# Évaluation des Méthodes Meshfree pour les Simulations Géomécaniques en Transformations Finies

Evelyne Foerster

ÉCOLE CENTRALE DES ARTS
ET MANUFACTURES
« ÉCOLE CENTRALE PARIS »

# THÈSE

**présentée par**

**Evelyne FOERSTER**

**pour l'obtention du**

## GRADE DE DOCTEUR

**Spécialité : Modélisation Géomécanique**

**Laboratoire d'accueil : Mécanique des Sols, Structures et Matériaux**

**SUJET :**

# Évaluation des Méthodes Meshfree pour les Simulations Géomécaniques en Transformations Finies

**soutenue le : 9 décembre 2003**

**devant un jury composé de :**

| | | |
|---|---|---|
| **MM.** | **O. COUSSY** | **Président** |
| | **A. HUERTA** | **Rapporteur** |
| | **G. PIJAUDIER-CABOT** | **Rapporteur** |
| | **D. AUBRY** | **Examinateur** |
| | **H. MODARESSI** | **Examinateur** |

**2003 - 37**

*à Pierre et Yann*

*à Alexis*

*à Mimi*

# REMERCIEMENTS

Je tiens tout d'abord à exprimer ma profonde reconnaissance à Hormoz MODARESSI, Directeur du Service Aménagements et Risques Naturels du BRGM, qui est à l'origine de ce travail et qui l'a dirigé pendant toutes ces années. Merci pour son soutien permanent et la confiance qu'il a su me redonner tout au long de ces années. Je lui dois mes premiers pas dans la vie professionnelle et plus particulièrement dans le monde de la recherche. Sans lui, le mot même de recherche ne ferait sans doute pas partie de mon vocabulaire. Je le remercie enfin pour son amitié indéfectible qui m'a aidé à continuer malgré les doutes et les difficultés rencontrées.

Je tiens également à remercier chaleureusement Denis AUBRY, Directeur du laboratoire de Mécanique des Sols, Structures et Matériaux (LMSS-Mat) de l'Ecole Centrale de Paris, pour avoir accepter de co-diriger cette thèse et pour avoir faciliter toutes les formalités administratives au niveau de l'Ecole. Son encadrement, ses conseils précieux et ses encouragements ont été très importants pour moi et m'ont beaucoup aider à finaliser ce travail.

Je remercie Monsieur Olivier Coussy, Directeur de l'Institut Navier (Ecole Nationale des Ponts et Chaussées), pour avoir accepté d'examiner ce travail et de présider le jury de ma soutenance.

Je tiens également à remercier vivement Monsieur A. Huerta, Professeur au laboratoire de Calculs Numériques de l'Université Polytechnique de Barcelone, et Monsieur G. Pijaudier-Cabot, Professeur et Directeur de la recherche à l'Ecole Centrale de Nantes, pour avoir accepté d'être rapporteurs de cette thèse.

Je ne saurai oublier l'équipe Sols du laboratoire MSS-Mat de l'Ecole Centrale de Paris, et je remercie plus particulièrement Arezou Modaressi et Fernando Lopez-Caballero qui malgré leur propre charge de travail ont toujours répondu à mes sollicitations.

Je remercie également tous mes collègues du BRGM qui ont contribué directement ou indirectement à la réussite de ce travail.

Enfin, je ne saurai terminer sans remercier infiniment toute ma famille pour m'avoir soutenue patiemment tout au long de ces années et plus particulièrement Mimi qui m'a beaucoup aidée dans les moments difficiles.

# RESUME

Le présent travail propose une synthèse des principales méthodes meshfree (*i.e.* sans maillage) existant actuellement, en fournissant une écriture commune à toutes les méthodes et en mettant ainsi directement en évidence les spécificités de chacune vis-à-vis de la construction des fonctions de forme. L'étude a été réalisée en distinguant cinq grandes familles de méthodes dans le cas d'une formulation variationnelle de type Galerkin :

- les méthodes *MLS*, construites à partir d'une approximation par moindres carrés mobiles (méthodes des éléments diffus ou DEM, ***E**lement **F**ree **G**alerkin* et *h-p Clouds*) ;

- les méthodes *RKM* ou de particules (méthodes ***S**mooth **P**article **H**ydrodynamics*, ***R**eproducing **K**ernel **P**article **M**ethod*, ***R**eproducing **K**ernel **I**nterpolation* et ***M**oving **L**east **S**quare/**R**eproducing **K**ernel*) ;

- les méthodes basées uniquement sur le concept de partition de l'unité (***C**orrected **P**artition of **U**nity **M**ethod*) ;

- les méthodes construites à partir d'une interpolation purement polynomiale (***P**oint **I**nterpolation **M**ethod*) ;

- et enfin les méthodes mixtes couplant des fonctions de bases radiales, soit avec une approximation par moindres carrés mobiles (méthode ***M**oving **L**east **S**quare/**R**adial **B**asis **F**unction*), soit avec une interpolation polynomiale (***R**adial **P**oint **I**nterpolation **M**ethod*).

Plusieurs techniques spécifiques à la mise en œuvre des méthodes meshfree sont également présentées et discutées, notamment concernant l'imposition des conditions aux limites et l'intégration spatiale. Les performances numériques des différentes méthodes et techniques sont ensuite comparées pour des tests mécaniques standards.

La seconde partie de ce travail présente tout d'abord la formulation dynamique non linéaire utilisée pour modéliser les milieux poreux saturés en transformations finies. Puis, nous nous intéressons aux approches adoptées classiquement pour établir la loi de comportement de la phase solide, et reposant sur la décomposition multiplicative du gradient de la transformation. Le modèle complet est finalement évalué sur quelques applications géomécaniques simples et également plus complexes, pour lesquelles nous comparons les résultats obtenus pour différentes méthodes numériques (meshfree et éléments finis).

**Mots-clés** : Méthodes meshfree (sans maillage), méthode des éléments finis, milieux poreux saturés, transformations finies, modélisation numérique lagrangienne non linéaire, sollicitation sismique.

# ABSTRACT

This work presents a synthesis of the main available meshfree methods, providing a unified formalism for all methods and thus enhancing at once specific features for building shape functions. Five groups of Galerkin-type methods were distinguished for this study:

- *MLS* methods based on a moving least square approximation, regrouping essentially Diffuse Element (DEM), Element Free Galerkin (EFGM) and h-p Clouds methods;

- particle or *RKM* methods, namely Smooth Particle Hydrodynamics (SPH), Reproducing Kernel Particle (RKPM), Reproducing Kernel Interpolation (RKI) and Moving Least Square/Reproducing Kernel (MLS/RK) methods;

- methods based solely on a partition of unity concept (Corrected Partition of Unity Method);

- methods based on a polynomial interpolation, such as the Point Interpolation Method (PIM);

- and finally mixed methods combining radial basis functions and either moving least square approximation, such as the Moving Least Square/Radial Basis Function method (MLS/RBF), or polynomial interpolation (Radial PIM).

Several techniques dedicated to the implementation of meshfree methods are presented and discussed as well, specially concerning the imposition of boundary conditions and spatial integration. Numerical performances of the various methods and techniques are then compared on standard mechanical applications.

The second part of this work first deals with the nonlinear dynamic formulation used to model saturated porous media at finite strains. Then, the approaches usually adopted for the constitutive model of the solid phase and based on the multiplicative decomposition of the transformation gradient, are examined. The complete model is finally applied to some simple and also complex geomechanical examples, comparing results obtained with various numerical methods (*i.e.* meshfree and finite elements).

**Key-Words** : Meshfree methods, finite element method, saturated porous media, finite strains, nonlinear lagrangian numerical modeling, seismic loading.

# TABLE DES MATIERES

# Principales Notations

***Notations générales***

La convention de sommation d'Einstein est adoptée pour les indices muets.

Les vecteurs et tenseurs sont soulignés en fonction de leur ordre (les quantités scalaires ne sont pas soulignées) :

$\underline{a} = a_i \underline{e}_i$ vecteur euclidien de composantes $a_i,\ i = 1,2,3$, avec $\underline{e}_i$, vecteur unitaire de la base orthonormée directe associée à l'espace euclidien $E = \mathbb{R}^3$ ;

$\underline{\underline{A}}$ tenseur euclidien du 2nd ordre ;

$\underline{\underline{1}}$ tenseur identité du 2nd ordre (euclidien) ;

$\underline{\underline{\underline{\underline{A}}}}$ tenseur euclidien du 4ème ordre.

D'une manière générale, les opérateurs matriciels sont écrits en caractères gras.

L'opérateur de transposition, ainsi que celui d'inversion, sont placés en exposant à droite de la matrice (ou tenseur) : $\boldsymbol{A}^T$ et $\boldsymbol{A}^{-1}$. Pour simplifier, la combinaison de ces deux opérateurs sera notée : $\boldsymbol{A}^{-T}$.

Les parties symétrique et anti-symétrique d'un opérateur matriciel sont indiquées respectivement par les exposants *S* et *A* placés à droite de l'opérateur : $\boldsymbol{A}^S = \frac{1}{2}\left(\boldsymbol{A} + \boldsymbol{A}^T\right)$ et $\boldsymbol{A}^A = \frac{1}{2}\left(\boldsymbol{A} - \boldsymbol{A}^T\right)$.

***Opérations usuelles sur les champs scalaires, vectoriels et tensoriels***

- Produit scalaire : $\underline{u} \cdot \underline{v} = u_i\, v_i$
- Produit tensoriel :

$$\left(\underline{u} \otimes \underline{v}\right)_{ij} = u_i v_j$$

$$\underline{\underline{A}} = A_{ij}\,\underline{e}_i \otimes \underline{e}_j$$

$$\underline{\underline{\underline{A}}} = A_{ijkl}\,\underline{e}_i \otimes \underline{e}_j \otimes \underline{e}_k \otimes \underline{e}_l$$

- Produit matriciel :

$$\left(\boldsymbol{A}\cdot\boldsymbol{B}\right)_{ij} = A_{ik}B_{kj}$$

$$\left(\boldsymbol{A}\cdot\underline{v}\right)_i = A_{ij}\,v_j$$

- Produit doublement contracté :

$$\underline{\underline{A}} : \underline{\underline{B}} = tr\left(\underline{\underline{A}}\cdot\underline{\underline{B}}^T\right) = A_{ij}\,B_{ij} \text{ avec } tr\left(\underline{\underline{A}}\right) = A_{ii}$$

$$\underline{\underline{A}} : \underline{\underline{B}}\cdot\underline{\underline{C}} = \underline{\underline{A}}\cdot\underline{\underline{C}}^T : \underline{\underline{B}} = \underline{\underline{B}}^T\cdot\underline{\underline{A}} : \underline{\underline{C}}$$

$$\left(\underline{\underline{\underline{D}}} : \underline{\underline{B}}\right)_{ij} = D_{ijkl}\,B_{kl}$$

***Opérateur de dérivations :***

$(.)_{,i} = \partial_{x_i}(.)$ : dérivée partielle d'ordre 1 par rapport à la variable d'espace $x_i$.

$(.)_{,ij} = \partial^2_{x_i x_j}(.)$ : dérivée partielle d'ordre 2 par rapport aux variables d'espace $x_i$ et $x_j$.

$\partial_z(.), d_z(.)$ : dérivées respectivement partielle et totale d'ordre 1 par rapport à la variable $z$.

$d^2_{zz'}(.)$ : dérivée totale d'ordre 2 par rapport aux variables $z$ et $z$'.

$div\underline{a} = a_{i,i}$

$\underline{div\,\underline{\underline{A}}} = A_{ij,j}\,\underline{e}_i$

$\underline{grad\,a} = a_{,i}\,\underline{e}_i$

$\underline{\underline{grad\,\underline{a}}} = a_{i,j}\,\underline{e}_i \otimes \underline{e}_j$

$div(a\underline{u}) = a\,div\underline{u} + \underline{u}\cdot\underline{grad\,a}$

***Notations relatives aux transformations finies***

L'indice situé à gauche d'une quantité indique la configuration de référence utilisée pour sa mesure et l'exposant gauche, la configuration dans laquelle elle est exprimée. Par exemple, ${}^{t'}_{t}A$ signifie que la quantité $A$ est exprimée dans la configuration $\Omega_{t'}$ (instant $t'$ quelconque) et mesurée par rapport à la configuration $\Omega_t$ (instant $t \le t'$, avec $\Omega_0$, la configuration initiale si $t = 0$).

Pour simplifier les notations, on omettra en général l'exposant gauche pour une quantité exprimée dans la configuration actuelle. Par exemple, si l'on note $\Omega = \Omega_{t'}$, la configuration actuelle, alors on écrira : ${}_{t}A = {}^{t'}_{t}A$. Dans le cas des quantités eulériennes, c'est-à-dire mesurées par rapport à la configuration actuelle, l'indice gauche (et en général l'exposant gauche) sera omis : $A = {}^{t'}A = {}^{t'}_{t'}A$. La même règle s'applique aux opérateurs. Par exemple, pour un champ vectoriel $\underline{u}$ exprimé dans la configuration actuelle et mesuré par rapport à la configuration actuelle $\Omega = \Omega_{t'}$ ou de référence $\Omega_t$, on notera :

$$\begin{cases} div\underline{u} = u_{i,i} \\ \underline{\underline{grad}}\,\underline{u} = u_{,i}\,\underline{e}_i \end{cases} \text{ avec } \underline{u} = {}^{t'}\underline{u} \text{ (mesure sur } \Omega)$$

$$\begin{cases} {}_{t}div\underline{u} = {}_{t}u_{i,i} \\ {}_{t}\underline{\underline{grad}}\,\underline{u} = {}_{t}u_{,i}\,\underline{e}_i \end{cases} \qquad \text{avec } \underline{u} = {}_{t}\underline{u} = {}^{t'}_{t}\underline{u} \text{ (mesure sur } \Omega_t)$$

$\underline{x}$ représentera le vecteur position eulérien d'un point matériel, mesuré dans la configuration actuelle et exprimé par rapport à un repère cartésien fixe de l'espace euclidien $E = \mathbb{R}^3$.

$\underline{X} = {}_{t}\underline{x}$ représentera le vecteur position lagrangien d'un point matériel, mesuré dans la configuration de référence $\Omega_t$ considérée à l'instant *t*, et exprimé par rapport au même repère cartésien fixe.

${}_{t}(.)_{,i} = \partial_{{}_{t}x_i}(.)$ représentera la dérivée partielle d'une quantité par rapport à la direction spatiale ${}_{t}x_i$, mesurée dans la configuration $\Omega_t$. Si $\Omega_t$ représente la configuration de référence, on notera également : ${}_{t}(.)_{,i} = \partial_{X_i}(.)$ et pour la configuration actuelle, on notera simplement : $(.)_{,i} = \partial_{x_i}(.)$.

Pour exprimer le mouvement d'une particule matérielle entre les configurations de référence $\Omega_t$ et actuelle $\Omega = \Omega_{t'}$, on définit une application bijective $\underline{\Phi}$ entre $\Omega_t$ et $\Omega$, telle que : $\underline{x} = \underline{\Phi}(\underline{X}, t')$ ou $\underline{X} = \underline{\Phi}^{-1}(\underline{x}, t')$.

