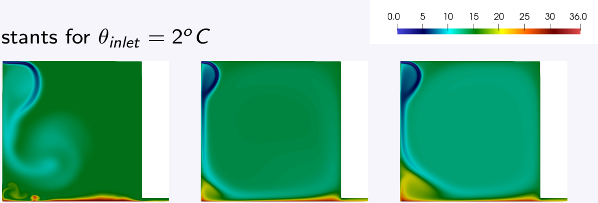


Θ^* at $y^* = 0.5$

▶ a satisfactory agreement can be noticed

Dynamics of the mixed convection flow

- three instants for $\theta_{inlet} = 2^o C$

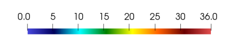


  - the air falls along the left wall, warmed by the hot floor, and finally lifted by natural convection

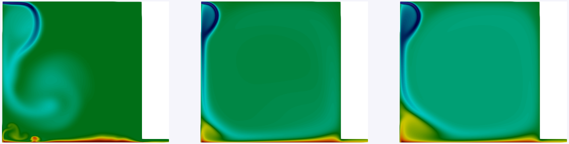
- three instants for $\theta_{inlet} = 26^o C$
  - the injected air is hot and remains in a large region along the ceiling, it falls afterwards along the left and right cold walls, and lift up along the heated floor

- the training sampling solutions is complex and rich

Dynamics of the mixed convection flow



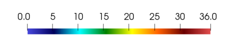
- three instants for $\theta_{inlet} = 2^o C$



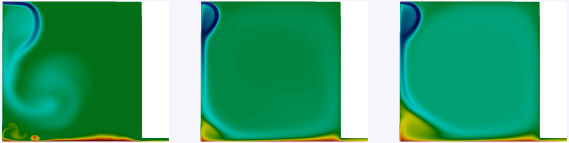
  - the air falls along the left wall, warmed by the hot floor, and finally lifted by natural convection

- three instants for $\theta_{inlet} = 26^o C$
  - the injected air is hot and remains in a large region along the ceiling, it falls afterwards along the left and right cold walls, and lift up along the heated floor
- the training sampling solutions is complex and rich

Dynamics of the mixed convection flow

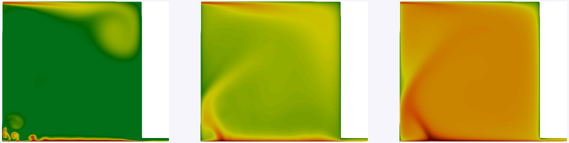


- three instants for $\theta_{inlet} = 2^o C$



  - the air falls along the left wall, warmed by the hot floor, and finally lifted by natural convection

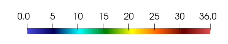
- three instants for $\theta_{inlet} = 26^o C$



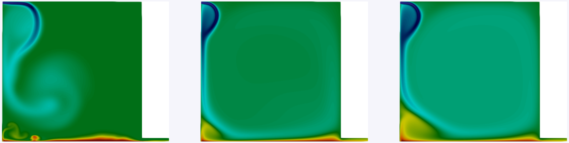
  - the injected air is hot and remains in a large region along the ceiling, it falls afterwards along the left and right cold walls, and lift up along the heated floor

- the training sampling solutions is complex and rich

Dynamics of the mixed convection flow

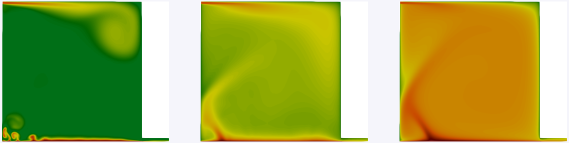


- three instants for $\theta_{inlet} = 2^o C$



  - the air falls along the left wall, warmed by the hot floor, and finally lifted by natural convection

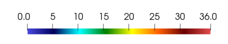
- three instants for $\theta_{inlet} = 26^o C$



