Sommaire

Sommaire **1**
1 La PGD 2
1.1 Généralités 2
1.2 Mise en place de nouvelles variables : nouvelles coordonnées 2
1.3 Équations avec paramètre 3
2 La méthode LATIN 4
2.1 Notations 5
2.2 Stratégie de résolution LATIN avec PGD 5
3 La PGD en dynamique 9
3.1 Définition du problème 9
3.2 Problème en espace 11
3.3 Problème en temps 12
3.4 Problème en paramètre 12

Bibliographie **14**

Bibliographie **14**

1 La PGD

1.1 Généralités

La PGD, pour "Proper Generalized Decomposition", trouve son origine dans [1] sous le nom de "radial time-space approximation", comme faisant partie la méthode LATIN (voir également [2]).

Parmi les possibilités apportées par la réduction de modèles, elle peut permettre de résoudre des problèmes jusque là inenvisageables, par exemple dans la chimie quantique (quand des processus chimiques impliquent un nombre de molécules de réactifs si petit que le concept de continuité de concentration n'est plus valide, par exemple dans la recherche génétique). De tels problèmes peuvent rapidement être impossibles à résoudre par des méthodes standards, car ils souffrent de ce que l'on appelle la malédiction de la dimension. La solution est inaccessible par des approches traditionnelles de discrétisation utilisant un maillage car elles présentent une taille du système à résoudre qui croit exponentiellement en fonction du nombre de particules. Plus de détails sur ce sujet sont donnés dans [3].

La diminution drastique de temps de calcul apportée par la PGD permet d'envisager de nouvelles utilisations de la simulation, notamment dans les DDDAS (pour Dynamic Data-Driven Application System [4]), dont l'enjeu est de permettre à la simulation d'interagir avec une manipulation en temps réel. C'est-à-dire que les instruments actionneurs seront influencés par les résultats de calculs apportés par la simulation et les capteurs du système renverront des données à la simulation pour réajuster les calculs.

La PDG rencontre donc un franc succès depuis quelques années et ses applications sont pléthores. Un aperçu des différents travaux concernant cette méthode peut être trouvé dans [5].

Parmi les utilisations de la PGD on peut citer, l'application aux plaques dans [6]. La PGD est aussi utilisée dans le cadre de la dynamique dans [7]. La méthode est étudiée également dans le cadre des problèmes inverses par [8, 9]

1.2 Mise en place de nouvelles variables : nouvelles coordonnées

La PGD présente un avantage, par la mise en place de coordonnées supplémentaires, comme le décrivent [3] et [4]. Il s'agit de rajouter au produit de fonctions à variables séparées (à deux variables comme celui de la POD [10]), une fonction d'une nouvelle variable. Ceci permet de conserver une variable comme une inconnue et ainsi d'avoir un problème résolu quelque soit la valeur de cette variable. Pour obtenir un tel résultat sans cette méthode il faudrait

effectuer de nombreux tirages de valeurs pour la variable et réaliser une approche de Monte Carlo, ce qui requerrait autant de résolutions. L'ajout de cette coordonnée permet donc de passer outre la nécessité de résoudre le problème pour chaque valeur d'une variable, et permet de résoudre une famille de problèmes qui dépendent de cette variable. C'est le principe de base de l'analyse stochastique, qui fait l'objet d'un intérêt grandissant. De cette manière la résolution du problème par la PGD apporte plus encore qu'une résolution directe, et peut permettre facilement la mise en place d'une optimisation. Ceci ne se limite pas à une valeur matériau et peut s'appliquer à une condition initiale ou à une valeur géométrique. Dans le cas de la valeur géométrique cela complexifie le problème et fait l'objet d'études en cours comme [11, 12].

1.3 Équations avec paramètre

Comme la POD, la PDG approxime la solution à l'aide de l'hypothèse de séparation des variables comme une somme de produits de fonctions ; prenons la fonction solution $z(x,t)$, elle sera approximée sous la forme :

$$z(x,t) \approx \sum_{m=1}^{m_{max}} \varphi_m(x) g_m(t) h_m(\theta) \tag{1}$$

Mais à la différence de la POD, la fonction $z(x,t)$ est inconnue au préalable, on ne connaît alors pas les modes et on les calcule "à la volée".

On considère le problème :

$$B(u,v) = L(v) \quad \forall v \in H \tag{2}$$

Sous les hypothèses du théorème de Lax-Milgram :

- H un espace de Hilbert
- B un forme bilinéaire, continue et coercive sur H
- L une forme linéaire continue sur H

il existe un unique u ∈ H solution de l'équation 2.