Dans ce cas, on peut écrire :

| | |
|---|---|
| $_t\underline{u} = {}^{t'}_{t}\underline{u} = \underline{\Phi}\left(\underline{X}, t'\right) - \underline{X}$ | Vecteur des déplacements (mesure sur la configuration de référence $\Omega_t$). |
| $\underline{u} = {}^{t'}\underline{u} = \underline{x} - \underline{\Phi}^{-1}\left(\underline{x}, t'\right)$ | Vecteur des déplacements (mesure sur la configuration actuelle). |
| $_t\dot{\underline{u}} = {}_t\underline{v} = \partial_t \underline{\Phi}$ | Vecteur vitesse (mesure sur la configuration de référence $\Omega_t$). |
| $\dot{\underline{u}} = \underline{v} = \partial_t \underline{\Phi}\left(\underline{\Phi}^{-1}\left(\underline{x}, t'\right), t'\right)$ | Vecteur vitesse (mesure sur la configuration actuelle). |
| $_t\ddot{\underline{u}} = {}_t\dot{\underline{v}} = \partial^2_{tt} \underline{\Phi}$ | Vecteur accélération (mesure sur la configuration de référence $\Omega_t$). |
| $\ddot{\underline{u}} = \dot{\underline{v}} = \partial^2_{tt} \underline{\Phi}\left(\underline{\Phi}^{-1}\left(\underline{x}, t'\right), t'\right)$ | Vecteur accélération (mesure sur la configuration actuelle). |
| $\dot{f} = d_t f = \partial_t f + \underline{v} \cdot \underline{gradf}$ | Dérivée particulaire de la quantité spatiale $\underline{f}$. |
| $_t\underline{\underline{F}} = {}_t\underline{\underline{grad \Phi}}\left(\underline{X}, t'\right)$ | Tenseur gradient de la transformation entre les configurations de référence $\Omega_t$ et actuelle $\Omega = \Omega_{t'}$. |
| $_t\dot{\underline{\underline{F}}} = {}_t\underline{\underline{grad \partial_t \Phi}}$ | Dérivée du tenseur gradient de la transformation entre les configurations de référence $\Omega_t$ et actuelle $\Omega = \Omega_{t'}$. |
| $_t J = det\, {}_t\underline{\underline{F}}$ | Jacobien de la transformation entre les configurations de référence $\Omega_t$ et actuelle $\Omega = \Omega_{t'}$. |
| $\underline{\underline{l}} = \underline{\underline{grad\, v}} = {}_t\dot{\underline{\underline{F}}} \cdot {}_t\underline{\underline{F}}^{-1}$ | Tenseur gradient de vitesse (mesure sur la configuration actuelle). |
| $_t\underline{\underline{C}} = {}_t\underline{\underline{F}}^T \cdot {}_t\underline{\underline{F}}$ | Tenseur de Cauchy - Green droit (mesure sur la configuration de référence $\Omega_t$). |
| $\underline{\underline{b}} = {}_t\underline{\underline{F}} \cdot {}_t\underline{\underline{F}}^T$ | Tenseur de Cauchy – Green gauche (mesure sur la configuration actuelle). |
| $_t\underline{\underline{E}} = \frac{1}{2}\left({}_t\underline{\underline{C}} - \underline{\underline{1}}\right)$ | Tenseur des déformations de Green – Lagrange (mesure sur la configuration de référence $\Omega_t$). |
| $\underline{\underline{\varepsilon}} = \frac{1}{2}\left(\underline{\underline{grad\, u}} + \underline{\underline{grad\, u}}^T\right)$ | Tenseur linéarisé des déformations (mesure sur la configuration actuelle). |
| $_t\Delta A = {}^{t+\Delta t}_{t}A - {}^{t}_{t}A$ | Incrément de la quantité $A$ entre les instants $t$ et $t+\Delta t$ (mesure sur la configuration de référence $\Omega_t$). |

| | |
|---|---|
| ${}_t\delta A$ | Variation virtuelle de la quantité $A$ entre les instants $t$ et $t+\Delta t$ (mesure sur la configuration de référence $\Omega_t$). |
| $\underline{\underline{d}} = \underline{\underline{l}}^S$ | Tenseur des taux de déformations (mesure sur la configuration actuelle). |
| $\underline{\underline{\omega}} = \underline{\underline{l}}^A$ | Tenseur des taux de rotations (mesure sur la configuration actuelle). |
| ${}_t\dot{\underline{\underline{E}}} = {}_t\underline{\underline{F}}^T \cdot \underline{\underline{d}} \cdot {}_t\underline{\underline{F}}$ | Tenseur des taux de déformations de Green – Lagrange (mesure sur la configuration de référence $\Omega_t$). |
| $\underline{\underline{\sigma}}$ | Tenseur des contraintes totales de Cauchy (mesure sur la configuration actuelle). |
| ${}_t\underline{\underline{\Pi}} = J\underline{\underline{\sigma}} \cdot {}_t\underline{\underline{F}}^{-T}$ | Premier tenseur des contraintes totales de Piola-Kirchhoff (mesure sur la configuration de référence $\Omega_t$). |
| ${}_t\underline{\underline{S}} = {}_tJ\,{}_t\underline{\underline{F}}^{-1} \cdot \underline{\underline{\sigma}} \cdot {}_t\underline{\underline{F}}^{-T}$ | Second tenseur des contraintes totales de Piola-Kirchhoff (mesure sur la configuration de référence $\Omega_t$). |
| $\underline{\underline{\tau}} = {}_tJ\underline{\underline{\sigma}}$ | Tenseur des contraintes totales de Kirchhoff-Trefftz (mesure sur la configuration actuelle). |
| $\underline{\underline{\tau}} : \underline{\underline{d}} = {}_t\underline{\underline{S}} : {}_t\dot{\underline{\underline{E}}} = {}_t\underline{\underline{\Pi}} : {}_t\dot{\underline{\underline{F}}}$ | Dualité contraintes - déformations (taux de) |
| ${}_t\underline{\underline{F}} = {}_t\underline{\underline{R}} \cdot {}_t\underline{\underline{U}} = \underline{\underline{V}} \cdot {}_t\underline{\underline{R}}$ | Décomposition polaire de la transformation, avec :<br>• ${}_t\underline{\underline{R}}$ : tenseur de rotation orthogonale ( ${}_t\underline{\underline{R}}^T \cdot {}_t\underline{\underline{R}} = {}_t\underline{\underline{R}} \cdot {}_t\underline{\underline{R}}^T = \underline{\underline{1}}$ ) ;<br>• ${}_t\underline{\underline{U}}$ : tenseur des déformations pures droit ;<br>• $\underline{\underline{V}}$ : tenseur des déformations pures gauche. |
| ${}_t\underline{\underline{\Omega}} = {}_t\dot{\underline{\underline{R}}} \cdot {}_t\underline{\underline{R}}^T$ | Autre tenseur des taux de rotations utilisés pour certaines dérivées objectives (tenseur antisymétrique). |

Remarques :

- D'après la définition du tenseur ${}_t\underline{\underline{F}}$, tout vecteur $\underline{a}$ de la configuration actuelle peut s'écrire : $\underline{a} = {}_t\underline{\underline{F}} \cdot {}_t\underline{A}$, où ${}_t\underline{A}$ représente le transporté de $\underline{a}$ dans la configuration de référence $\Omega_t$. Nous utiliserons également les relations de transformations suivantes au cours de nos travaux :

$$\underline{gradf}\cdot\underline{a}=\left[{}_t\underline{\underline{F}}^{-T}\cdot{}_t\underline{gradf}\right]\cdot\underline{a}={}_t\underline{gradf}\cdot\left[{}_t\underline{\underline{F}}^{-1}\cdot\underline{a}\right]={}_t\underline{gradf}\cdot{}_t\underline{A}\quad\forall\underline{a}\in\Omega$$

$$\underline{\underline{grad\,a}}={}_t\underline{\underline{F}}^{-T}\cdot{}_t\underline{\underline{grad\,a}}$$

$$div\underline{a}={}_t div\left[{}_t\underline{\underline{F}}^{-1}\cdot\underline{a}\right]={}_t div\,{}_t\underline{A}$$

- Pour exprimer les variations virtuelles ou incréments de la quantité $A$, nous avons supposé que l'intervalle de temps d'étude $[0,T]$ était découpé en un certain nombre de pas de temps notés $\Delta t$.

Par ailleurs, nous rappelons le théorème de Gauss :

$$\forall\underline{\underline{a}}\in\Omega,\ \int_\Omega\underline{\underline{div\,a}}\,d\Omega=\int_\Gamma\underline{\underline{a}}\cdot\underline{n}\,d\Gamma$$

où $\Gamma$ représente la frontière du domaine $\Omega$.

***Notations relatives aux milieux poreux saturés***

De manière générale, nous utiliserons les exposants *s* et *w* pour désigner les quantités se rapportant respectivement aux phases solide et fluide saturant, et l'exposant $\alpha$ sera utilisé pour représenter l'une ou l'autre phase.

La convention de signe de la Mécanique des Milieux Continus sera adoptée pour les contraintes (compression négative).

Nous utiliserons également les notations suivantes :

| | |
|---|---|
| $\Omega=\Omega_{t'}$ | Volume total du milieu poreux dans la configuration actuelle considérée à l'instant $t'$ quelconque ($\Omega_t$ : volume à l'instant $t$ pris comme référence). Par abus de langage, $\Omega$ sera appelé configuration actuelle et $\Omega_t$, configuration de référence ($\Omega_0$ : configuration initiale). |
| $\delta\Omega$ | Volume élémentaire quelconque inclus dans $\Omega$ ($\delta\Omega_t=\delta\Omega/{}_tJ$ : volume élémentaire de la configuration $\Omega_t$). |
| $\Gamma$ | Frontière du volume $\Omega$ ($\Gamma_t$ pour la frontière du volume $\Omega_t$). |
| $\underline{n}$ | Normale extérieure à la frontière $\Gamma$ ($\underline{N}={}_t\underline{n}$ pour la normale extérieure à la frontière $\Gamma_t$). |
| $n$ | Porosité du milieu poreux dans la configuration actuelle ($n_t$ : porosité sur $\Omega_t$). |
| $\rho^\alpha=\rho^\alpha(\underline{x},t)$ | Masse volumique de la phase $\alpha$ en un point $\underline{x}$ de la configuration actuelle (${}^t\rho^\alpha$ : masse volumique sur $\Omega_t$). |

| | |
|---|---|
| $\theta^{\alpha}$ | Fraction volumique de l'espace poreux total occupé par la phase $\alpha$ dans la configuration actuelle ($\theta^{w}=n$, $\theta^{s}=1-n$). |
| $\rho^{sat}=\sum_{\alpha}\rho^{\alpha}\theta^{\alpha}$ | Masse volumique moyenne du milieu poreux saturé dans la configuration actuelle ($^{t}\rho^{sat}$ : masse volumique sur $\Omega_t$). |
| $_{t}J^{\alpha}=\theta^{\alpha}/{}^{t}\theta^{\alpha}$ | Variation de la fraction volumique de l'espace poreux total occupé par la phase $\alpha$ entre les configurations $\Omega_t$ et actuelle. |
| $\underline{v}^{rw}\left(\underline{x},t'\right)=\underline{v}^{w}-\underline{v}^{s}$ | Vecteur vitesse relative de la phase fluide par rapport à la phase solide. |
| $d_{t}^{\alpha}\left(.\right)$ | Dérivée particulaire en suivant la phase $\alpha$ (c'est-à-dire exprimée avec la vitesse $\underline{v}^{s}$ pour le solide et $\underline{v}^{w}$ pour le fluide). |
| $\beta^{w}$ | Coefficient de compressibilité de la phase fluide (supposée constant). |
| $k^{w}$ | Perméabilité isotrope absolue du milieu poreux (supposée constante). |
| $\mu^{w}$ | Viscosité dynamique de la phase fluide (supposée constante). |
| $\underline{\underline{\sigma}}$, $\underline{\underline{\sigma}}'$ | Tenseur des contraintes totales et effectives de Cauchy dans le milieu poreux. |

***Rappel sur les grandeurs matériellement extensives***

Les densités volumiques de masse, de quantité de mouvement, d'énergies interne et cinétique, d'entropie sont des grandeurs géométriquement (ou matériellement) extensives : la quantité de la grandeur extensive $a\left(\underline{x},t'\right)$ contenue dans un domaine $\Omega$ quelconque, est égale à la somme des contributions de chacun des milieux continus (phases) contenus dans $\Omega$ :

$$\begin{cases} a\left(\underline{x},t\right)=\sum_{\alpha}a^{\alpha}\left(\underline{x},t'\right) \\ a^{\alpha}\left(\underline{x},t'\right)=\int_{\Omega}\overline{a}^{\alpha}\left(\underline{x},t'\right)d\Omega \end{cases}$$

où $\overline{a}^{\alpha}\left(\underline{x},t'\right)$ représente la densité eulérienne volumique de la grandeur considérée pour la phase $\alpha$.

# INTRODUCTION GENERALE

La prédiction quantitative des instabilités du sol, qu'elles soient ou non d'origine sismique (phénomènes de liquéfaction, glissements statiques, effondrements, écroulements, etc.), qu'elles soient superficielles ou profondes, de faible ampleur ou bien à en grande masse, constituent aujourd'hui un enjeu majeur :

- pour les autorités civiles qui cherchent à prévoir et à prévenir les catastrophes naturelles en mettant en place des systèmes de surveillance et/ou des dispositifs d'alerte ;

- pour les équipes en charge du dimensionnement des ouvrages nouveaux ou devant évaluer la tenue des ouvrages existants ;

- pour les équipes devant mettre en œuvre des mesures de protection et de renforcement dans les zones à risque.

Quelle que soit leur ampleur, les instabilités du sol représentent un risque préoccupant pour les personnes et pour les biens, car les dégâts induits sont souvent très importants au niveau des constructions, mais également des infrastructures et des réseaux enterrés. Ce risque constitue une contrainte supplémentaire pour l'aménagement et l'urbanisme et son coût économique est généralement proportionnel à la surface du territoire menacé. Il est à présent largement pris en compte dans les politiques d'aménagement et influence donc les règlements d'urbanisme. Dans certaines régions ou agglomérations, il intervient sur une proportion telle de l'espace disponible qu'il peut limiter, freiner, voire stopper toute nouvelle possibilité de développement. L'évaluation précise des zones d'influence de ce risque, c'est-à-dire susceptibles d'être réellement affectées par les instabilités du sol, peut donc devenir prioritaire pour certaines communautés.

Lorsque la typologie de l'instabilité est connue, on cherche généralement à répondre aux questions : où ? quand ? comment ? jusqu'où ? Pour ce faire, trois phases principales peuvent être identifiées :

- une phase d'initiation (où ?) : identification des facteurs de prédisposition (géologie, topographie, pentes, lithologie, contextes hydrologiques, altération, érosion, etc.) et des mécanismes physiques impliqués dans le déclenchement de l'instabilité (séismes, fortes pluies, aménagement et activités humaines, etc.) ;

- une phase de rupture (quand, comment ?) : évaluation des critères d'instabilité et caractérisation de la rupture dans l'espace (rupture diffuse ou localisée suivant une surface identifiable) ;

- une phase de propagation (jusqu'où ?) : évaluation de la trajectoire des masses mobilisées, quantification de l'ampleur du mouvement et identification des facteurs naturels ou anthropiques susceptibles de l'arrêter.

Les phases d'initiation et de rupture peuvent généralement être regroupées au niveau de la modélisation numérique, pour laquelle l'hypothèse des transformations infinitésimales est classiquement utilisée. De ce fait, les variables d'Euler et de Lagrange sont confondues. Cette hypothèse n'est cependant plus applicable lorsque l'on cherche à modéliser la propagation des masses de sols mobilisées. Il est dans ce cas nécessaire d'utiliser l'hypothèse des transformations finies.