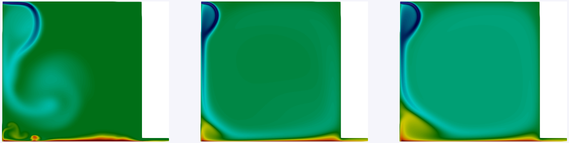
  - the injected air is hot and remains in a large region along the ceiling, it falls afterwards along the left and right cold walls, and lift up along the heated floor

- the training sampling solutions is complex and rich

Dynamics of the mixed convection flow

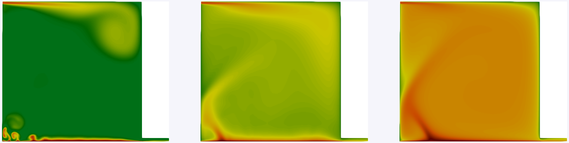


- three instants for $\theta_{inlet} = 2^o C$



  - the air falls along the left wall, warmed by the hot floor, and finally lifted by natural convection

- three instants for $\theta_{inlet} = 26^o C$



  - the injected air is hot and remains in a large region along the ceiling, it falls afterwards along the left and right cold walls, and lift up along the heated floor

- the training sampling solutions is complex and rich

Application of the surrogate GA to the mixed-convection flow pb

- the training injection temp. values belong to the set

$$I_{\text{tr}} = \{2, 5, \ldots, 23, 26^o C\}$$

- given a temp. distribution $\hat{\Theta}$, the aim is to apply the surrogate GA to approximate the associated optimal inlet temp.
  - the space of search by surrogate GA is set to

$$K = \{(\theta, ne_t, ne_x, m) \in \mathbb{R}_+ \times \mathbb{N}^3,\ 2 \leq \theta \leq 26^o C;\ 2 \leq ne_t, ne_x \leq 13;\ 4 \leq m \leq q\}$$

  - a population of 20 chromosomes formed by 4 genes randomly generated in K is used as initial guess to run the surrogate GA.
  - different tests are performed for temp. distributions associated to 16 inlet values in the set

$$I_{\text{te}} = \{3, 4, 6, 7, \ldots, 24, 25^o C\}$$

Application of the surrogate GA to the mixed-convection flow pb

- the training injection temp. values belong to the set

$$I_{\text{tr}} = \{2, 5, \ldots, 23, 26^o C\}$$

- given a temp. distribution $\hat{\Theta}$, the aim is to apply the surrogate GA to approximate the associated optimal inlet temp.
  - the space of search by surrogate GA is set to

$$K = \left\{(\theta, ne_t, ne_x, m) \in \mathbb{R}_+ \times \mathbb{N}^3,\ 2 \leq \theta \leq 26^o C;\ 2 \leq ne_t, ne_x \leq 13;\ 4 \leq m \leq q\right\}$$

  - a population of 20 chromosomes formed by 4 genes randomly generated in K is used as initial guess to run the surrogate GA.
  - different tests are performed for temp. distributions associated to 16 inlet values in the set

$$I_{\text{test}} = \{3, 4, 6, 7, \ldots, 24, 25^o C\}$$

Application of the surrogate GA to the mixed-convection flow pb

- the training injection temp. values belong to the set

$$I_{\text{tr}} = \{2, 5, \dots, 23, 26^o C\}$$

- given a temp. distribution $\hat{\Theta}$, the aim is to apply the surrogate GA to approximate the associated optimal inlet temp.
  - the space of search by surrogate GA is set to

$$K = \left\{(\theta, ne_t, ne_x, m) \in \mathbb{R}_+ \times \mathbb{N}^3,\ 2 \leq \theta \leq 26^o C;\ 2 \leq ne_t, ne_x \leq 13;\ 4 \leq m \leq q\right\}$$