On applique alors la démarche classique pour la PGD, appelée Galerkin progressive. On suppose z_{m-1} (l'approximation de z à l'ordre $m-1$) connue. Un nouveau triplet $(\varphi, g, h) \in U \times V \times W$ est défini pour la décomposition d'ordre m comme celui qui vérifie le critère de double orthogonalité de Galerkin suivant :

$$\begin{gathered} B(z_{m-1} + \varphi g h, \varphi^* g h + \varphi g^* h + \varphi g h^*) = L(\varphi^* g h + \varphi g^* h + \varphi g h^*) \\ \forall (\varphi, g, h) \in U \times V \times W \end{gathered} \tag{3}$$

Alors $S_m^{\varphi}(g,h)$ est définie par :

$$B(z_{m-1} + \varphi g h, \varphi^* g h) = L(\varphi^* g h) \quad \forall \varphi^* \in U \tag{4}$$

et $S_m^g(\varphi, h)$ par :

$$B(z_{m-1} + \varphi g h, \varphi g^* h) = L(\varphi g^* h) \quad \forall g^* \in V \tag{5}$$

Le triplet (φ, g, h) vérifie l'équation 3 si et seulement si $\varphi = S_m^\varphi(g, h)$, $g = S_m^g(\varphi, h)$ et $h = S_m^h(\varphi, g)$, qui est un problème non-linéaire. La résolution de celui-ci par la PGD peut être vue comme un problème aux valeurs propres. Ceci permet, en s'inspirant d'algorithmes utilisés dans les problèmes aux valeurs propres, d'obtenir l'algorithme 1, qui donne la décomposition en n-uplet. Cet algorithme peut être modifié, notamment par l'ajout de l'orthogonalisation qui peut généralement améliorer les résultats (dans le cas ou il n'y a que deux variables). L'orthogonalisation au delà de 2 variables séparées devient complexe et nécessite des travaux complémentaires..

Algorithme 1 Résolution PGD

```
for m = 1 à m_max do
    initialiser a_1
    for k = 1 à k_max do
        φ_k ← S_m^φ(g_k, h_k)
        normer φ_k (‖φ_k‖_Ω = 1)
        g_k ← S_m^g(φ_k, h_k)
        normer g_k (‖g_k‖_T = 1)
        h_k ← S_m^h(φ_k, g_k)
        vérifier la convergence de (φ_k.g_k.g_k)
    end for
    φ_m ← φ_k, g_m ← g_k et h_m ← h_k
    z_m = z_{m-1} + φ_m g_m h_m
end for
```

La méthode pour trouver les n-uplet de fonctions présentée ici est appelée Galerkin progressive, elle utilise un point fixe (boucle sur k dans l'algorithme 1). Mais il existe d'autres méthodes, comme la minimisation du résidu. Une comparaison des méthodes est présentée dans [13], on peut noter que la minimisation apporte l'assurance de la convergence mais est plus couteuse en ressources de calcul.

—-[14]—-

2 La méthode LATIN

La méthode LATIN (Large Time INcrement method) [1, 15] a été initialement proposée pour traiter des problèmes non linéaires dépendants du temps.

Depuis son introduction, cette méthode a été exploitée autour de nombreux grands axes de développements :

- Grandes transformations, formulation corotationnelle (PGD), voir par exemple [16]
- Elasto-visco-plasticité, grand nombre de cycles, thermo-élasto-visco-plasticité (PGD) voir [17, 18]
- Endommagement et composites, voir [19, 20, 21]
- Assemblages, voir [22, 23]
- Décomposition de domaine au sens large, voir [24, 25, 26, 27, 28]
- Identification - Problèmes inverses, voir [29, 30]
- Multiparamétrique sans PGD, voir [31]
- Multiparamétrique avec PGD, voir [32]

2.1 Notations

Par simplicité est présenté ici un problème de thermique, avec T la température et $\underline{Y}$ le flux. Pour simplifier l'écriture du problème, on introduit les espaces suivants (où $\bullet^*$ désigne les espaces homogènes associés) :

- L'espace $\mathscr{T}$ des champs T admissibles est comme précédemment :

$$\mathscr{T} = \{T \mid \quad T|_{\partial_T\Omega} = T_d \quad \text{et} \quad T|_{(t=0)} = T_{init}\} \tag{6}$$

- L'espace $\mathbf{A_d}$ des champs solution $(T, \underline{Y})$ admissibles :

$$\mathbf{A_d} = \{(T, \underline{Y}) \mid \quad T \in \mathscr{T} \quad \text{et} \quad \operatorname{div}\underline{Y} + r = \rho c \frac{\partial T}{\partial t} \quad \text{avec} \quad \underline{Y} \cdot \underline{n}|_{\partial_Y\Omega} = y_d\} \tag{7}$$

- L'espace $\mathbf{\Gamma}$ des champs solution $(T, \underline{Y})$ vérifiant la loi de Fourier :

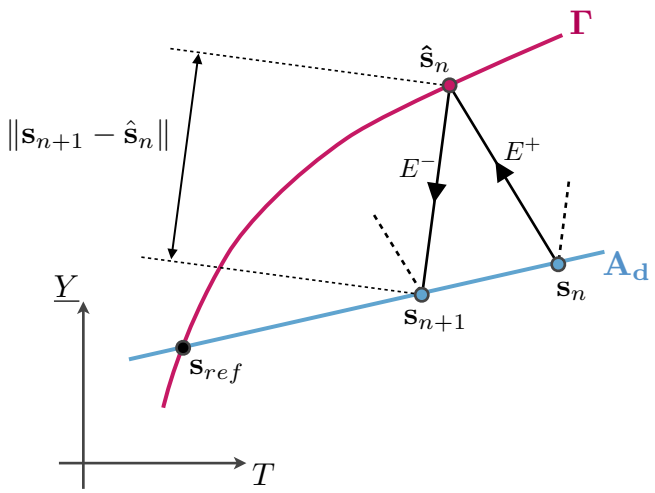
$$\mathbf{\Gamma} = \left\{(T, \underline{Y}) \mid \underline{Y} = \lambda \, \underline{\operatorname{grad}}\, T\right\} \tag{8}$$

2.2 Stratégie de résolution LATIN avec PGD

On peut schématiser la méthode de résolution LATIN comme étant une stratégie qui génère une approximation de la solution **s** du problème sur l'ensemble de l'espace et du temps et en améliore automatiquement la qualité à chaque itération, quitte à commencer par une approximation très grossière. En ceci, la méthode est dite non incrémentale. Cette méthode repose sur 3 principes de base ; la séparation des difficultés, l'approche itérative à 2 étapes et la représentation adaptée des inconnues.