L'étude des transformations finies, thème classique de la Mécanique des Milieux Continus, constitue actuellement un axe de recherche important dans son application à la géomécanique. Du fait qu'elles s'inscrivent dans un cadre où la continuité de la matière est respectée (pas de cavitation, de dislocation ou de déchirure), les transformations finies présentent l'intérêt d'être quantitatives, contrairement aux méthodes de la mécanique des milieux discrets qui sont plus qualitatives et pas toujours très physiques. En termes de modélisation numérique et notamment appliquée aux sols, les transformations finies posent néanmoins des problèmes importants du fait des fortes non-linéarités qu'elles induisent à la fois au niveau de la géométrie du problème qu'au niveau du comportement rhéologique des matériaux constitutifs. Une autre difficulté, peut-être plus spécifique aux sols, est que les lois de comportement en transformations finies sont encore assez mal connues, essentiellement du fait que la plupart des caractérisations physiques et mécaniques effectuées en laboratoire, sont réalisées en petites déformations.

Les méthodes usuelles de modélisation numérique reposent sur l'utilisation d'un maillage permettant de discrétiser la géométrie du domaine à étudier. Cependant, le maillage peut devenir une tâche assez fastidieuse, lorsque la géométrie du problème est trop complexe ou lorsque le calcul nécessite des remaillages successifs. En effet, certains problèmes requièrent de réaliser des raffinements du maillage initial (ex : coexistence de zones ayant des niveaux de déformation très différents ou ayant des mouvements relatifs très importants) ou de faire évoluer le maillage avec le phénomène étudié (problèmes évolutifs). En outre, le maillage doit souvent respecter certaines contraintes géométriques, telle la conformité des éléments pour la méthode des éléments finis. D'une façon générale, l'exigence d'un maillage adéquat conduit à des difficultés numériques liées aux techniques de remaillage, à une certaine lourdeur des calculs décuplée par le nombre de remaillages à effectuer. L'émergence récente des méthodes sans maillage découle donc de la volonté de s'affranchir des difficultés liées à la génération et à la gestion des maillages. Le concept fondateur des ces méthodes est la non-existence du maillage, ce qui leur confère une plus grande souplesse vis-à-vis de la discrétisation spatiale. En effet, la discrétisation des équations aux dérivées partielles du problème considéré repose sur un ensemble de nœuds non connectés les uns aux autres. Lorsqu'un raffinement local de la solution est nécessaire, il est possible soit de repositionner ou d'augmenter le nombre de nœuds, soit d'augmenter le degré de l'approximation utilisée. Par ailleurs, pour les problèmes transitoires dont la solution présente un fort gradient, il est également possible d'utiliser un ensemble de nœuds mobiles pour suivre le gradient du champ solution. Ce type de méthodes est donc particulièrement adapté pour simuler les phénomènes d'instabilités pour lesquels la mobilisation d'une grande masse de sol peut se produire.

Cependant, étant donné le nombre important de méthodes meshfree existant actuellement, il nous est tout d'abord apparu indispensable d'homogénéiser les formalismes de façon à mieux cerner les spécificités éventuelles des principales méthodes actuellement à notre disposition.

L'évaluation des principales méthodes meshfree était également souhaitable dans l'optique de les utiliser pour simuler différents aspects de la géomécanique en transformations finies, et notamment les instabilités du sol. Afin de répondre à ces différents objectifs, nous avons organisé ce mémoire en quatre chapitres et une annexe.

Le *premier chapitre* présente la synthèse des formalismes des principales méthodes meshfree existant actuellement, méthodes regroupées en cinq grandes familles d'approximation, à savoir :

- les méthodes *MLS* (« *Moving Least Square* »), construites à partir d'une approximation par moindres carrés mobiles ;
- les méthodes *PU* (« *Partition of Unity* »), basées uniquement sur le concept de partition de l'unité ;
- les méthodes que nous qualifierons de *RKM* (« *Reproducing Kernel Methods* ») ou encore méthodes de particules ;
- les méthodes que nous qualifierons de polynomiales, car construites à partir d'une interpolation purement polynomiale ;
- et enfin les méthodes que nous qualifierons de *RBF*, car elles utilisent des fonctions de bases radiales (« *Radial Basis Function* »).

La majorité des méthodes meshfree présentées dans ce mémoire, ont été implémentées dans le logiciel « *MoveFree* », outil numérique développé en langage orienté objet (C++) dans le cadre de ce travail, et intégrant également la méthode des éléments finis. Une présentation de la structure de données et de l'architecture du logiciel est fournie en *annexe*.

Le *second chapitre* est consacré aux techniques numériques utilisables pour une implantation pratique des méthodes meshfree dans un outil de calculs. Quelques applications numériques réalisées avec « *MoveFree* » pour des problèmes mécaniques simples sont également présentées dans ce chapitre, essentiellement afin d'évaluer et de comparer les performances numériques entre les différentes méthodes meshfree du premier chapitre.

Dans le *troisième chapitre*, nous rappelons d'une part, la formulation mathématique du modèle dynamique applicable aux milieux poreux saturés en transformations finies. Nous présentons d'autre part, les principales approches théoriques utilisées pour les modèles de comportement élastoplastiques en transformations finies, à savoir l'approche basée sur l'hypoélasticité et celle reposant sur l'hyperélasticité. Enfin, nous abordons les différents aspects de la modélisation numérique, à savoir la formulation variationnelle en description lagrangienne totale, les discrétisations en espace et en temps du système obtenu, et le schéma de résolution non linéaire adopté.

Le *dernier chapitre* est consacré aux simulations numériques réalisées en transformations finies avec les méthodes meshfree qui nous ont semblé les plus performantes. Les premières simulations présentées consistent en des applications simples dont la solution est généralement connue, afin de valider l'implantation dans « *MoveFree* » de la formulation présentée au chapitre 3. Au niveau de la dernière simulation, nous analysons la réponse sismique 2D du barrage en terre de « *El Infiernillo* » (Mexique) lors du tremblement de terre du 14/3/1979. Pour ce faire, nous comparons les résultats obtenus avec trois méthodes

meshfree différentes, à savoir les méthodes RKPM, EFGM et RPIM, ainsi qu'avec la méthode des éléments finis, en utilisant la loi de comportement élastoplastique multi-mécanismes de Hujeux [AUB 82 ; HUJ 85] dans le cadre des transformations finies.

# 1. Synthèse des Principales Méthodes Meshfree

## 1.1 INTRODUCTION

Les méthodes sans maillage, appelées couramment « meshless » ou encore « meshfree », font l'objet de recherches importantes seulement depuis une dizaine d'années. Cependant dès la fin des années 70, Lucy [LUC 77] proposait la méthode SPH (*Smooth Particle Hydrodynamics*), pour simuler l'interaction des particules célestes. Monaghan et co-auteurs [GIN 77 ; MON 82 ; MON 88], en lui apportant une base plus théorique, ouvraient la voie vers les méthodes meshfree. De nombreux auteurs ont ensuite largement étudié et utilisé cette méthode dans divers domaines de la mécanique [SWE 93 ; SWE 95 ; JOH 96 ; RAN 96]. Vers le milieu des années 90, Liu et co-auteurs [LIU 95a ; LIU 95b] introduisait la méthode RKPM (*Reproducing Kernel Particle Method*), dérivée de la méthode SPH, afin d'obtenir de meilleures performances numériques, notamIment en termes de précision. Une autre famille de méthodes sans maillage est apparue au cours des années 90 [NAY 91a ; NAY 91b ; NAY 92 ; BEL 94 ; BEL 96 ; MOD 96 ; MOD 97 ; MOD 98 ; AUB 97 ; MUK 97a ; MUK 97b], à partir des travaux de Lancaster et Salkauskas [LAN 81] sur les moindres carrés mobiles. Enfin plus récemment, certaines méthodes meshfree ont permis de combiner divers types d'approximations [BEL 95 ; MOD 95 ; LIU 97 ; WAN 00 ; WAN 02a ; WAN 02b].

Suivant la définition donnée par Krysl et Belytschko [KRY 00], une méthode peut-être considérée comme meshfree si la base d'approximation est construite à partir de supports pouvant être associés à des nœuds dispersés et se recouvrant de manière arbitraire, sans recourir à une partition du domaine considéré en sous-domaines juxtaposés, comme c'est le cas pour les éléments finis ou les différences finies. Cette définition recoupe en fait le principe de « la partition de l'unité », que l'on retrouve notamment dans les travaux de Duarte et Oden [DUA 95 ; DUA 96a ; DUA 96b], ainsi que de Babuška et Melenk [BAB 95; MEL 96 ; BAB 97]. Ce principe peut être considéré comme une généralisation de la plupart des méthodes meshfree, mais également des méthodes avec maillage (FEM : éléments finis ou DF : différences finies). Nous rappelons dans la section §1.2, le principe et quelques propriétés de la partition de l'unité.

Par ailleurs, il n'est pas dans l'objectif de ce mémoire de présenter les fondements mathématiques complets relatif au formalisme ou aux propriétés (convergence, stabilité, etc.) des méthodes meshfree présentées dans les sections §1.3 à §1.6, ceux-ci pouvant se trouver dans la littérature de plus en plus abondante sur le sujet. Notre contribution consistera notamment à établir un tableau synthétique de ces différentes méthodes, dans l'optique de fournir une écriture commune à toutes les méthodes et ainsi de mettre directement en évidence les spécificités de chacune vis-à-vis de la construction des fonctions de forme.

Pour ce faire, nous avons choisi de classer les principales méthodes meshfree suivant les grandes familles suivantes :

- les méthodes que nous qualifierons de *MLS* (« *Moving Least Square* »), c'est-à-dire construites à partir d'une approximation par moindres carrés mobiles. Cette famille regroupe essentiellement les méthodes DEM (« *Diffuse Element Method* » [NAY 91a ; NAY 91b ; NAY 92]), EFGM (« *Element Free Galerkin Method* », [BEL 94]) et h-p Clouds [DUA 95 ; DUA 96a ; DUA 96b]) ;

- les méthodes *PU* (« *Partition of Unity* »), basées uniquement sur le concept de partition de l'unité, avec notamment la méthode PUM (« *Partition of Unity Method* », [BAB 95 ; BAB 97 ; MEL 96]), modifiée par Krysl et Belytschko [KRY 00], et que nous appellerons C-PUM dans ce travail (« *Corrected PUM* ») ;

- les méthodes que nous qualifierons de *RKM* (« *Reproducing Kernel Methods* ») ou encore méthodes de particules, regroupant notamment les méthodes SPH, RKPM (« *Reproducing Kernel Particle Method* »), RKI (« *Reproducing Kernel Interpolation* », [CHE 03]) et MLS/RK (« *Moving Least Square Reproducing Kernel Method* », [LIU 97]), cette dernière méthode combinant les approximations *MLS* et *RKM* ;

- les méthodes que nous qualifierons de polynomiales, car construites à partir d'une interpolation purement polynomiale, telle que la méthode PIM (« *Point Interpolation Method* », [LIU 99 ; LIU 01a]) ;

- et enfin les méthodes que nous qualifierons de *RBF*, car elles utilisent des fonctions de bases radiales (« *Radial Basis Function* »), incluant notamment une méthode que nous avons appelée MLS/RBF (approximation *MLS* utilisant les fonctions de bases radiales), ainsi que la méthode RPIM (« *Radial Point Interpolation Method* », [WAN 00 ;WAN 02a ; WAN 02b]), ainsi nommée car elle est basée sur une interpolation couplant une base polynomiale comme pour la méthode PIM et une base de fonctions radiales à support global.

Avertissements :

- Toutes les méthodes analysées dans le cadre de nos travaux reposent sur une formulation de type Galerkin. De ce fait, nous ne présenterons pas la méthode MLPG (« *Meshfree Local Petrov-Galerkin Method* »), méthode *MLS* développée initialement par Atluri et Zhu [ATL 98] et utilisée dans le cadre d'une formulation variationnelle locale de type Petrov-Galerkin, ni les méthodes LPIM (« *Local Point Interpolation Method* » [LIU 01b]) et LRPIM (« *Local Radial Point Interpolation Method* » [LIU 01c]). Une synthèse de ces méthodes (MLPG, LPIM, LR-PIM et BMM) peut également être trouvée dans le livre de G.R. Liu [LIU 03].

- Par ailleurs, n'utilisant pas dans ce travail, les techniques de résolution des problèmes aux limites par équations intégrales, nous avons choisi de ne pas présenter les méthodes meshfree de frontière ou BMM (« *Boundary Meshfree Methods* »), méthodes *MLS*, polynomiales ou *RBF*, dont le formalisme s'apparente à celui utilisé dans la méthode des éléments frontières (BEM). On peut citer notamment les méthodes BNM (« *Boundary Node Method* », [MUK 97a]), BPIM (« *Boundary Point Interpolation Method* », [LIU 00b ; GU 02]), BRPIM (« *Boundary Radial Point Interpolation Method* », [GU 01]) et

LBIE (« *Local Boundary Integral Equation Method* », [ZHU 98]). De même que précédemment, une synthèse peut être trouvée dans [LIU 03].

Au niveau de ce premier chapitre, nous avons choisi de présenter le formalisme général pour chacune des familles d'approximation ci-dessus, et pour chacune, nous détaillons le principe et les propriétés des différentes méthodes associées, méthodes que nous avons par ailleurs implantées dans l'outil de calculs *MoveFree*, outil écrit en langage C++ et développé spécifiquement dans le cadre de ce travail.

Enfin, nous renvoyons le lecteur au second chapitre de ce mémoire, pour tous les aspects numériques concernant la mise en œuvre pratique des méthodes meshfree, ainsi que pour les applications numériques simples que nous avons réalisées avec *MoveFree*, afin d'évaluer et de comparer les performances des différentes méthodes (convergence, précision, etc.) pour des problèmes mécaniques donnés. Les applications géomécaniques plus complexes sont quant à elles regroupées au niveau du chapitre 4.

## 1.2 PRINCIPE DE LA PARTITION DE L'UNITE

Si l'on considère un ouvert convexe $\Omega$ de frontière $\Gamma$, inclus dans l'espace euclidien $\mathbb{E}=\mathbb{R}^n$ à $n$ dimensions (avec $n=1,2 \text{ ou } 3$) et soit un nuage de $N$ points, souvent appelés « nœuds », répartis arbitrairement sur le fermé $\overline{\Omega}=\Omega\cup\Gamma$ et de positions $\underline{x}_I, I=1,N$. A ce nuage de points peut être associé l'ensemble $\mathcal{B}_N$ des boules ouvertes $\Omega_I=\mathcal{B}\left(\underline{x}_I,h_I\right)$, définies par : $\Omega_I=\left\{\underline{x}\in\mathbb{E}/\ \left\|\underline{x}_I-\underline{x}\right\|_2<h_I\right\}$ et tel que : $\mathcal{B}_N=\bigcup_{I=1}^{N}\Omega_I$ et $\overline{\Omega}\subset\mathcal{B}_N$.

Une classe de fonctions $\mathcal{F}_N=\left\{\phi_I\right\}_{I=1}^{N}$ constitue une partition de l'unité associée à l'ouvert $\mathcal{B}_N$, si elle vérifie les conditions suivantes :

$$\forall I,\phi_I\in C^s\left(\Omega_I\right), s\in\mathbb{N} \tag{1.1}$$

$$\forall x\in\overline{\Omega}, \sum_{I=1}^{N}\phi_I\left(\underline{x}\right)=1 \tag{1.2}$$

Au niveau de la terminologie, la boule $\Omega_I$, support de la fonction de forme $\phi_I$, sera appelée *voisinage* ou *domaine d'influence* du nœud *I*. $h_I$ sera appelé *rayon* du domaine d'influence.

La définition topologique de $\Omega_I$ peut être de nature radiale (un segment en 1D, un disque en 2D ou une sphère en 3D), rectangulaire (un segment en 1D, un rectangle en 2D, un hexaèdre en 3D), ou encore elliptique (ellipsoïde en 3D). Les figures 1 et 2 montrent par exemple deux couverts 2D avec supports radiaux et rectangulaires.