  - a population of 20 chromosomes formed by 4 genes randomly generated in K is used as initial guess to run the surrogate GA.
  - different tests are performed for temp. distributions associated to 16 inlet values in the set

$$I_{\text{test}} = \{3, 4, 6, 7, \dots, 24, 25^o C\}$$

Application of the surrogate GA to the mixed-convection flow pb

- the training injection temp. values belong to the set

$$I_{\mathrm{tr}} = \{2, 5, \ldots, 23, 26^{o}C\}$$

- given a temp. distribution $\hat{\Theta}$, the aim is to apply the surrogate GA to approximate the associated optimal inlet temp.
  - the space of search by surrogate GA is set to

$$K = \left\{(\theta, ne_t, ne_x, m) \in \mathbb{R}_+ \times \mathbb{N}^3,\ 2 \leq \theta \leq 26^{o}C;\ 2 \leq ne_t, ne_x \leq 13;\ 4 \leq m \leq q\right\}$$

  - a population of 20 chromosomes formed by 4 genes randomly generated in K is used as initial guess to run the surrogate GA.
  - different tests are performed for temp. distributions associated to 16 inlet values in the set

$$I_{\mathrm{test}} = \{3, 4, 6, 7, \ldots, 24, 25^{o}C\}$$

Application of the surrogate GA to the mixed-convection flow pb

- the training injection temp. values belong to the set

$$I_{\text{tr}} = \{2, 5, \ldots, 23, 26^o C\}$$

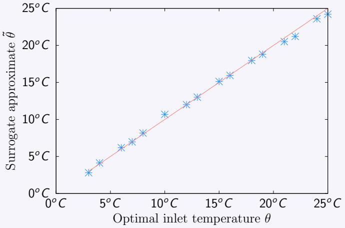
- given a temp. distribution $\hat{\Theta}$, the aim is to apply the surrogate GA to approximate the associated optimal inlet temp.
  - the space of search by surrogate GA is set to

$$K = \left\{(\theta, ne_t, ne_x, m) \in \mathbb{R}_+ \times \mathbb{N}^3,\ 2 \leq \theta \leq 26^o C;\ 2 \leq ne_t, ne_x \leq 13;\ 4 \leq m \leq q\right\}$$

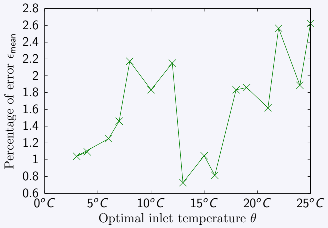
  - a population of 20 chromosomes formed by 4 genes randomly generated in K is used as initial guess to run the surrogate GA.
  - different tests are performed for temp. distributions associated to 16 inlet values in the set

$$I_{\text{test}} = \{3, 4, 6, 7, \ldots, 24, 25^o C\}$$

Application of the surrogate GA to the mixed-convection flow pb



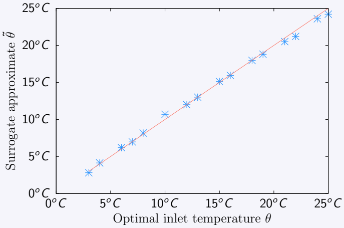
Comparison of inlet temperatures obtained by the GA



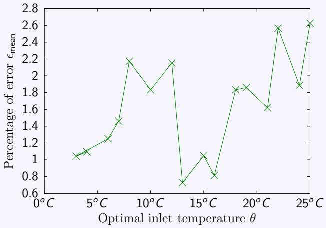
Percentage of error with respect to the CFD

- the surrogate GA succeeds to recover a good approximation $\tilde{\theta}$ of the sought inlet temperature $\hat{\theta}$
- good accuracy of the reconstructed temperature distribution of less than 3% of error
- for all the optimization tests, the optimal values of q, ne_x and ne_t are within the ranges $q \geq 7$, $ne_x \leq 4$ and $ne_t \leq 6$

Application of the surrogate GA to the mixed-convection flow pb



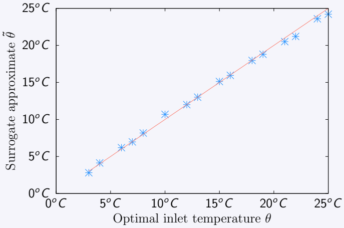
Comparison of inlet temperatures obtained by the GA