La séparation des difficultés Ce point consiste à traiter séparément les difficultés du problème en partitionnant l'espace constitué par l'ensemble des équations que le problème doit vérifier en 2 espaces distincts $\mathbf{\Gamma}$ et $\mathbf{A_d}$ comme définis en partie 2.1. L'espace $\mathbf{\Gamma}$ est composé des champs vérifiant les équations locales, éventuellement non linéaires (par exemple, pour le problème de diffusion, si la conductivité λ dépend de la température T), et l'espace $\mathbf{A_d}$ rassemble les champs vérifiant les équations linéaires éventuellement globales. La solution du problème $\mathbf{s}_{ref} = (T, \underline{Y})$ vérifie l'ensemble des équations et se trouve donc à l'intersection de ces 2 espaces : $\mathbf{s}_{ref} = \mathbf{A_d} \cap \mathbf{\Gamma}$. (figure 1)

L'approche itérative à 2 étapes Pour trouver l'intersection de ces 2 espaces, on cherche alternativement une solution de l'un puis l'autre des espaces $\mathbf{\Gamma}$ et $\mathbf{A_d}$: la stratégie est donc fondée sur une stratégie itérative à 2 étapes.



$$\cdots \longrightarrow \mathbf{s}_n \in \mathbf{A_d} \underbrace{\xrightarrow{\text{étape locale}} \hat{\mathbf{s}}_n \in \mathbf{\Gamma} \xrightarrow{\text{étape linéaire}} \mathbf{s}_{n+1} \in \mathbf{A_d}}_{\text{iteration } n+1} \longrightarrow \hat{\mathbf{s}}_{n+1} \longrightarrow \cdots \tag{9}$$

FIGURE 1: Une itération à l'ordre $n+1$ de la méthode LATIN

Il apparait sur le schéma de la figure 1 que l'on doit introduire des « directions de recherche » $\mathbf{E}^+$ et $\mathbf{E}^-$. En effet, le problème complet étant divisé en deux sous-problèmes complémentaires, chacun de ces sous-problèmes manque d'information. Les sous-problèmes n'ont donc pas d'unicité de la solution ou deviennent mal posés et on vient rajouter une équation par cette direction de re-

cherche pour trouver une solution et qu'elle soit unique.
On se donne les directions de recherche $\mathbf{E}^+$ pour l'étape locale et $\mathbf{E}^-$ pour l'étape linéaire (où h est un paramètre de la méthode) :

$$\mathbf{E}^+ : \quad h\,(\underline{\hat{Y}}_n - \underline{Y}_n) + (\underline{\text{grad}}\,\hat{T}_n - \underline{\text{grad}}\,T_n) = 0 \tag{10}$$

$$\mathbf{E}^- : \quad h\,(\underline{Y}_{n+1} - \underline{\hat{Y}}_n) - (\underline{\text{grad}}\,T_{n+1} - \underline{\text{grad}}\,\hat{T}_n) = 0 \tag{11}$$

Comme schématisé sur la figure 1, l'itération $n+1$ est donc composée :

- d'une étape locale, qui, partant d'une solution connue $\mathbf{s}_n = (T_n, \underline{Y}_n) \in \mathbf{A_d}$ cherche une solution $\hat{\mathbf{s}}_n = (\hat{T}_n, \underline{\hat{Y}}_n) \in \mathbf{\Gamma}$ en suivant la direction de recherche E^+,

- d'une étape linéaire, qui, partant d'une solution connue $\hat{\mathbf{s}}_n = (\hat{T}_n, \underline{\hat{Y}}_n) \in \mathbf{\Gamma}$ cherche une solution $\mathbf{s}_{n+1} = (T_{n+1}, \underline{Y}_{n+1}) \in \mathbf{A_d}$ en suivant la direction de recherche E^-.

Réduction de modèles L'utilisation de la méthode LATIN autour des 2 seuls premiers points suffit à trouver la solution du problème. Cependant la convergence de la méthode est coûteuse avec cette architecture car la solution doit être améliorée à chaque itération sur l'ensemble de l'espace et du temps. En pratique, l'étape locale conduit à la résolution en chaque point d'espace d'un système différentiel en temps (généralement de petite taille) ce qui engendre un coût de calcul modique. En revanche, l'étape linéaire consiste à résoudre un problème global en espace pour chaque pas de temps.
La manipulation de la solution sur l'ensemble de l'espace et du temps est donc *a priori* un inconvénient majeur pour le temps de calcul de la stratégie. Cependant, la connaissance de la solution sur $\mathscr{I} \times \Omega$ permet de mettre en œuvre les notions d'approximations spatio-temporelles qui vont en définitive permettre une grande réduction du temps de calcul. C'est donc avec ce troisième aspect que la méthode LATIN prend tout son sens.