Les supports non radiaux permettent notamment de suivre une certaine anisotropie matérielle ou comportementale existant au niveau du modèle, tandis que le premier type est mieux adapté dans le cas de problèmes isotropes. Le cas rectangulaire est également avantageux lorsqu'un arrangement régulier des nœuds est choisi pour la simulation. $\Omega_I$ est alors défini sous forme de composantes : $\Omega_I=\left\{\underline{x}\in\mathbb{E}/\ \left|x_{Ij}-x_j\right|<h_{Ij}, j=1,n\right\}$, en considérant une mesure vectorielle $\underline{h}_I$ du support et non plus scalaire comme dans le cas radial.

Remarques :

- On préfère généralement utiliser des supports compacts, plutôt que des ouverts pour la définition des domaines d'influence.
- Il est à noter que les fonctions $\phi_I$ peuvent prendre des valeurs négatives. Elles sont par ailleurs nulles en dehors de leur support. Nous verrons plus loin comment sont construites ces fonctions suivant la famille de méthodes meshfree considérée.
- La condition (1.2) correspond en fait à ce que l'on nomme « consistance d'ordre zéro » de l'approximation considérée. Nous reviendrons plus en détails sur la notion de consistance dans les sections suivantes.

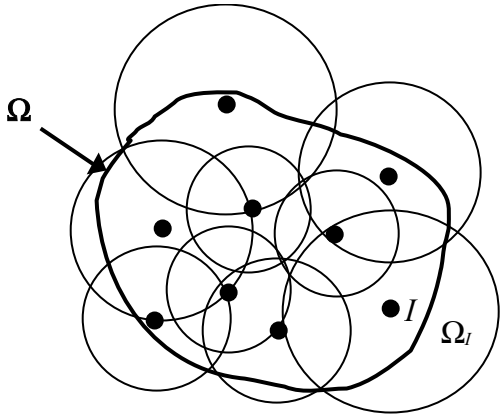



***Fig. 1.1 : Exemple de recouvrement 2D d'un domaine Ω : supports circulaires.***

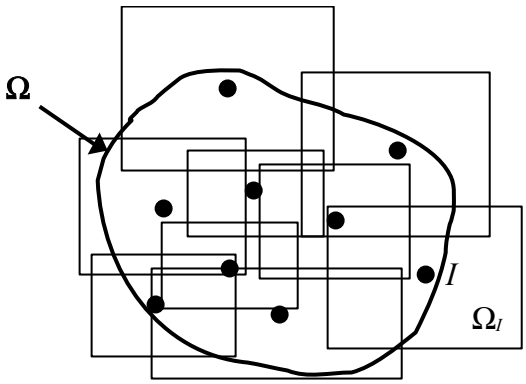



***Fig. 1.2 : Exemple de recouvrement 2D d'un domaine Ω : supports rectangulaires.***

Nous allons à présent présenter dans les sections §1.3 à §1.6, le formalisme des différentes familles de méthodes meshfree. Pour ce faire, nous noterons $u^h(\underline{x})$, l'approximation discrète d'une fonction $u(\underline{x})$ définie sur $\overline{\Omega}$, cette fonction pouvant être scalaire, vectorielle ou tensorielle. Pour chacune des familles de méthodes meshfree, nous exprimerons $u^h(\underline{x})$ de la manière générale suivante :

$$u^h(\underline{x}) = \sum_{I=1}^{N} \phi_I(\underline{x})\, d_I \tag{1.3}$$

où les fonctions $\phi_I(\underline{x})$ représentent les fonctions de forme à expliciter et où $\underline{d} = \{d_I\}_{I=1}^{N}$ représente le vecteur des paramètres nodaux associés à l'approximation sur $\overline{\Omega}$.

Nous utiliserons également le vecteur $\underline{p}^T(\underline{x}) = \{p_i(\underline{x})\}_{i=1}^{m}$, comme vecteur de base ou *base primaire* de l'approximation (appellation initialement utilisée par la méthode EFGM). Ce

vecteur est constitué de *m* fonctions $p_i(\underline{x})$, le plus souvent des monômes, calculées au point de coordonnées $\underline{x}$.

Dans le cas où d'une base polynomiale, nous noterons $\tilde{m}$, l'ordre de la base $\underline{p}(\underline{x})$.

Par exemple, pour le cas d'une base polynomiale monodimensionnelle, le terme $p_i(\underline{x})$, *i* variant de *1* à *m*, représente le monôme d'ordre *i* : $p_i(x) = x^i$, et le nombre de termes *m* de la base est simplement donné par l'égalité : $m = \tilde{m} + 1$.

Dans le cas bidimensionnel, l'utilisation du triangle de Pascal (*cf.* Fig. 1.3) permet de déterminer facilement l'expression de tous les termes de la base :

$$\begin{cases} p_i(x) = x^k y^{j-k} & 0 \le j+k \le \tilde{m},\ 0 \le i \le \frac{1}{2}(j+1)(j+2) \\ m = \frac{1}{2}(\tilde{m}+1)(\tilde{m}+2) \end{cases} \tag{1.4}$$

L'extension au cas tridimensionnel ne pose pas plus de difficultés.

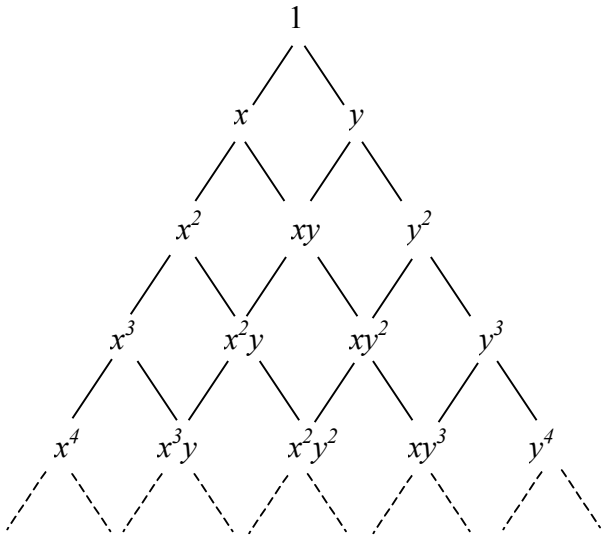



***Fig. 1.3 : Le triangle de Pascal (cas bidimensionnel).***

Il est également possible dans certains cas, de compléter la base polynomiale avec des fonctions non polynomiales, afin d'améliorer l'approximation vis-à-vis du problème que l'on cherche à modéliser. Cette technique d'enrichissement de la base d'approximation est notamment employée lorsqu'il existe des singularités (fracture, etc.) au niveau du domaine d'étude, et plus généralement dans le cas des domaines non convexes (voir par exemple [BEL 96 ; ORG 96]). Elle s'inspire de techniques utilisées dans la méthode des éléments finis pour représenter des champs singuliers [BEN 74 ; ZIE 91]. Par exemple, lorsque l'expression analytique du champ solution cherché est connue, la base est enrichie avec des fonctions apparaissant dans la forme analytique. Pour notre part, nous limiterons nos travaux à l'étude des domaines convexes.

Nous allons à présent rappeler dans les sections suivantes, le principe et les propriétés des principales familles de méthodes meshfree, en mettant en évidence les spécificités de chacune vis-à-vis de la construction des fonctions de forme et leur lien avec le principe de la partition de l'unité. Ceci nous amènera en fin de chapitre, à établir un tableau synthétique de ces différentes méthodes.

## 1.3 METHODES BASEES SUR L'APPROXIMATION PAR MOINDRES CARRES MOBILES

Contrairement au lissage par moindre carrés classique, l'approximation *MLS* introduit une dépendance des coefficients du lissage vis-à-vis des points du domaine $\overline{\Omega}$ où est réalisée l'approximation, ce qui, bien qu'étant une approximation globale, lui confère un caractère local (i.e. le terme « mobile »). Dans la suite de cette section, nous rappelons le principe de construction ainsi que les propriétés des fonctions de forme pour les méthodes basées sur ce type d'approximation.

### 1.3.1 Principe de l'approximation *MLS*

#### *1.3.1.1 Détermination des fonctions de forme*

Tout d'abord, la construction des fonctions de forme pour les méthodes reposant sur l'approximation aux moindres carrés mobiles se caractérise par l'utilisation d'une fonction monotone décroissante *W*, appelée fonction de pondération pour la méthode EFGM, définie sur le même support que la fonction de forme et ayant les propriétés suivantes :

$$\forall I, W_I\left(\underline{x}\right) \in C^s\left(\Omega_I\right), s \in \mathbb{N} \tag{1.5}$$

$$W_I\left(\underline{x}\right)\begin{cases} > 0 & \text{si } \underline{x} \in \Omega_I \\ = 0 & \text{si } \underline{x} \notin \Omega_I \end{cases} \tag{1.6}$$

$$\begin{cases} W_I\left(\underline{x}\right) = W\left(r\right) \\ r = r\left(\underline{x} - \underline{x}_I, \underline{h}_I\right) \end{cases} \tag{1.7}$$

La condition (1.6) est cruciale pour ce type d'approximation car elle permet en fait de construire une approximation globale sur le domaine à partir d'une représentation locale [LAN 81 ; AUB 97].

Dans le cas de fonctions à support radial ou elliptique, on calcule la fonction $r = \left\|\underline{r}\right\|_2$ en prenant pour composantes de $\underline{r}$ : $r_j = \left|x_j - x_{Ij}\right| / h_{Ij}$ pour $j = 1, n$. Dans le cas purement radial, on a en fait : $h_{Ij} = h_I$ et $r = \left\|\underline{x} - \underline{x}_I\right\|_2 / h_I$ .

Lorsque le support est de type rectangulaire, une façon simple de définir la fonction de pondération associée est de calculer le produit des fonctions suivant les différentes directions de l'espace (ou par rapport à chaque composante $h_{Ij}$ de $\underline{h}_I$ ) :

$$W\left(r\right) = \prod_{j=1}^{n} W\left(r_j\right) \tag{1.8}$$

Dans l'approximation *MLS*, la solution approchée $u^h(\underline{x})$ en tout point $\underline{x}$ du domaine $\overline{\Omega}$ s'exprime par la relation suivante :

$$u^h(\underline{x}) = \underline{p}^T(\underline{x}) \cdot \underline{\alpha}(\underline{x}) \tag{1.9}$$

Le caractère local de l'approximation provient du fait que le vecteur des coefficients $\underline{\alpha}$ dépendent de $\underline{x}$, contrairement au lissage classique.

La détermination de $\underline{\alpha}$ s'effectue ensuite de manière habituelle, en minimisant la norme pondérée discrète $J(\underline{\alpha})$, représentant le carré de la distance entre l'approximation en tout point $\underline{x}$ de $\overline{\Omega}$ et l'ensemble des valeurs aux nœuds :

$$J(\underline{\alpha})\big|_{\underline{x}} = \sum_{I=1}^{N} W_I(\underline{x}) \left[ \underline{p}^T(\underline{x}_I) \cdot \underline{\alpha}(\underline{x}) - d_I \right]^2 \tag{1.10}$$

Si l'on note :

$$\mathbf{P} = \begin{bmatrix} p_1(\underline{x}_1) & \cdots & p_m(\underline{x}_1) \\ \vdots & \vdots & \vdots \\ p_1(\underline{x}_N) & \cdots & p_m(\underline{x}_N) \end{bmatrix}_{N \times m} \tag{1.11}$$

et également :

$$\mathbf{W}(\underline{x}) = \begin{bmatrix} W_1(\underline{x}) & & 0 \\ & \ddots & \\ 0 & & W_N(\underline{x}) \end{bmatrix}_{N \times N} \tag{1.12}$$

alors, minimiser (1.10) revient à calculer :

$$\left. \frac{\partial J(\underline{\alpha})}{\partial \underline{\alpha}} \right|_{\underline{x}} = \mathbf{M}(\underline{x}) \cdot \underline{\alpha}(\underline{x}) - \mathbf{P}^T \cdot \mathbf{W}(\underline{x}) \cdot \underline{d} = 0 \tag{1.13}$$

où la matrice **M**, encore appelée matrice des moments, est définie par la relation :

$$\mathbf{M}(\underline{x}) = \mathbf{P}^T \cdot \mathbf{W}(\underline{x}) \cdot \mathbf{P} \tag{1.14}$$

La résolution du système (1.13) permet de déterminer l'expression des composantes de $\underline{\alpha}(\underline{x})$ et donc celle des fonctions de forme, en identifiant (1.9) et (1.3) terme à terme :

$$\underline{\phi}^T(\underline{x}) = \left\{ \phi_I(\underline{x}) \right\}_{I=1}^{N} = \underline{p}^T(\underline{x}) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \mathbf{P}^T \cdot \mathbf{W}(\underline{x}) \tag{1.15}$$

Cette résolution suppose néanmoins que l'inverse de la matrice **M** existe, ce qui se traduit par la condition nécessaire suivante :

$$\forall \underline{x} \in \overline{\Omega}, N_{\underline{x}} \geq m \tag{1.16}$$

où $N_{\underline{x}}$ représente le nombre de nœuds dont le voisinage contient $\underline{x}$. Pour que la condition (1.16) devienne suffisante, il est notamment nécessaire de vérifier que l'arrangement spatial des nœuds voisins permet de déterminer complètement les termes de la base primaire. Par exemple (*cf*. Fig. 1.4), une disposition en ligne de tous les nœuds voisins de $\underline{x}$, associés à une base linéaire ($\widetilde{m}=1$) dans le cas bidimensionnel, entraîne une singularité de $\boldsymbol{A}$, bien que (1.16) soit vérifiée.

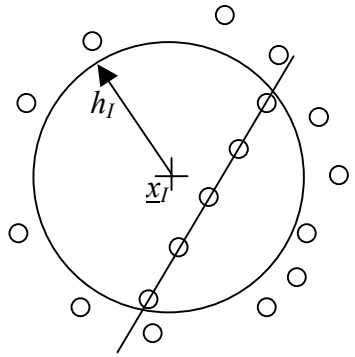



***Fig. 1.4 : Arrangement nodal 2D singulier pour la base*** $\underline{p}^T(\underline{x}) = \{1, x, y\}$**.**

En posant :

$$\underline{b}_I(\underline{x}) = W_I(\underline{x})\,\underline{p}(\underline{x}_I) \tag{1.17}$$

on peut alors expliciter la fonction de forme au nœud *I* comme suit :

$$\phi_I(\underline{x}) = \sum_{i=1}^{m} p_i(\underline{x}) \left[\mathbf{M}^{-1}(\underline{x})\right]_{ij} b_{Ij}(\underline{x}) \tag{1.18}$$

L'expression (1.18) est commune aux méthodes EFGM et DEM. Les spécificités de ces deux méthodes se situent plutôt au niveau du calcul des dérivées et des hypothèses faites quant à l'imposition des conditions aux limites essentielles (Dirichlet) ou encore sur les propriétés interpolantes des fonctions de forme (*cf*. §1.3.2).

### *1.3.1.2 Cas particulier de la méthode meshfree h-p Clouds*

La construction des fonctions de forme h-p Clouds, telle que présentée par Duarte et Oden [DUA 95 ; DUA 96a ; DUA 96b], repose sur l'utilisation des fonctions *MLS* (1.18), associée à une base de fonctions complémentaire, appelée base *secondaire*. La définition de cette base est propre à chaque nœud, conférant ainsi à l'approximation des possibilités intéressantes d'enrichissement local, essentiellement de type $p^1$ et facilement adaptable au problème traité.