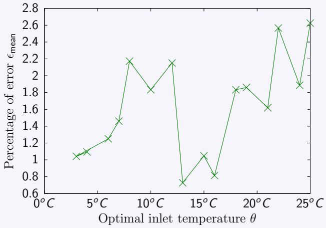
Percentage of error with respect to the CFD

- the surrogate GA succeeds to recover a good approximation $\tilde{\theta}$ of the sought inlet temperature $\hat{\theta}$
- good accuracy of the reconstructed temperature distribution of less than 3% of error
- for all the optimization tests, the optimal values of q, ne_x and ne_t are within the ranges $q \geq 7$, $ne_x \leq 4$ and $ne_t \leq 6$

Application of the surrogate GA to the mixed-convection flow pb



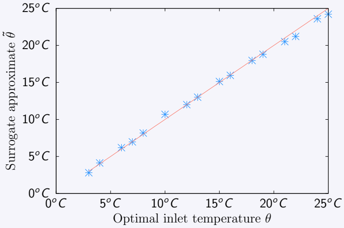
Comparison of inlet temperatures obtained by the GA



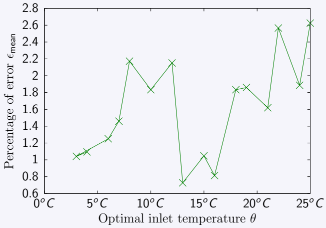
Percentage of error with respect to the CFD

- the surrogate GA succeeds to recover a good approximation $\tilde{\theta}$ of the sought inlet temperature $\hat{\theta}$
- good accuracy of the reconstructed temperature distribution of less than 3% of error
- for all the optimization tests, the optimal values of q, ne_x and ne_t are within the ranges $q \geq 7$, $ne_x \leq 4$ and $ne_t \leq 6$

Application of the surrogate GA to the mixed-convection flow pb



Comparison of inlet temperatures obtained by the GA

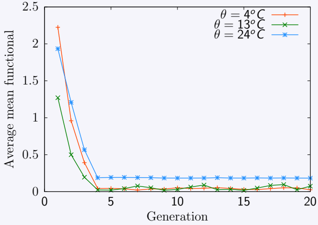


Percentage of error with respect to the CFD

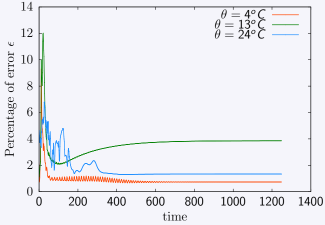
- the surrogate GA succeeds to recover a good approximation $\tilde{\theta}$ of the sought inlet temperature $\hat{\theta}$
- good accuracy of the reconstructed temperature distribution of less than 3% of error
- for all the optimization tests, the optimal values of q, ne_x and ne_t are within the ranges $q \geq 7$, $ne_x \leq 4$ and $ne_t \leq 6$

Application of the surrogate GA to the mixed-convection flow pb

- the surrogate GA predictions for three $\theta_{inlet} = 4, 13, 24^o C$ (that correspond to three different flow regimes)



Mean averaged functional by the surrogate GA

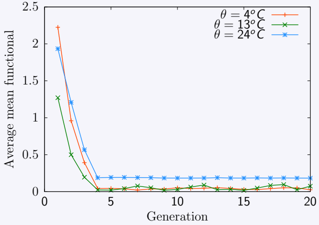


Percentage of errors over the time

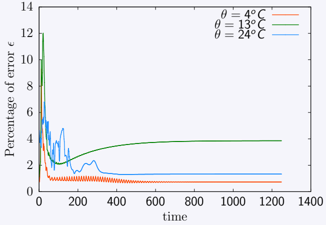
  - the cost functional has a good decay behavior
  - the recorded percentage of error at the end of the surrogate GA is nearly less than 4% almost everywhere in the time interval

Application of the surrogate GA to the mixed-convection flow pb

- the surrogate GA predictions for three $\theta_{inlet} = 4, 13, 24^o C$ (that correspond to three different flow regimes)