La PGD permet de calculer l'approximation à variables séparées de la solution sans avoir besoin de réalisations. Elle doit donc être associée à une stratégie permettant de générer à la fois les fonctions du temps et les fonctions spatiales les plus pertinentes pour la solution. De plus, cette stratégie ne peut être incrémentale car chaque couple généré durant le processus doit améliorer

l'approximation de la solution sur $\mathscr{I} \times \Omega$, ce qui est donc compatible avec la méthode LATIN.

Étant donné que le coût de l'étape locale est déjà réduit, on estime que l'introduction de la PGD au cours de cette étape ne constituera pas un réel gain et on choisit d'utiliser la représentation PGD uniquement durant l'étape linéaire.

L'ajout de la contrainte de représentation sous forme PGD des inconnues rend le problème, déjà constitué des 2 premiers points, sur-contraint. On ne pourra donc pas vérifier l'ensemble des équations. Le but étant ici de réduire le temps de calcul par l'utilisation de la PGD, on impose la représentation PGD des inconnues, et on ne peut ainsi jouer que sur les 2 premiers points de la méthode. Il apparait donc que 2 versions de la méthode LATIN avec PGD sont envisageables :

- Soit en vérifiant « au mieux » la direction de recherche. La solution d'une étape s'obtiendra par la résolution d'un problème de minimisation sur l'écart entre la direction de recherche empruntée et celle voulue. On vérifiera donc exactement l'admissibilité de la solution et la représentation PGD. L'idée de cette version peut être schématisée par la figure 2.

- Soit en vérifiant « au mieux » l'admissibilité de la solution à l'aide d'une approche Galerkin. On vérifiera donc exactement la direction de recherche et la représentation PGD. L'idée de cette version peut être schématisée par la figure 3.

Pour des raisons de simplicité de mise en œuvre, on choisit de se placer dans le cadre d'une approche Galerkin. Ainsi, on vérifiera exactement la direction de recherche de la méthode ainsi que la représentation PGD de la solution, alors que la solution à chaque itération ne sera admissible qu'au mieux. Cette technique peut parfois mal converger mais ce n'est pas le cas pour le problème ici étudié [28].
Etant donné que l'on vérifie exactement la direction de recherche (11), on peut toujours directement lier $\underline{Y}$ à T. On peut donc toujours substituer la variable $\underline{Y}$ par une expression ne dépendant que de T. La seule recherche de T sous forme PGD est donc suffisante.
Durant l'itération $n+1$, on cherche à améliorer la solution $T_n \in \mathscr{T}$ à l'aide d'un terme correctif $\Delta T_{n+1} \in \mathscr{T}^*$. C'est cette correction qui est cherchée sous une forme à variables séparées.

$$T_{n+1} = T_n + \Delta T_{n+1} \quad \text{où} \quad \Delta T_{n+1} = \tau_{n+1}(t)\, \mathbb{T}_{n+1}(\underline{M}) \tag{12}$$

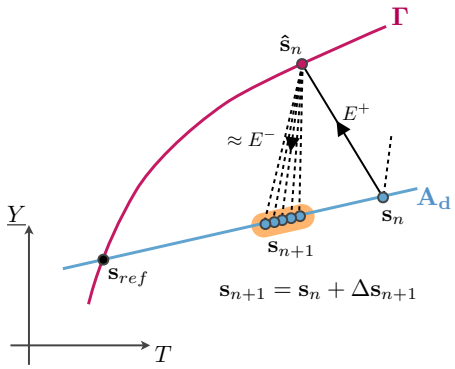


FIGURE 2: Vérification au mieux de la direction de recherche

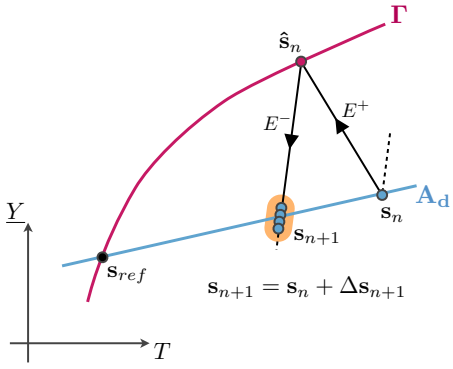


FIGURE 3: Admissibilité au mieux de $\mathbf{s}_{n+1}$

3 La PGD en dynamique

Dans le cadre du projet MECASIF, nous cherchons à appliquer la PGD à un problème de dynamique, voici le développement des équations.

3.1 Définition du problème

- Admissibilité cinématique :
  $U \in [H^1(\Omega)]^3, U_{|\partial\Omega_U} = U_d$
- Admissibilité statique :
  $div\sigma + f_v = \rho\ddot{U} + \mu\dot{U}$ sur Ω et $\sigma.n_{|\partial\Omega_f} = f_d$
- Relation de comportement :
  $\sigma = H : \varepsilon(U)$ sur Ω
- Conditions initiales :
  $U(0) = U_0$ et $\dot{U}(0) = \dot{U}_0$

Formulation variationnelle :

$$\forall U^* \text{ CA0}, \quad \int_\Omega [\varepsilon(U^*) : \sigma + U^*\mu\dot{U} + U^*\rho\ddot{U} - U^* f_d] d\Omega \\ - \int_{\partial\Omega_f} U^* f_d ds - \underbrace{\int_{\partial\Omega_U} U^*\sigma.nds}_{=0} = 0 \tag{13}$$