Soit le vecteur de base $\underline{L}_I^T(\underline{x}) = \{L_{Ii}(\underline{x})\}_{i=1}^{m_I}$, associé au nœud $I$, et constitué de $m_I$ fonctions $L_{Ii}(\underline{x})$, calculées au point de coordonnées $\underline{x}$ et correspondant le plus souvent à des polynômes. Il est néanmoins intéressant dans certains problèmes, de choisir des fonctions spécifiques (harmoniques, etc.) permettant d'obtenir un enrichissement local de l'approximation.

Le champ approché $u^h(\underline{x})$ peut alors s'écrire de la manière générale suivante :

$$u^h(\underline{x}) = \sum_{I=1}^{N} \phi_I(\underline{x}) \left( d_I + \sum_{k=1}^{m_I} L_{Ik}(\underline{x})\, d_{Ik} \right) \tag{1.19}$$

Les paramètres $d_{Ik}$ correspondent à des contributions nodales complémentaires spécifiquement associées au nœud $I$.

Dans le cas d'une base secondaire formée de monômes, l'ordre de la base sera noté $\tilde{m}_I$ et vérifiera l'inégalité : $\tilde{m} < \tilde{m}_I$ (*cf.* exemple Fig. 1.5). On voit d'après (1.19) que le cas particulier : $\tilde{m} = \tilde{m}_I$ correspond à la méthode EFGM classique ($\underline{L}_I = \underline{0}$). Le nombre de paramètres additionnels se calculent simplement par la relation :

$$m_I = \tfrac{1}{2}(\tilde{m}_I + 1)(\tilde{m}_I + 2) - m \tag{1.20}$$

où $m$ est donné en (1.4).

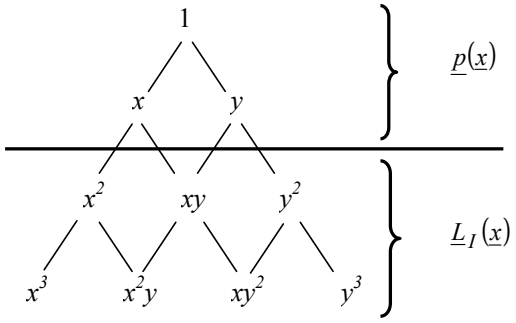



***Fig. 1.5 : Bases polynomiales primaire et secondaire pour la méthode h-p Clouds ($\tilde{m} = 1, \tilde{m}_I = 3$).***

[1] Deux techniques d'enrichissement sont classiquement utilisées en éléments finis afin d'améliorer l'approximation numérique. En effet, suivant le type de raffinement recherché (local ou global), l'enrichissement de type *p* consiste à modifier l'expression de certaines fonctions de forme, tandis que celui de type *h* consiste à augmenter le nombre de points de la discrétisation.

### *1.3.1.3 Fonctions de pondération usuelles*

Nous rappelons ci-après les fonctions de pondération $W(r), r \leq 1$ à support radial, les plus couramment utilisées en pratique.

***Fonctions de Gauss (ou exponentielles) :***

*Gauss1 :*

$$W(r) = e^{-(r/c_I)^{2k}}, k \geq 1, c_I > 0 \tag{1.21}$$

*Gauss2 :*

$$W(r) = \frac{e^{-(r\,h_{max})^{2k}} - e^{-(h_{max})^{2k}}}{1 - e^{-(h_{max})^{2k}}} \tag{1.22}$$

en posant dans ce cas :

$$h_I = c_I h_{max} \tag{1.23}$$

$h_{max}$ représente le rayon maximale possible pour tous les domaines d'influence. Le coefficient $c_I$ est représentatif de la distance minimale calculée à partir du nœud *I*, pour que la matrice ***A*** associée ne soit pas singulière. En pratique, $c_I$ est généralement pris égal à la distance du nœud *I* au second ou troisième plus proche voisin.

***Fonction conique :***

$$W(r) = 1 - r^k, k \geq 1 \tag{1.24}$$

***Fonction sinusoïdale :***

$$W(r) = \frac{1}{2}\left(1 + Cos(\pi r)\right) \tag{1.25}$$

***Fonctions splines cubiques :***

*Spline1 :*

$$W(r) = \begin{cases} 1 - 6r^2 + 6r^3 & \text{si } r \leq \frac{1}{2} \\ 2(1-r)^3 & \text{si } \frac{1}{2} < r \leq 1 \end{cases} \tag{1.26}$$

*Spline2 :*

$$W(r)=\begin{cases}\frac{2}{3}-4r^2+4r^3 & \text{si } r\le\frac{1}{2}\\ \frac{4}{3}(1-r)^3 & \text{si } \frac{1}{2}<r\le 1\end{cases} \tag{1.27}$$

***Fonction polynomiale d'ordre 4 :***

$$W(r)=1-6r^2+8r^3-3r^4 \tag{1.28}$$

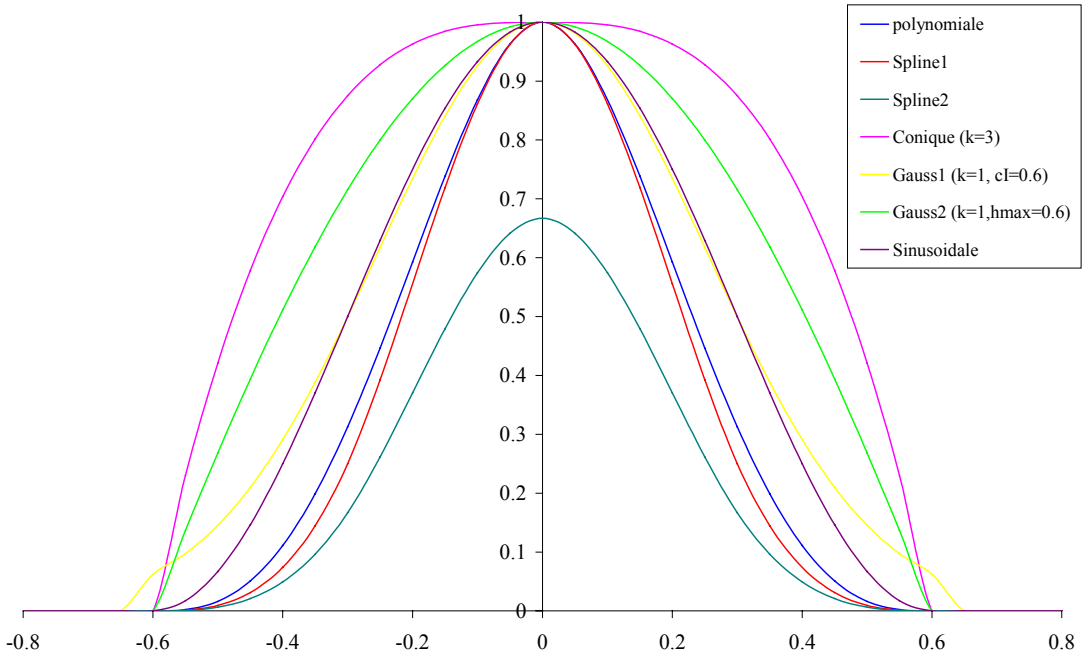



***Fig. 1.6 : Fonctions de pondération usuelles.***

*1.3.1.4 Exemples de fonctions de forme MLS*

Les fonctions de forme *MLS* sont généralement rationnelles et difficiles à exprimer de manière explicite. Certains auteurs en fournissent une expression dans des cas très particuliers. Des exemples 1D et 2D sont fournis sur les figures 7 à 9.

Par exemple, les fonctions les plus simples sont celles proposées par Shepard [SHE 68], qui utilisent une base primaire constante ( $\tilde{m}=0$ ) :

$$\phi_I(\underline{x})=\frac{W_I(\underline{x})}{\sum_{J=1}^{N} W_J(\underline{x})} \tag{1.29}$$

De manière triviale, des fonctions de forme ainsi construites vérifient les conditions (1.1), (1.2), et forment donc une partition de l'unité.

Breitkopf et co-auteurs [BRE 00] fournissent également l'expression des fonctions de forme 1D, 2D et 3D, ainsi que leurs dérivées, pour le cas d'une base primaire linéaire :

***Cas 1D :***

$$\phi_I(x) = \frac{W_I(x)}{D_1(x)} \sum_{J=1}^{N} (x_J - x_I)(x_J - x) W_J(x) \tag{1.30}$$

en posant :

$$D_1(x) = \sum_{J=1}^{N-1} W_J(x) \sum_{K=J+1}^{N} (x_K - x_J)^2 W_K(x) \tag{1.31}$$

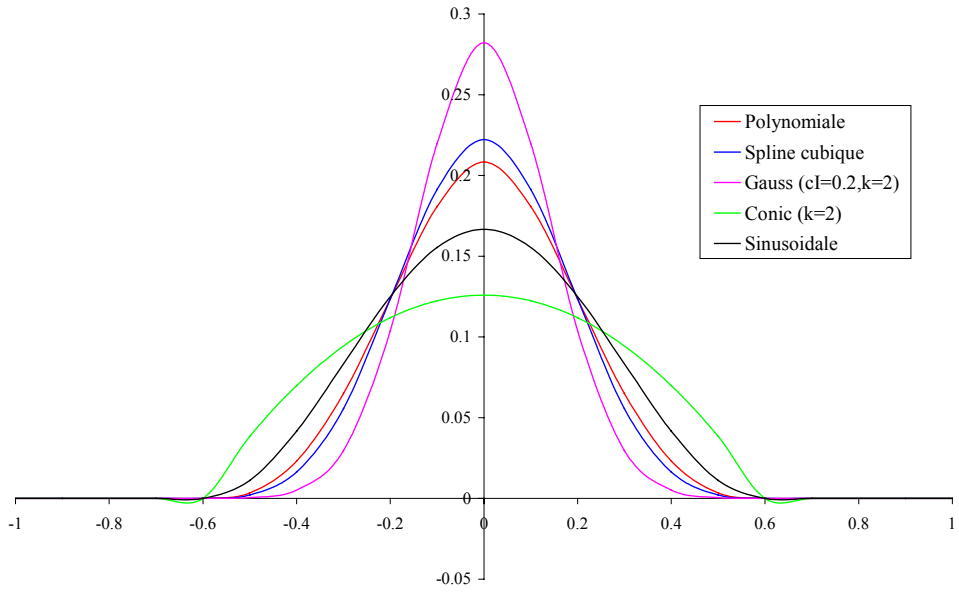



***Fig. 1.7 : Fonctions de forme EFGM 1D pour différentes fonctions de pondération usuelles (base primaire constante, $h_{max}$ = 0.6).***

***Cas 2D :***

$$\phi_I(\underline{x}) = \frac{W_I(\underline{x})}{D_2(\underline{x})} \sum_{J \neq I} W_J(\underline{x}) \sum_{\substack{K \neq I, \\ K > J}} W_K(\underline{x}) \Theta_2(\underline{x}, \underline{x}_J, \underline{x}_K) \Theta_2(\underline{x}_I, \underline{x}_J, \underline{x}_K) \tag{1.32}$$

en posant :

$$D_2(\underline{x}) = \sum_{I=1}^{N-2} W_I(\underline{x}) \sum_{J=I+1}^{N-1} W_J(\underline{x}) \sum_{K=J}^{N} [\Theta_2(\underline{x}_I, \underline{x}_J, \underline{x}_K)]^2 W_K(\underline{x}) \tag{1.33}$$

et :

$$\Theta_2\left(\underline{x}_a,\underline{x}_b,\underline{x}_c\right)=y_a\left(x_c-x_b\right)+y_b\left(x_c-x_a\right)+y_c\left(x_b-x_a\right) \tag{1.34}$$

***Cas 3D :***

$$\phi_I\left(\underline{x}\right)=\frac{W_I\left(\underline{x}\right)}{D_3\left(\underline{x}\right)}\sum_{J\neq I}W_J\left(\underline{x}\right)\sum_{\substack{K\neq I,\\K>J}}W_K\left(\underline{x}\right)\sum_{\substack{L\neq I,\\L>K}}W_L\left(\underline{x}\right)\Theta_3\left(\underline{x},\underline{x}_J,\underline{x}_K,\underline{x}_L\right)\Theta_3\left(\underline{x}_I,\underline{x}_J,\underline{x}_K,\underline{x}_L\right) \tag{1.35}$$

en posant :

$$D_3\left(\underline{x}\right)=\sum_{I=1}^{N-3}W_I\left(\underline{x}\right)\sum_{J=I+1}^{N-2}W_J\left(\underline{x}\right)\sum_{K=J+1}^{N-1}W_K\left(\underline{x}\right)\sum_{L=K}^{N}W_L\left(\underline{x}\right)\left[\Theta_3\left(\underline{x}_I,\underline{x}_J,\underline{x}_K,\underline{x}_L\right)\right]^2 \tag{1.36}$$

et :

$$\begin{aligned}\Theta_3\left(\underline{x}_a,\underline{x}_b,\underline{x}_c,\underline{x}_d\right)=z_a\,\Theta_2\left(x_b,x_c,x_d\right)+z_b\,\Theta_2\left(x_a,x_c,x_d\right)\\+z_c\,\Theta_2\left(x_a,x_b,x_d\right)+z_d\,\Theta_2\left(x_a,x_b,x_c\right)\end{aligned} \tag{1.37}$$

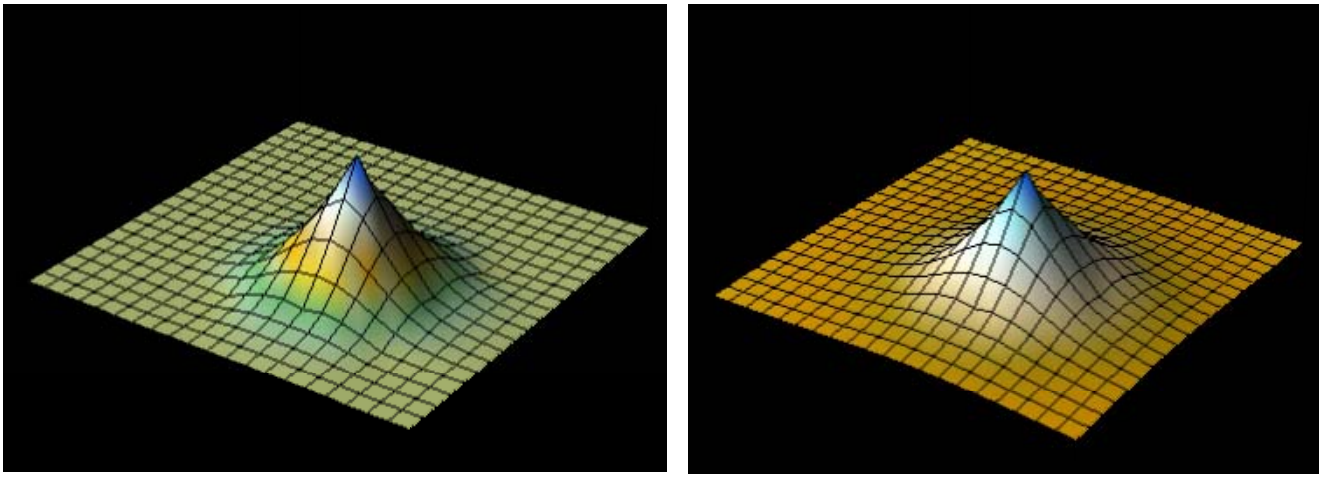


*Fig. 1.8 : Fonction de pondération polynomiale 2D et fonction de forme EFGM associée (base primaire linéaire, $h_{max} = 0.3$).*