Mean averaged functional by the surrogate GA

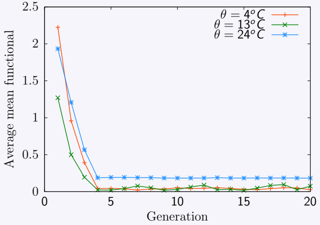


Percentage of errors over the time

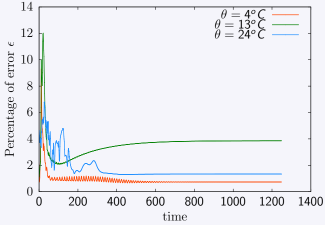
  - the cost functional has a good decay behavior
  - the recorded percentage of error at the end of the surrogate GA is nearly less than 4% almost everywhere in the time interval

Application of the surrogate GA to the mixed-convection flow pb

- the surrogate GA predictions for three $\theta_{inlet} = 4, 13, 24^o C$ (that correspond to three different flow regimes)



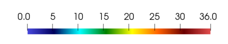
Mean averaged functional by the surrogate GA



Percentage of errors over the time

  - the cost functional has a good decay behavior
  - the recorded percentage of error at the end of the surrogate GA is nearly less than 4% almost everywhere in the time interval

Comparison of the high fidelity and surrogate GA for $\hat{\theta} = 4^o C$

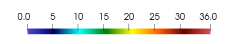


High fidelity temperature

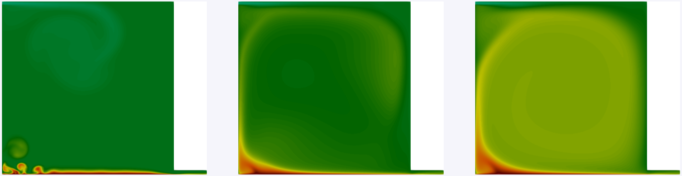
Approximate temperature by the surrogate GA

- the surrogate GA succeeded to track the provided target temp. and catch most of the dynamics features

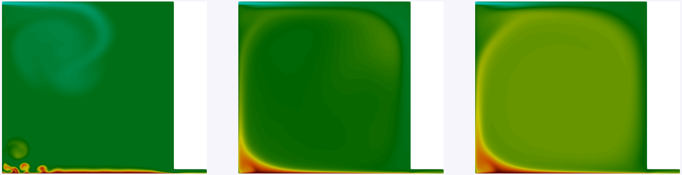
Comparison of the high fidelity and surrogate GA for $\hat{\theta} = 13^o C$



High fidelity temperature



Approximate temperature by the surrogate GA

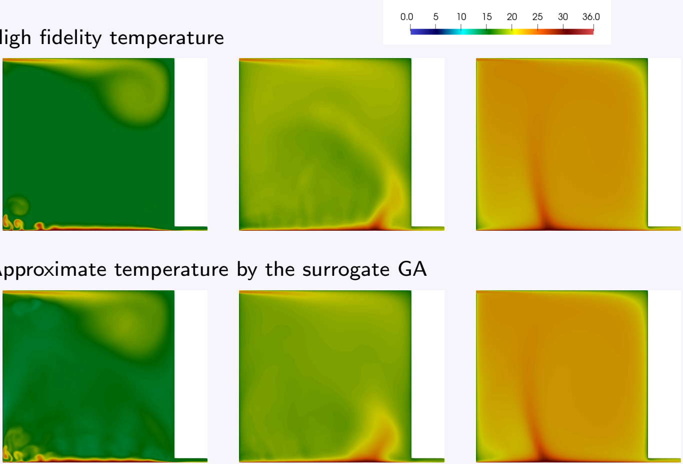


- the surrogate GA succeeded to track the provided target temp. and catch most of the dynamics features

Comparison of the high fidelity and surrogate GA for $\hat{\theta} = 24^o C$

High fidelity temperature

Approximate temperature by the surrogate GA



- the surrogate GA succeeded to track the provided target temp. and catch most of the dynamics features