Où

- U est le déplacement
- Ω l'espace, avec le bord $\partial\Omega = \partial\Omega_U \cap \partial\Omega_f$
- σ le tenseur des contraintes

- ε le tenseur des déformations
- ρ la masse volumique
- μ l'amortissement volumique (mal connue)
- H le tenseur de Hooke
- f_v l'effort volumique
- f_d l'effort imposé sur $\partial\Omega_f$

l'utilisation de l'hypothèse des variables séparables donne une expression de la forme :

$$U(x,t,\theta) = \sum_{k=1}^{n} \varphi_k(x) g_k(t) h_k(\theta) \tag{14}$$

Où

- x est la variable d'espace
- t la variable de temps
- θ une variable paramètre (par exemple un module d'Young, un amortissement...)

Il faut discrétiser les fonctions sur leurs domaines de définition respectif. Ici la discrétisation en espace donne :

$$\varphi(x) \quad = \quad \sum_{i=1}^{Nbc_x} N_{\varphi_i}(x)\varphi_i \quad = \quad N_\varphi(x)\boldsymbol{\varphi_q} \tag{15}$$

Nbc_x représente le nombre de composantes de discrétisation spatiale, l'indice q représente le vecteur sur le domaine discrétisé.

On remplace dans la formulation variationnelle U^* par

$$(\varphi g h)^* = (\varphi^* g h + \varphi g^* h + \varphi g h^*)$$

On rassemble les termes en $N_\varphi(x)$ ce qui permet de définir :

$$\mathbf{K} = \int_\Omega \varepsilon(N_\varphi(x)) : H : \varepsilon(N_\varphi(x)) \ \ dx \tag{16}$$

$$\mathbf{C} = \int_\Omega N_\varphi(x)^T \mu N_\varphi(x) \ \ dx \tag{17}$$

$$\mathbf{M} = \int_\Omega N_\varphi(x)^T \rho N_\varphi(x) \ \ dx \tag{18}$$

$$\mathbf{f} = \int_\Omega N_\varphi(x)^T f_v \ \ dx + \int_{\partial\Omega_f} N_\varphi(x)^T f_d \ dsdtd\theta \tag{19}$$

Tous calculs fait cela donne :

$$\int_T\int_\Theta \left(\varphi_q^* gh + \varphi_q g^* h + \varphi_q g h^*\right)^T \quad \Big[\mathbf{K}\,\varphi_q gh + \mathbf{C}\,\varphi_q \dot{g}h + \mathbf{M}\,\varphi_q \ddot{g}h \\ + \mathbf{K}\sum_{k=1}^{n-1}(\varphi_q)_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\varphi_q)_k \dot{g}_k h_k \\ + \mathbf{M}\sum_{k=1}^{n-1}(\varphi_q)_k \ddot{g}_k h_k - \mathbf{f}\Big]\, dt d\theta = 0 \tag{20}$$

Ce qui permet d'obtenir un problème à résoudre pour chaque variable, comme indiqué en 1.3

3.2 Problème en espace

On annule les termes en g^* et h^*, pour obtenir :

$$\forall \varphi_q^* \quad \int_T\int_\Theta \varphi_q^{*T} gh \quad [\,\mathbf{K}\,\varphi_q gh + \mathbf{C}\,\varphi_q \dot{g}h + \mathbf{M}\,\varphi_q \ddot{g}h \\ + \mathbf{K}\textstyle\sum_{k=1}^{n-1}(\varphi_q)_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\varphi_q)_k \dot{g}_k h_k \\ + \mathbf{M}\textstyle\sum_{k=1}^{n-1}(\varphi_q)_k \ddot{g}_k h_k - \mathbf{f}\,]\, dt d\theta \;\; = 0 \tag{21}$$

Donc :

$$\int_T\int_\Theta gh \quad [\,\mathbf{K}\,\varphi_q gh + \mathbf{C}\,\varphi_q \dot{g}h + \mathbf{M}\,\varphi_q \ddot{g}h \\ + \mathbf{K}\textstyle\sum_{k=1}^{n-1}(\varphi_q)_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\varphi_q)_k \dot{g}_k h_k \\ + \mathbf{M}\textstyle\sum_{k=1}^{n-1}(\varphi_q)_k \ddot{g}_k h_k - \mathbf{f}\,]\, dt d\theta \;\; = 0 \tag{22}$$

Soit :

$$-\int_T\int_\Theta gh[\,\mathbf{K}\,gh + \mathbf{C}\,\dot{g}h + \mathbf{M}\,\ddot{g}h]\, dt d\theta\, \varphi_q \;\; = \\ \int_T\int_\Theta gh[\mathbf{K}\sum_{k=1}^{n-1}(\varphi_q)_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\varphi_q)_k \dot{g}_k h_k + \mathbf{M}\sum_{k=1}^{n-1}(\varphi_q)_k \ddot{g}_k h_k - \mathbf{f}]\, dt d\theta \tag{23}$$