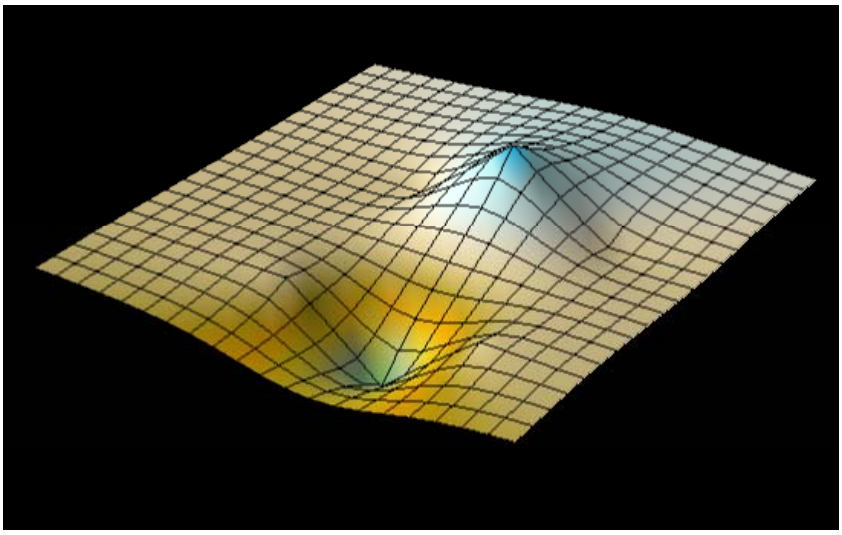


*Fig. 1.9 : Dérivée partielle d'une fonction de forme EFGM (fonction de pondération polynomiale, base primaire linéaire, $h_{max} = 0.3$).*

## 1.3.2 Propriétés des fonctions de forme *MLS*

### *1.3.2.1 Consistance*

L'ordre de consistance de l'approximation $u^h(\underline{x})$ dépend directement de celui des équations aux dérivées partielles (EDP) du problème à résoudre. Ainsi, pour des EDP d'ordre 2*k*, l'ordre de l'approximation devra être au moins égal à *k*, c'est-à-dire que la relation suivante doit être vérifiée :

$$u^h(\underline{x}) - u(\underline{x}) = \mathcal{O}\left(h^k\right) \tag{1.38}$$

où *h* est représentatif de l'arrangement nodal utilisé pour la discrétisation spatiale. Cette condition exprime le fait que la solution cherchée $u(\underline{x})$ peut être reproduite exactement par un polynôme d'ordre *k*, qui dans le cas des méthodes meshfree, correspond à l'ordre $\tilde{m}$ de la base primaire polynomiale :

$$\forall \underline{x} \in \overline{\Omega}, \quad \sum_{I=1}^{N} \phi_I(\underline{x})\, p_j(\underline{x}_I) = p_j(\underline{x}) \quad 1 \le j \le m \tag{1.39}$$

On montre donc aisément que les fonctions de forme *MLS* permettent notamment de vérifier la consistance d'ordre zéro, ce qui se traduit par ( $p_1(\underline{x}) = 1$ ) :

$$\sum_{I=1}^{N} \phi_I(\underline{x}) 1 = 1 \tag{1.40}$$

On retrouve ainsi la condition (1.2) : les fonctions de forme *MLS* constituent donc une partition de l'unité.

### *1.3.2.2 Non interpolation et conditions aux limites de Dirichlet*

D'une manière générale et contrairement à la méthode des éléments finis, les fonctions de forme (1.18) (méthodes meshfree basées sur l'approximation *MLS*) ne sont pas interpolantes, ce qui se traduit par la relation :

$$\phi_I(\underline{x}_J) \neq \delta_{IJ} \tag{1.41}$$

Par conséquent, les valeurs du champ approché $u^h$ aux nœuds, c'est-à-dire : $u_I = u^h(\underline{x}_I)$, et les contributions nodales $d_I$ ne sont pas égales (*cf.* Fig. 1.10) :

$$\forall I, u_I \neq d_I \tag{1.42}$$

Cette non interpolation des fonctions de forme entraîne un non respect des conditions aux limites essentielles (Dirichlet). Cependant pour la méthode DEM, les conditions aux limites essentielles sont imposées au sens fort comme pour le méthode des éléments finis, c'est-à-dire pour tout $\underline{x}_I$ de la frontière concernée $\Gamma_u$, $d_I = \overline{u}(\underline{x}_I)$, ce qui entraîne des erreurs numériques au niveau de la solution calculée [AUB 97].

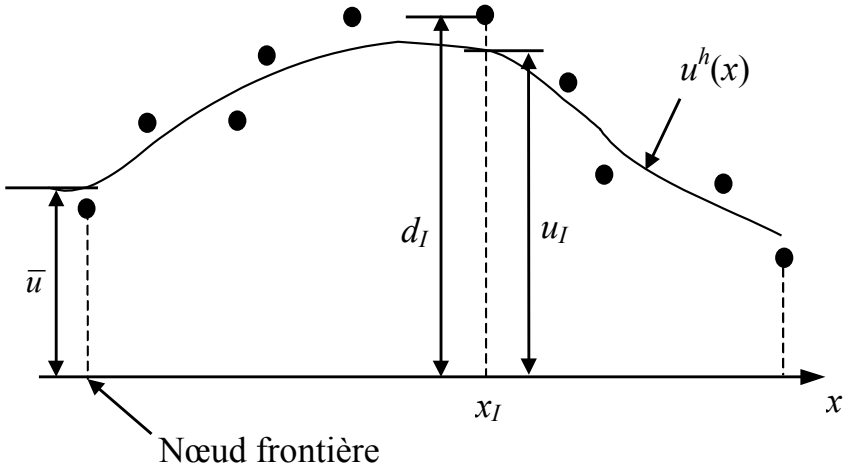



***Fig. 1.10 : Distinction entre valeurs et contributions nodales (1D).***

Différentes méthodes spécifiques sont donc proposées dans la littérature pour résoudre ce problème, les plus usuelles et les plus utilisées en pratique étant :

- l'ajout de multiplicateurs de Lagrange [BEL 95] ;
- l'élimination du multiplicateur de Lagrange par modification de la formulation variationnelle [LU 94] ;
- l'ajout d'une fonction de pénalisation à la formulation variationnelle [ZIE 91 ; LAO 96 ; GAV 00] ;
- l'utilisation de fonctions de pondération singulières [LAN 81 ; DUA 96b ; LAO 96 ; BRE 00] ;
- le couplage avec des éléments finis au niveau de la frontière [BEL 95] .

Une synthèse des deux premières méthodes, ainsi que des avantages et inconvénients de chacune peuvent notamment être trouvés dans [AUB 97]. La troisième méthode est déjà couramment utilisée en éléments finis.

Nous allons présenter plus particulièrement ici les deux dernières méthodes, pour lesquelles les performances numériques sont comparées dans les applications du chapitre 2.

***Fonctions de pondération singulières :***

Cette technique consiste à rendre les fonctions de forme interpolantes en utilisant des fonctions de pondération discontinues, notées $\widetilde{W}_I(\underline{x})$, telles que : $\forall I, \widetilde{W}_I(\underline{x}) \xrightarrow[\underline{x}\to\underline{x}_I]{} +\infty$, et uniquement pour les nœuds sur la frontière. Par exemple, Laouar [LAO 96] propose d'utiliser la forme suivante :

$$\widetilde{W}_I(\underline{x}) = \frac{W_I(\underline{x})}{1 - W_I(\underline{x})} \quad \text{si } \underline{x} \neq \underline{x}_I \tag{1.43}$$

Au nœud $\underline{x}_I$ sur la frontière, la fonction de forme est alors calculée directement et vérifie la condition d'interpolation : $\phi_I(\underline{x}_J) = \delta_{IJ}$.

Récemment, Breitkopf et co-auteurs [BRE 00] ont proposé d'utiliser des fonctions de pondération modifiées calculées en normalisant avec une interpolation de type Shepard :

$$\tilde{W}_I(\underline{x}) = \frac{W_I(\underline{x})}{W_I(\underline{x}) + \left[1 - W_I(\underline{x})\right] \sum_{J \neq I} \dfrac{W_J(\underline{x})}{\left[1 - W_J(\underline{x})\right]}} \tag{1.44}$$

On voit que, contrairement à la technique utilisée en (1.43), les fonctions de pondération ainsi construites sont continues sur leur support et vérifient par ailleurs :

$$\begin{cases} \tilde{W}_I(\underline{x}) \in [0,1] \\ \sum_{I=1}^{N} \tilde{W}_I(\underline{x}) = 1 \\ \forall I, \tilde{W}_I(\underline{x}_J) = \delta_{IJ} \end{cases} \tag{1.45}$$

Cette technique est intéressante, car du fait des propriétés de continuité des fonctions de pondération modifiées, leur utilisation est possible pour *tous les nœuds* du domaine, et non pas uniquement sur la frontière, rendant l'approximation vraiment interpolante.

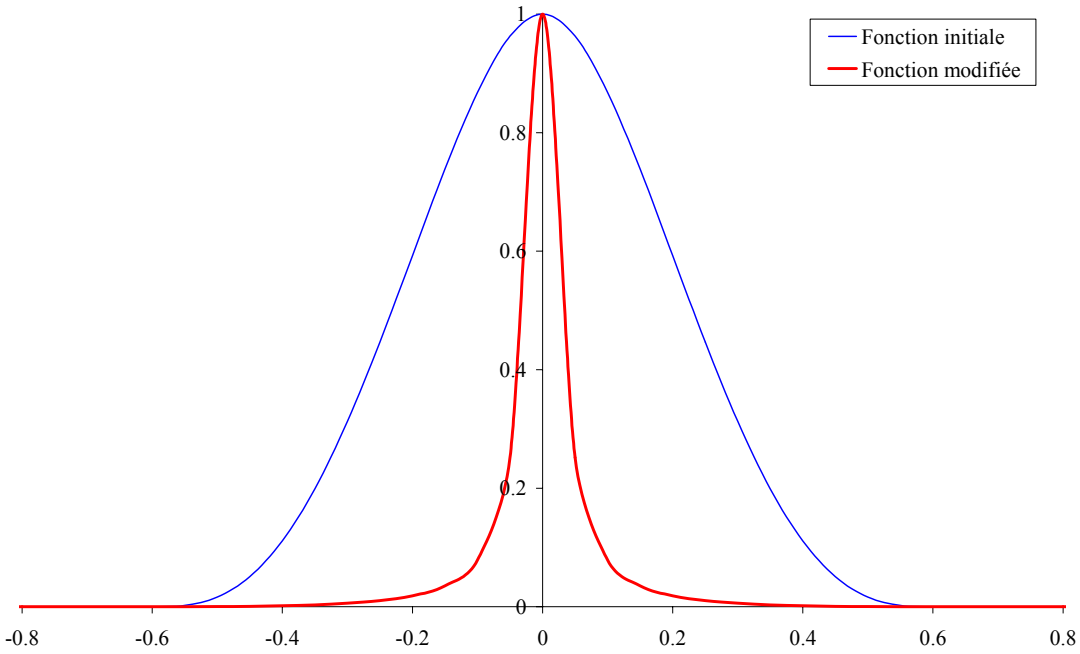



***Fig. 1.11 : Fonction de pondération polynomiale initiale et fonction modifiée proposée par Breitkopf.***

***Couplage avec des éléments finis au niveau de la frontière :***

Une approche différente de couplage EFGM-FEM a été proposée par Aubert [AUB97], dans laquelle l'un des champs inconnus (déplacements) est approximé par la méthode des éléments finis et les autres champs (pressions de fluides), par la méthode EFGM.

Cette technique, présentée par l'équipe de Belytschko [BEL 95 ; BEL 96 ; KRO 96], consiste à introduire une rangée d'éléments à l'interface entre un sous-domaine pour lequel une approximation meshfree, en l'occurrence la méthode EFGM, est utilisée, et un sous-domaine de type éléments finis (FEM). Notons $\Omega_M$, $\Omega_I$ et $\Omega_F$, les sous-domaines meshfree, interface et éléments finis et définissons $\Gamma_M$, la frontière entre $\Omega_M$ et $\Omega_I$, et $\Gamma_F$, celle entre $\Omega_I$, $\Omega_F$ (*cf.* Fig. 1.12 et Fig. 1.13).

Le champ approché $\underline{u}^h(\underline{x})$ est alors évalué au niveau d'un élément d'interface $\Omega_I^e$, en utilisant l'approximation mixte suivante :

$$\begin{aligned} \forall \underline{x} \in \Omega_I^e,\ \underline{u}^h(\underline{x}) &= \left[1 - R(\underline{x})\right] \sum_{I=1}^{N_F} N_I(\underline{x})\, \underline{d}_I + R(\underline{x}) \sum_{I=1}^{N} \phi_I(\underline{x})\, \underline{d}_I \\ &= \sum_{I=1}^{N} \tilde{N}_I(\underline{x})\, \underline{d}_I \end{aligned} \tag{1.46}$$

$N_F$ et $N_I(\underline{x})$ représentent respectivement le nombre de nœuds et les fonctions de forme de l'élément $\Omega_I^e$ en tant qu'élément fini ; $N$ est le nombre de nœuds meshfree et/ou FEM utilisés

pour construire les fonctions de forme meshfree, c'est-à-dire dont le voisinage contient le point d'évaluation $\underline{x}$. $\tilde{N}_I(\underline{x})$ est la fonction de forme mixte au niveau de $\Omega_I^e$, que l'on peut également définir par l'expression :

$$\tilde{N}_I(\underline{x}) = \begin{cases} \left[1 - R(\underline{x})\right] N_I(\underline{x}) + R(\underline{x})\,\phi_I(\underline{x}) & \underline{x}_I \in \Omega_I^e \\ R(\underline{x})\,\phi_I(\underline{x}) & \underline{x}_I \notin \Omega_I^e \end{cases} \tag{1.47}$$

$R(\underline{x})$ est une fonction de transition introduite par les auteurs, de façon à vérifier :

$$R(\underline{x}) = \begin{cases} 1 & \underline{x} \in \Gamma_M \\ 0 & \underline{x} \in \Gamma_F \end{cases} \tag{1.48}$$

et définie simplement par :

$$R(\underline{x}) = \sum_{\substack{J \\ \underline{x}_J \in \Gamma_M}} N_J(\underline{x}) \tag{1.49}$$

Dans ce cas, la dérivée $\tilde{N}_{I,i}$ de $\tilde{N}_I$ au point $\underline{x}$ et par rapport à la direction spatiale $i$ s'écrit :

$$\tilde{N}_{I,i} = \begin{cases} -R_{,i}\, N_I + \left[1 - R\right] N_{I,i} + R_{,i}\,\phi_I + R\,\phi_{I,i} & \underline{x}_I \in \Omega_I^e \\ R_{,i}\,\phi_I + R\,\phi_{I,i} & \underline{x}_I \notin \Omega_I^e \end{cases} \tag{1.50}$$

avec :

$$R_{,i} = \sum_{\substack{J \\ \underline{x}_J \in \Gamma_M}} N_{J,i} \tag{1.51}$$

L'expression (1.50) montre clairement l'existence d'un saut au niveau de la dérivée $\tilde{N}_{I,i}$ à l'interface meshfree - FEM, du fait de la contribution de $R_{,i}$, dérivée de $R(\underline{x})$. De façon à diminuer le saut ainsi introduit, les auteurs proposent de remplacer $R(\underline{x})$ dans (1.47), par la fonction suivante :

$$\tilde{R}(\underline{x}) = R^2(\underline{x})\left[3 - 2R(\underline{x})\right] \tag{1.52}$$