Ceci est un système auquel il faut rajouter les conditions limites en déplacement U_d par une méthode adaptée : substitution, multiplicateurs de Lagrange... Dans le deuxième cas on change :

$$\mathbf{M} \leftarrow \begin{bmatrix} \mathbf{M} & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{C} \leftarrow \begin{bmatrix} \mathbf{C} & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{K} \leftarrow \begin{bmatrix} \mathbf{K} & \mathbf{D}^T \\ \mathbf{D} & 0 \end{bmatrix}, \varphi_q \leftarrow \begin{bmatrix} \varphi_q \\ \lambda \end{bmatrix} \text{ et } \mathbf{f} \leftarrow \begin{bmatrix} \mathbf{f} \\ \mathbf{b} \end{bmatrix} \tag{24}$$

où **D** et **b** représentent les déplacement imposés de la manière suivante :

$$\mathbf{D}\varphi_q = \mathbf{b}$$

3.3 Problème en temps

Partant de l'équation 20 on annule les termes en $\boldsymbol{\varphi_q^*}$ et h^*, pour obtenir :

$$\begin{array}{lll} \forall g^* & \int_T\int_\Theta \boldsymbol{\varphi_q}^T g^* h & [\,\mathbf{K}\,\boldsymbol{\varphi_q} g h + \mathbf{C}\,\boldsymbol{\varphi_q}\dot{g} h + \mathbf{M}\,\boldsymbol{\varphi_q}\ddot{g} h \\ & & +\mathbf{K}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \dot{g}_k h_k \\ & & +\mathbf{M}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \ddot{g}_k h_k - \mathbf{f}\,]\; dt d\theta \;\; = 0 \end{array} \tag{25}$$

Donc :

$$\begin{array}{lll} & \int_\Theta \boldsymbol{\varphi_q}^T h & [\,\mathbf{K}\,\boldsymbol{\varphi_q} g h + \mathbf{C}\,\boldsymbol{\varphi_q}\dot{g} h + \mathbf{M}\,\boldsymbol{\varphi_q}\ddot{g} h \\ & & +\mathbf{K}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \dot{g}_k h_k \\ & & +\mathbf{M}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \ddot{g}_k h_k - \mathbf{f}\,]\; d\theta \;\; = 0 \end{array} \tag{26}$$

Soit :

$$\begin{array}{l} \displaystyle\int_\Theta \boldsymbol{\varphi_q}^T h[\,\mathbf{K}\,\boldsymbol{\varphi_q} h]\; d\theta\; g + \int_\Theta \boldsymbol{\varphi_q}^T h[\,\mathbf{C}\,\boldsymbol{\varphi_q} h]\; d\theta\; \dot{g} + \int_\Theta \boldsymbol{\varphi_q}^T h[\,\mathbf{M}\,\boldsymbol{\varphi_q} h]\; d\theta\; \ddot{g} \;\; = \\ \displaystyle -\int_\Theta \boldsymbol{\varphi_q}^T h[\mathbf{K}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \dot{g}_k h_k + \mathbf{M}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \ddot{g}_k h_k - \mathbf{f}]\; d\theta \end{array} \tag{27}$$

On se trouve donc en présence d'une équation différentielle en g, que l'on peut résoudre comme on résout classiquement un problème de dynamique, i.e. en utilisant un schéma d'intégration ou en utilisant des élément finis temporels.

3.4 Problème en paramètre

Partant de l'équation 20 on annule les termes en $\boldsymbol{\varphi_q^*}$ et g^*, pour obtenir :

$$\begin{array}{lll} \forall h^* & \int_T\int_\Theta \boldsymbol{\varphi_q}^T g h^* & [\,\mathbf{K}\,\boldsymbol{\varphi_q} g h + \mathbf{C}\,\boldsymbol{\varphi_q}\dot{g} h + \mathbf{M}\,\boldsymbol{\varphi_q}\ddot{g} h \\ & & +\mathbf{K}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \dot{g}_k h_k \\ & & +\mathbf{M}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \ddot{g}_k h_k - \mathbf{f}\,]\; dt d\theta \;\; = 0 \end{array} \tag{28}$$

On sort h (indépendant de t) de l'integrale sur T :

$$\begin{array}{lll} \forall h^* & \int_\Theta h^* \int_T g \boldsymbol{\varphi_q}^T & \left[\,\mathbf{K}\,\boldsymbol{\varphi_q} g + \mathbf{C}\,\boldsymbol{\varphi_q}\dot{g} + \mathbf{M}\,\boldsymbol{\varphi_q}\ddot{g}\right]\; dt\; h\; d\theta \\ = & \int_\Theta h^* \int_T g \boldsymbol{\varphi_q}^T & -[\,\mathbf{K}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k g_k h_k + \mathbf{C}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \dot{g}_k h_k \\ & & +\mathbf{M}\sum_{k=1}^{n-1}(\boldsymbol{\varphi_q})_k \ddot{g}_k h_k - \mathbf{f}\,]\; dt d\theta \end{array} \tag{29}$$

Contrairement au problème précédant (en temps) on a une dépendance des termes intégrés par rapport à la variable θ. Pour continuer on fait donc apparaitre la forme discrétisée :

$$h(\theta) \;\; = \;\; \sum_{i=1}^{Nbc_\theta} N_{hi}(\theta) h_i \;\; = \;\; N_h(\theta)\boldsymbol{h_q} \tag{30}$$