De la même manière, $R_{,i}$ est remplacée dans (1.50) par :

$$\tilde{R}_{,i} = 6R\left[1 - R\right] R_{,i} \tag{1.53}$$

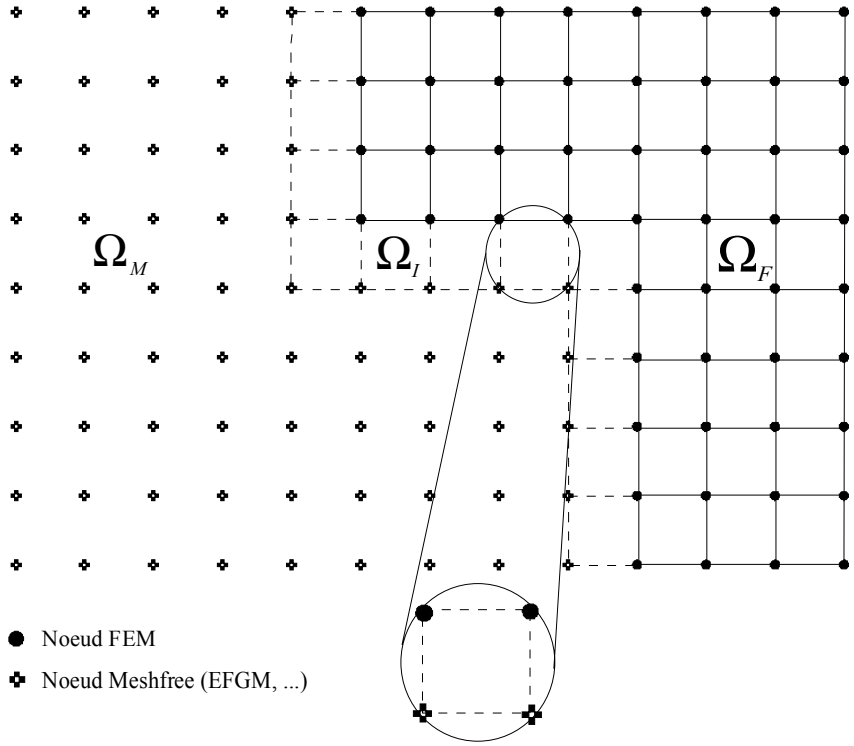



***Fig. 1.12 : Sous-domaines utilisés pour le couplage meshfree - FEM.***

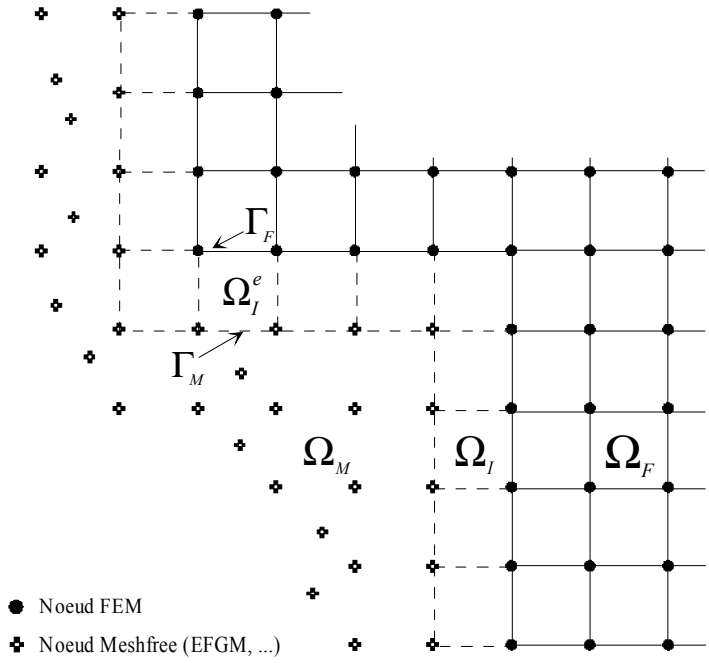



***Fig. 1.13 : Définition d'un élément d'interface $\Omega_I^e$.***

***Notre contribution :***

Pour imposer les conditions aux limites essentielles avec un couplage EFGM-FEM, Belytschko et co-auteurs utilisent des cellule d'intégration ou des éléments d'interface de même nature que les éléments finis à la frontière, à savoir par exemple des éléments quadrangulaires à 4 nœuds en 2D. Nous avons choisi de réaliser le couplage en introduisant à la frontière, des éléments finis de nature différente par rapport aux cellules d'intégration, à savoir par exemple des éléments linéiques en 2D (*cf*. Fig. 1.14 et voir les tests numériques du chapitre 2).

Remarques :

- Le calcul des fonctions de forme meshfree peut se faire soit en incluant les nœuds FEM dans le voisinage du point d'évaluation, soit en tronquant les supports au niveau de la frontière $\Gamma_F$. L'équipe de Belytschko montre que le fait d'exclure les nœuds FEM permet de réduire le saut au niveau des dérivées à l'interface. Les figures 13 et 14 montrent le cas 1D pour des fonctions de forme modifiées $\tilde{N}_I$ et leurs dérivées $\tilde{N}_{I,x}$ calculées en excluant les nœuds FEM et en utilisant les fonctions de transition $R(\underline{x})$ et $\tilde{R}(\underline{x})$.

- Une approche différente de couplage EFGM-FEM a été proposée par Modaressi et Aubert [MOD 95 ; MOD 96 ; AUB 97], dans laquelle le champ de déplacement est évalué par la méthode des éléments finis et les champs de pressions interstitielles (fluides mouillant et non-mouillant), par une méthode meshfree (EFGM ou DEM). Hormis le fait qu'elle fournit naturellement la structure d'intégration adaptée, cette technique offre notamment une plus grande souplesse quant au choix du nombre de degrés de liberté en pression, ce qui permet de mieux vérifier les critères de stabilité numérique pour les milieux poreux quasi-imperméables saturés par un fluide incompressible.

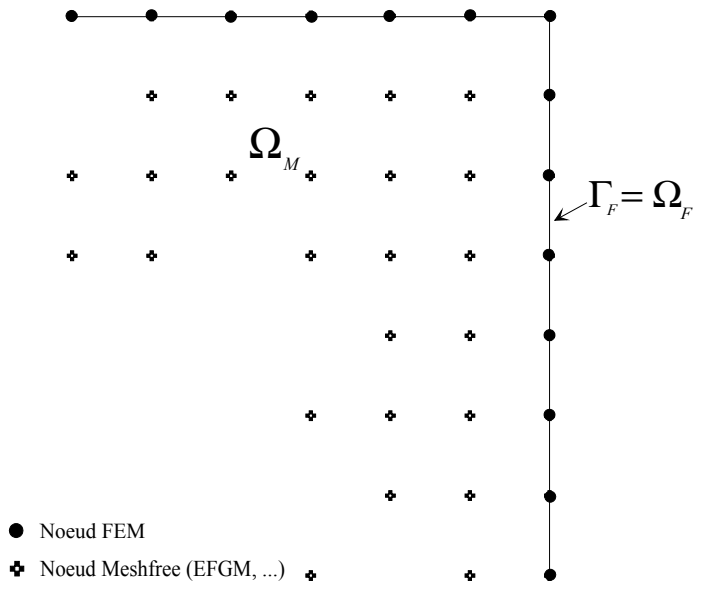



***Fig. 1.14 : Couplage meshfree - FEM à la frontière : utilisation d'éléments finis linéiques.***

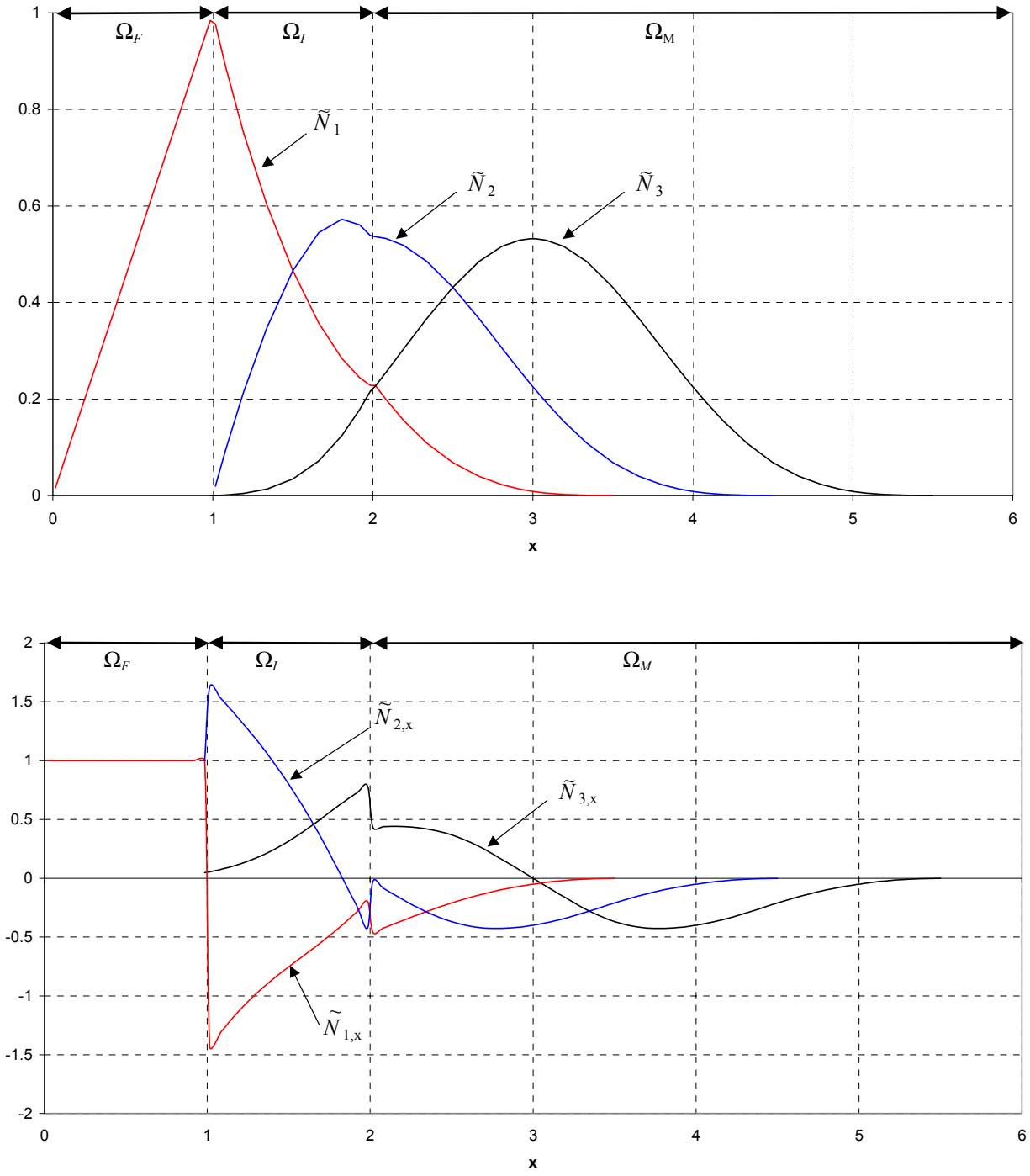



***Fig. 1.15 : Couplage EFGM-FEM 1D : calcul des fonctions de forme $\tilde{N}_I$ et des dérivées $\tilde{N}_{I,x}$ avec $R(\underline{x})$, sans les nœuds FEM (base primaire linéaire, spline1, $h_{max}$ = 2.5)***

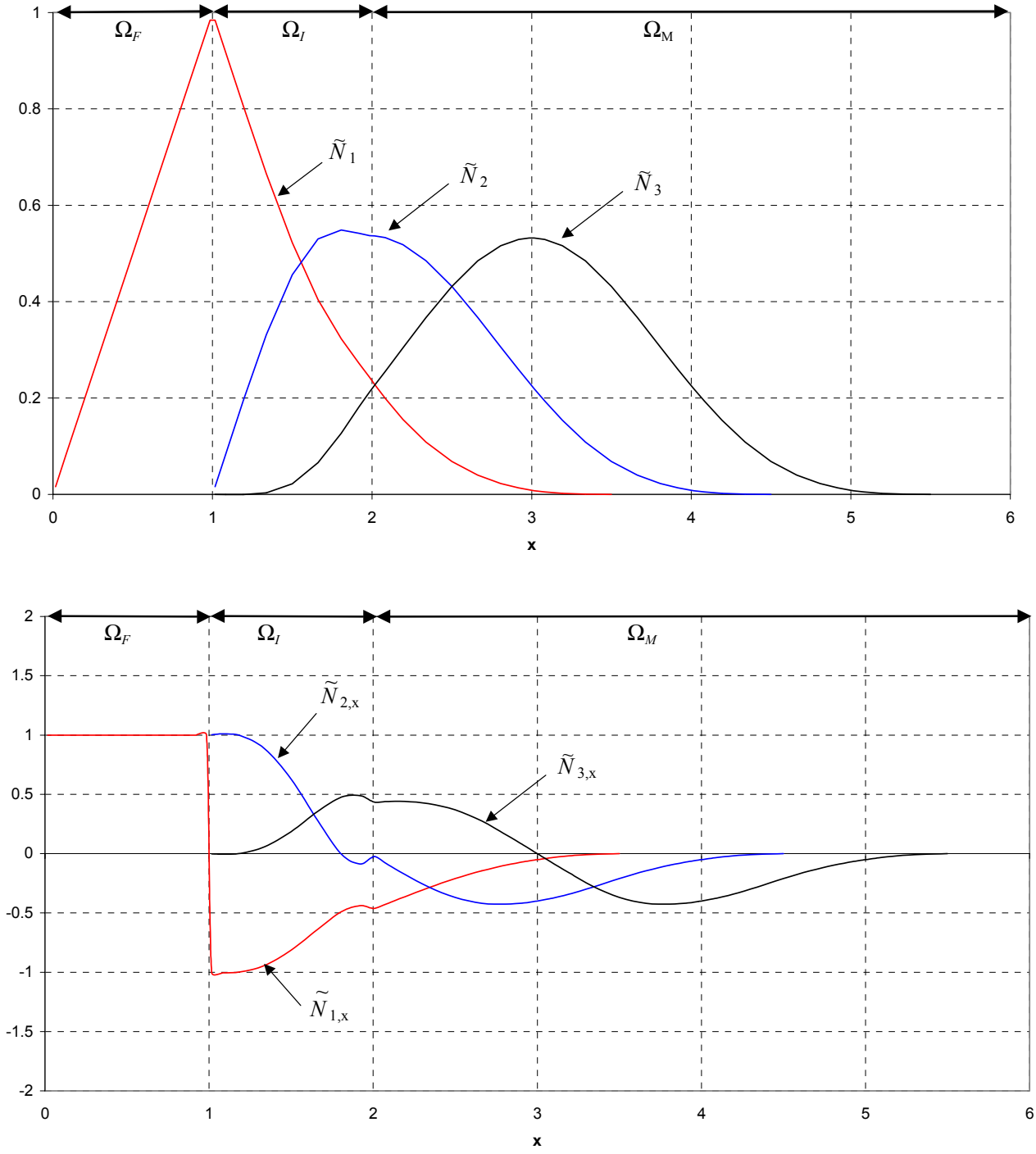



***Fig. 1.16 : Couplage EFGM-FEM 1D : calcul des fonctions de forme $\tilde{N}_I$ et des dérivées $\tilde{N}_{I,x}$ avec $\tilde{R}(\underline{x})$, sans les nœuds FEM (base primaire linéaire, spline1, $h_{max}$ = 2.5)***

### *1.3.2.3 Calcul rapide des dérivées partielles*

En pratique, les fonctions de forme *MLS* sont de continûment dérivable de classe $C^{Min(\tilde{m},s)}$ sur $\overline{\Omega}$, où d'après (1.5), *s* représente l'ordre de dérivabilité des fonctions de pondération. En exprimant les fonctions (1.18) sous la forme matricielle suivante :

$$\phi_I(\underline{x}) = \underline{p}^T(\underline{x}) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \underline{b}_I(\underline{x}) \tag{1.54}$$

où l'on rappelle que : $\underline{b}_I(\underline{x}) = W_I(\underline{x})\,\underline{p}(\underline{x}_I)$.