Ce qui donne :

$$\begin{array}{lll}
\forall \boldsymbol{h}_{\boldsymbol{q}}^{*} & \int_{\Theta} \boldsymbol{h}_{\boldsymbol{q}}^{*T} N_h(\theta)^T \int_T g \boldsymbol{\varphi}_{\boldsymbol{q}}^T & \left[\mathbf{K}\boldsymbol{\varphi}_{\boldsymbol{q}} g + \mathbf{C}\boldsymbol{\varphi}_{\boldsymbol{q}} \dot{g} + \mathbf{M}\boldsymbol{\varphi}_{\boldsymbol{q}} \ddot{g}\right] dt N_h(\theta) \boldsymbol{h}_{\boldsymbol{q}} d\theta \\
= & \int_{\Theta} \boldsymbol{h}_{\boldsymbol{q}}^{*T} N_h(\theta)^T \int_T g \boldsymbol{\varphi}_{\boldsymbol{q}}^T & -[\,\mathbf{K} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k g_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad + \mathbf{C} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k \dot{g}_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad + \mathbf{M} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k \ddot{g}_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad - \mathbf{f}\,]\, dt d\theta
\end{array} \tag{31}$$

Donc :

$$\begin{array}{lll}
 & \int_{\Theta} N_h(\theta)^T \int_T g \boldsymbol{\varphi}_{\boldsymbol{q}}^T & \left[\mathbf{K}\boldsymbol{\varphi}_{\boldsymbol{q}} g + \mathbf{C}\boldsymbol{\varphi}_{\boldsymbol{q}} \dot{g} + \mathbf{M}\boldsymbol{\varphi}_{\boldsymbol{q}} \ddot{g}\right] dt N_h(\theta) \boldsymbol{h}_{\boldsymbol{q}} d\theta \\
= & \int_{\Theta} N_h(\theta)^T \int_T g \boldsymbol{\varphi}_{\boldsymbol{q}}^T & -[\,\mathbf{K} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k g_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad + \mathbf{C} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k \dot{g}_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad + \mathbf{M} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k \ddot{g}_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad - \mathbf{f}\,]\, dt d\theta
\end{array} \tag{32}$$

On sort l'inconnue $\boldsymbol{h}_{\boldsymbol{q}}$ pour obtenir le système :

$$\begin{array}{lll}
 & \int_{\Theta} N_h(\theta)^T \int_T g \boldsymbol{\varphi}_{\boldsymbol{q}}^T & \left[\mathbf{K}\boldsymbol{\varphi}_{\boldsymbol{q}} g + \mathbf{C}\boldsymbol{\varphi}_{\boldsymbol{q}} \dot{g} + \mathbf{M}\boldsymbol{\varphi}_{\boldsymbol{q}} \ddot{g}\right] dt N_h(\theta) d\theta \boldsymbol{h}_{\boldsymbol{q}} \\
= & \int_{\Theta} N_h(\theta)^T \int_T g \boldsymbol{\varphi}_{\boldsymbol{q}}^T & -[\,\mathbf{K} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k g_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad + \mathbf{C} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k \dot{g}_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad + \mathbf{M} \sum_{k=1}^{n-1} (\boldsymbol{\varphi}_{\boldsymbol{q}})_k \ddot{g}_k N_h(\theta) (\boldsymbol{h}_{\boldsymbol{q}})_k \\
& & \quad - \mathbf{f}\,]\, dt d\theta
\end{array} \tag{33}$$

Et pour continuer les calculs il faudrait expliciter la dépendance des termes par rapport à θ.

Bibliographie

[1] P. Ladevèze and G. Duvaut, "Sur une famille d'algorithmes en mécanique des structures," *Comptes-Rendus de l'académie des sciences*, vol. 300, pp. pp 41–44, 1985. No. 2.

[2] P. Ladevèze, "La méthode à grand incrément de temps pour l'analyse de structures à comportement non linéaire décrit par variables internes," *Compte rendu de l'académie des Sciences*, vol. 309, pp. pp 1095–1099, 1989. No. 2.

[3] F. Chinesta, P. Ladevèze, A. Ammar, E. Cueto, and A. Nouy, "Proper generalized decomposition in extreme simulations : Towards a change of paradigm in computational mechanics," *IACM expressions*, vol. 26, pp. pp 2–7, 2009.

[4] C. Ghnatios, F. Masson, A. Huerta, A. Leygue, E. Cueto, and F. Chinesta, "Proper generalized decomposition based dynamic data-driven control of thermal processes," *Computer Methods in Applied Mechanics and Engineering*, vol. 213-216, pp. pp 29–41, 2012.

[5] F. Chinesta, P. Ladevèze, and E. Cueto, "A short review on model order based on proper generalized decomposition," *Archives of Computational Methods in Engineering*, vol. 18, pp. pp 395–404, 2011. Issue 4.

[6] B. Bognet, F. Bordeu, F. Chinesta, A. Leygue, and A. Poitou, "Advanced simulation of models defined in plate geometries : 3d solutions with 2d computational complexity," *Computer Methods in Applied Mechanics and Engineering*, vol. 201, pp. 1–12, 2012.

[7] L. Boucinha, A. Ammar, A. Gravouil, and A. Nouy, "Ideal minimal residual-based proper generalized decomposition for non-symmetric multi-field models–application to transient elastodynamics in space-time domain," *Computer Methods in Applied Mechanics and Engineering*, vol. 273, pp. 56–76, 2014.

[8] F. Louf and L. Champaney, "Fast validation of stochastic structural models using a pgd reduction scheme," *Finite Elements in Analysis and Design*, vol. 70, pp. 44–56, 2013.

[9] C. Ghnatios, F. Chinesta, E. Cueto, A. Leygue, A. Poitou, P. Breitkopf, and P. Villon, "Methodological approach to efficient modeling and optimization of thermal processes taking place in a die : application to pultrusion," *Composites Part A : Applied Science and Manufacturing*, vol. 42, no. 9, pp. 1169–1178, 2011.

[10] A. Chatterjee, "An introduction to the proper orthogonal decomposition," *Current science*, vol. Vol 78 N°7, pp. pp 808–817, 2000.

[11] A. Courard, D. Néron, P. Ladevèze, P. Andolfatto, and A. Bergerot, "Abaques virtuels pour l'optimisation géométrique de structures," *21ème Congrès Français de Mécanique, 26 au 30 août 2013, Bordeaux, France (FR)*, 2013.

[12] A. Ammar, A. Huerta, F. Chinesta, E. Cueto, and A. Leygue, "Parametric solutions involving geometry : a step towards efficient shape optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 268, pp. 178–193, 2014.

[13] A. Nouy, "A priori model reduction through proper generalized decomposition for solving time-dependent partial differential equations," *Computer Methods in Applied Mechanics and Engineering*, vol. 199, no. 23, pp. 1603–1626, 2010.

[14] L. Boucinha, A. Gravouil, and A. Ammar, "Space–time proper generalized decompositions for the resolution of transient elastodynamic models," *Computer Methods in Applied Mechanics and Engineering*, vol. 255, pp. 67–88, 2013.

[15] P. Ladevèze, "Nonlinear computational structural mechanics - new approaches and non-incremental methods of calculation," *Springer Verlag*, 1999.

[16] P. Boucard, P. Ladevèze, M. Poss, and P. Rougee, "A nonincremental approach for large displacement problems," *Computers & Structures*, vol. 64, pp. 499–508, 1997.

[17] J.-Y. Cognard, P. Ladevèze, and P. Talbot, "A large time increment approach for thermo-mechanical problems," *Advances in Engineering Software*, vol. 30, no. 9–11, pp. 583 – 593, 1999.

[18] J.-P. Pelle and D. Ryckelynck, "An efficient adaptive strategy to master the global quality of viscoplastic analysis," *Computers & Structures*, vol. 78, no. 1, pp. 169–183, 2000.

[19] O. Allix and P. Ladevèze, "Interlaminar interface modelling for the prediction of delamination," *Composite Structures*, vol. 22, no. 4, pp. 235 – 242, 1992.

[20] S. Guinard, O. Allix, D. Guédra-Degeorges, and A. Vinet, "A 3d damage analysis of low-velocity impacts on laminated composites," *Composites Science and Technology*, vol. 62, no. 4, pp. 585 – 589, 2002.

[21] K. Saavedra, O. Allix, and P. Gosselet, "On a multiscale strategy and its optimization for the simulation of combined delamination and buckling," *Int Journal for Numerical Methods in Engineering*, vol. 91, pp. 772–798, 2012.

[22] H. Lemoussu, P. Boucard, and P. Ladevèze, "A 3d shock computational strategy for real assembly and shock attenuator," *Advances in Engineering Software*, vol. 33, pp. 517–526, 2002.

[23] L. Champaney, J. Cognard, and P. Ladevèze, "Modular analysis of assemblages of 3d structures with unilateral contact conditions," *Computers & Structures*, vol. 73, pp. 249–266, 1999.

[24] P. Ladevèze, O. Loiseau, and D. Dureisseix, "A micro-macro and parallel computational strategy for highly heterogeneous structures," *Int Jal for Numerical Methods in Engineering*, vol. 52, pp. 121–138, 2001.

[25] P. Ladevèze, A. Nouy, and O. Loiseau, "A multiscale computational approach for contact problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 43, pp. 4869 – 4891, 2002.

[26] P. Guidault, O. Allix, L. Champaney, and J. Navarro, "A two-scale approach with homogenization for the computation of cracked structures," *Computers & Structures*, vol. 85, pp. 1360–1371, 2007.

[27] P. Ladevèze, G. Lubineau, and D. Violeau, "A computational damage micromodel of laminated composites," *Int Jal of Fracture*, vol. 137, pp. 139–150, 2006.

[28] P. Ladevèze, J.-C. Passieux, and D. Néron, "The latin multiscale computational method and the proper generalized decomposition," *Computer Methods in Applied Mechanics & Engineering*, vol. 199, no. 21-22, pp. 1287–1296, 2010.

[29] O. Allix and P. Vidal, "A new multi-solution approach suitable for structural identification problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 25–26, pp. 2727 – 2758, 2002.

[30] H.-M. Nguyen, O. Allix, and P. Feissel, "A robust identification strategy for rate-dependent models in dynamics," *Inverse Problems*, vol. 24, no. 6, pp. 24–, 2008.

[31] P.-A. Boucard and L. Champaney, "A suitable computational strategy for the parametric analysis of problems with multiple contact," *International Journal for Numerical Methods in Engineering*, vol. 57, pp. 1259–1282, 2003.

[32] N. Relun, D. Néron, and P.-A. Boucard, "A model reduction technique based on the pgd for elastic-viscoplastic computational analysis," *Computational Mechanics*, vol. 51, no. 1, pp. 83–92, 2013.