Les dérivées partielles $\partial_i \phi_I$ de $\phi_I$ suivant la direction *i* de l'espace, se calculent alors par :

$$\begin{aligned} \partial_i \phi_I(\underline{x}) = {} & \partial_i \underline{p}^T(\underline{x}) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \underline{b}_I(\underline{x}) + \underline{p}^T(\underline{x}) \cdot \partial_i \mathbf{M}^{-1}(\underline{x}) \cdot \underline{b}_I(\underline{x}) \\ & + \underline{p}^T(\underline{x}) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \partial_i \underline{b}_I \end{aligned} \tag{1.55}$$

avec :

$$\partial_i \mathbf{M}^{-1} = -\mathbf{M}^{-1} \cdot \partial_i \mathbf{M} \cdot \mathbf{M}^{-1} \tag{1.56}$$

On voit que le calcul matriciel (1.55) est très lourd (nombre d'opérations, et donc temps *CPU* élevés). Pour « alléger » ce calcul, différentes techniques ont été proposées dans la littérature.

La méthode DEM propose par exemple de calculer une dérivée incomplète, appelée dérivée diffuse, en ne conservant que le premier terme de l'expression (1.55) :

$$\partial_i \phi_I(\underline{x}) = \partial_i \underline{p}^T(\underline{x}) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \underline{b}_I(\underline{x}) \tag{1.57}$$

Cependant, Aubert a pu montré dans ses travaux que cette dérivation tronquée pouvait induire des erreurs importantes notamment au niveau du calcul de la matrice de rigidité du problème [AUB 97].

Belytschko et co-auteurs [BEL 96] proposent quant à eux, d'accélérer le calcul des dérivées partielles en utilisant une technique basée sur la propriété de consistance linéaire de l'approximation *MLS*. C'est cette technique que nous avons choisie d'adopter dans nos applications. En posant : $\underline{V}(\underline{x}) = \mathbf{M}^{-1}(\underline{x}) \cdot \underline{p}(\underline{x})$, l'on exprime les fonctions de forme de la manière suivante :

$$\phi_I(\underline{x}) = \underline{V}^T(\underline{x}) \cdot \underline{b}_I(\underline{x}) \tag{1.58}$$

Ceci revient à résoudre le système suivant :

$$\mathbf{M}(\underline{x}) \cdot \underline{V}(\underline{x}) = \underline{p}(\underline{x}) \tag{1.59}$$

Par ailleurs, si l'on choisit de résoudre le système (1.13) en calculant la décomposition $LU$ de $\mathbf{M}$ et non pas directement $\mathbf{M}^{-1}$, ceci nous permet alors de déterminer le vecteur cherché $\underline{V}$ en résolvant le système (1.59) de la manière suivante :

$$\begin{cases} \boldsymbol{L} \cdot \underline{Y} = \underline{p} \\ \boldsymbol{U} \cdot \underline{V} = \underline{Y} \end{cases} \tag{1.60}$$

On utilise également la même décomposition $LU$ de $\mathbf{M}$ pour calculer les dérivées partielles $\underline{V}_{,i}(\underline{x})$ pour $i = 1, n$, en résolvant le système dérivé :

$$\mathbf{M} \cdot \underline{V}_{,i} = \underline{p}_{,i} - \mathbf{M}_{,i} \cdot \underline{V} \tag{1.61}$$

Finalement, les dérivées partielles de $\phi_I$ sont calculées simplement par :

$$\phi_{I,i} = \underline{V}_{,i}^{T} \cdot \underline{b}_I + \underline{V}^{T} \cdot \underline{b}_{I,i} \tag{1.62}$$

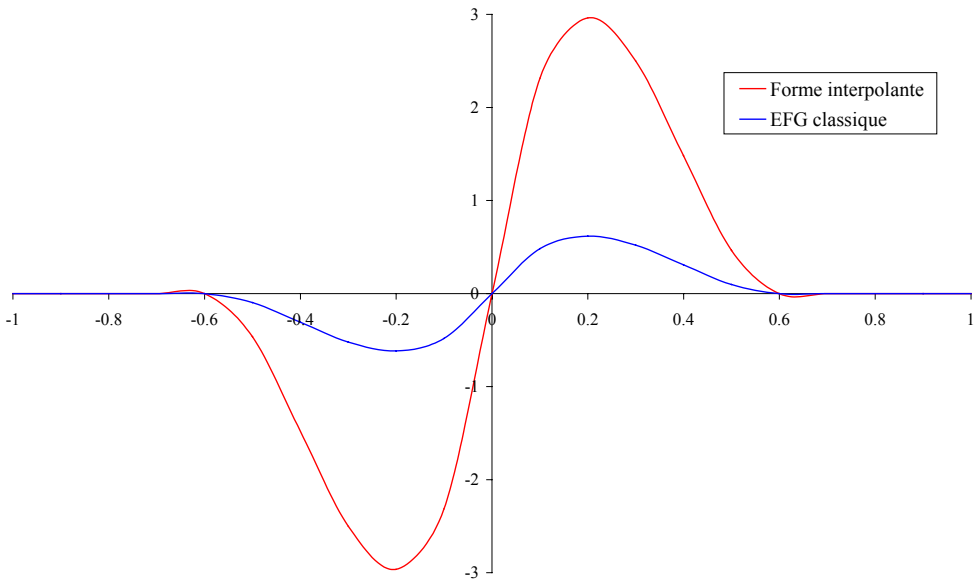



***Fig. 1.17 : Dérivées des fonctions de forme MLS polynomiales 1D (base primaire constante, $h_{max}$ = 0.6).***

## 1.4 METHODES BASEES SUR L'APPROXIMATION *RKM*

### 1.4.1 Formalisme continu et conditions de reproduction

Dans la forme continue de l'approximation *RKM*, on cherche à construire une fonction $W:\mathbb{R}^+ \to [0,1]$, continûment dérivable à l'ordre $s$ ($s \in \mathbb{N}$) et couramment appelée fenêtre ou « (*reproducing*) *kernel* » (méthode RKPM), ou encore fonction de lissage (méthode SPH). Cette fonction est construite de façon à reproduire la fonction solution $u(\underline{x})$ par une fonction $u^R(\underline{x})$, appelée fonction de reproduction, en écrivant le produit de convolution suivant : $u^R(\underline{x}) = u(\underline{x}) * W_\xi(\underline{x})$, soit encore sous forme intégrale :

$$u^R(\underline{x}) = \int_{-\infty}^{+\infty} u(\underline{\xi}) W_\xi(\underline{x}) d\Omega_\xi \tag{1.63}$$

où :

$$W_\xi(\underline{x}) = W\left(r\left(\underline{x} - \underline{\xi}, h_\xi\right)\right) \qquad r \in \mathbb{R}^+ \tag{1.64}$$

avec $h_\xi = 2^j h$ ($j \in \mathbb{Z}$), un coefficient réel strictement positif contrôlant la taille de la fenêtre $W$, appelé coefficient de dilatation et défini par [LIU 96a]. La variable $r$ est calculée de la même façon que pour l'approximation *MLS* (*cf.* §1.3.1). Le paramètre $h$ est en pratique représentatif de l'arrangement spatial des nœuds en cas de discrétisation numérique. Par exemple pour la méthode SPH, on prend en général $h_\xi = 2h$ ($j = 1$).

Pour permettre à l'approximation de conserver son caractère local, on choisit une fonction $W$ définie sur un support compact $\Omega_{\underline{\xi}}$ et ayant les propriétés suivantes :

$$W_\xi(\underline{x}) : \begin{cases} > 0 & \text{si } \underline{x} \in \Omega_\xi \\ = 0 & \text{si } \underline{x} \notin \Omega_\xi \end{cases} \tag{1.65}$$

et :

$$\lim_{h_\xi \to 0} W_\xi(\underline{x}) = \delta(\underline{x}) \tag{1.66}$$

où $\delta(\underline{x})$ représente la fonction Dirac.

On voit que la condition (1.65) est identique à (1.6) et que la fonction $W$ correspond en fait à la fonction de pondération des méthodes *MLS* (mêmes propriétés).

Remarque : On voit d'après (1.63) et (1.66) que : $\lim_{h_\xi \to 0} u^R(\underline{x}) = u(\underline{x})$.

Nous allons à présent résumer les différentes conditions de « reproduction » de la solution, c'est-à-dire en fait les conditions de consistance de l'approximation devant être vérifiées jusqu'à un certain ordre, ce dernier étant dépendant du problème à résoudre. Nous présentons ici uniquement le cas bidimensionnel, une analyse complète pour une dimension quelconque pouvant être trouvée dans les travaux de Liu et co-auteurs [LIU 96a].

Pour ce faire, on développe la fonction solution $u(\underline{\xi})$ avec $\underline{\xi} = (x_\xi, y_\xi)$, autour du point voisin $\underline{x} = (x, y)$, en utilisant la formule de Taylor-Lagrange à l'ordre $\tilde{m}$ :

$$u(\underline{\xi}) = \sum_{k=0}^{\tilde{m}} \frac{(-1)^k}{k!} \sum_{i+j=0}^{k} \mathrm{C}_k^i (x - x_\xi)^i (y - y_\xi)^j \left[ (\partial_x)^i (\partial_y)^j \right] u(\underline{x}) + R_{\tilde{m}+1}(\underline{\tilde{x}}) \tag{1.67}$$

où $R_{\tilde{m}+1}(\underline{\tilde{x}})$ représente le reste du développement et s'exprime par :

$$R_{\tilde{m}+1}(\underline{\tilde{x}}) = \frac{(-1)^{\tilde{m}+1}}{(\tilde{m}+1)!} \sum_{i+j=0}^{\tilde{m}+1} \mathrm{C}_{\tilde{m}+1}^i (x - x_\xi)^i (y - y_\xi)^j \left[ (\partial_x)^i (\partial_y)^j \right] u(\underline{\tilde{x}}) \tag{1.68}$$

avec $\underline{\tilde{x}} = \underline{x} + \theta(\underline{\xi} - \underline{x})$ et $\theta \in ]0,1[$.

En définissant la matrice des moments $\mathbf{m}(\underline{x})$ de la fenêtre $W$ par la relation :

$$\mathbf{m}(\underline{x}) = \int_{-\infty}^{+\infty} \underline{p}(\underline{x} - \underline{\xi}) \cdot \underline{p}^T(\underline{x} - \underline{\xi}) W_\xi(\underline{x}) d\Omega_\xi \tag{1.69}$$

Alors on obtient pour $u^R(\underline{x})$ en remplaçant la fonction $u(\underline{\xi})$ par son développement (1.67) dans l'intégrale (1.63) :

$$u^R(\underline{x}) = \sum_{k=0}^{\tilde{m}} \frac{(-1)^k}{k!} \sum_{i+j=0}^{k} \mathrm{C}_k^i \, m_{ij}(\underline{x}) \left[ (\partial_x)^i (\partial_y)^j \right] u(\underline{x}) + \tilde{R}_{\tilde{m}+1}(\underline{\tilde{x}}) \tag{1.70}$$

avec :

$$\tilde{R}_{\tilde{m}+1}(\underline{\tilde{x}}) = \frac{(-1)^{\tilde{m}+1}}{(\tilde{m}+1)!} \sum_{i+j=0}^{\tilde{m}+1} \mathrm{C}_{\tilde{m}+1}^i \, m_{ij}(\underline{x}) \left[ (\partial_x)^i (\partial_y)^j \right] u(\tilde{x}) \tag{1.71}$$

$m_{ij}(\underline{x})$ représente le moment d'ordre $i + j$.

On voit donc d'après (1.70) que pour reproduire la fonction $u(\underline{x})$ correctement jusqu'à l'ordre $\tilde{m}$, il faut vérifier les conditions suivantes :

$$m_{ij}(\underline{x}) = \delta_{0i}\delta_{0j} \qquad 0 \le i + j \le \tilde{m} \tag{1.72}$$

La condition $m_{00}(\underline{x}) = 1$ correspond en fait à ce que Monaghan [MON 82] appelait la condition de normalité pour la fonction « *kernel* » (méthode SPH).

Cependant, dans le cas de la résolution d'un problème aux limites, la fonction fenêtre initiale $W$ ne permet pas en général de vérifier les conditions de reproduction aux frontières du domaine. Pour remédier à ce problème, Liu et co-auteurs introduisent une nouvelle fenêtre $\overline{W}$, qu'ils définissent comme étant le produit de la fenêtre $W$ et d'une fonction de correction $C_\xi(\underline{x})$ déterminée de façon à satisfaire les conditions de reproduction sur tout le domaine :

$$\overline{W}_\xi(\underline{x}) = C_\xi(\underline{x})\, W_\xi(\underline{x}) \tag{1.73}$$

Ce qui permet d'écrire l'intégrale (1.63) sous la forme modifiée suivante :

$$u^R(\underline{x}) = \int_{-\infty}^{+\infty} u(\underline{\xi})\, \overline{W}_\xi(\underline{x})\, d\Omega_\xi \tag{1.74}$$

Ils supposent également que la fonction $C_\xi$ peut s'écrire sous la forme :

$$C_\xi(\underline{x}) = \underline{p}^T(\underline{x} - \underline{\xi}) \cdot \underline{b}(\underline{x}) \tag{1.75}$$

où $\underline{p}(\underline{x})$ correspond en fait à la base primaire polynomiale d'ordre $\tilde{m}$ (*cf.* §1.2) et où le vecteur des coefficients $\underline{b}$ s'écrit :

$$\underline{b}^T(\underline{x}) = \{b_{00}, b_{10}, b_{01}, \ldots, b_{\tilde{m}0}, \ldots, b_{0\tilde{m}}\} \tag{1.76}$$

Dans ce cas, on écrit le terme $\overline{m}_{ij}$ de la matrice des moments $\overline{\mathbf{m}}(\underline{x})$, associée à la nouvelle fenêtre sous la forme :

$$\overline{m}_{ij}(\underline{x}) = \sum_{s+k=0}^{\tilde{m}} m_{k+i\,s+j}(\underline{x})\, b_{ks}(\underline{x}) \tag{1.77}$$

Soit encore sous forme matricielle :

$$\overline{\mathbf{m}}(\underline{x}) = \mathbf{m}(\underline{x}) \cdot \underline{b}(\underline{x}) \tag{1.78}$$

Par ailleurs, les moments modifiés devant vérifier les conditions de consistance (1.72), on peut finalement déterminer $\underline{b}$ comme suit :

$$\underline{b}(\underline{x}) = \mathbf{m}^{-1}(\underline{x}) \cdot \underline{p}(\underline{0}) \tag{1.79}$$

Une analyse similaire doit également être réalisée pour déterminer les conditions permettant de reproduire les dérivées partielles de la solution.

La dérivée d'ordre $\alpha$, notée $D^{\alpha}$, de l'intégrale (1.63), correspondant en fait à la dérivée partielle par rapport aux *n* coordonnées spatiales, s'écrit :

$$D^{\alpha}u^{R}(\underline{x}) = \prod_{i=1}^{n}\left(\partial_{x_i}^{\alpha_i}\right)u^{R}(\underline{x}) = \int_{-\infty}^{+\infty} u(\underline{\xi})\, D^{\alpha}W_{\xi}(\underline{x})\, d\Omega_{\xi} \tag{1.80}$$

avec : $\sum_i \alpha_i = \alpha$.

En remplaçant (1.67) dans (1.80), on obtient alors :

$$D^{\alpha}u^{R}(\underline{x}) = \sum_{k=0}^{\tilde{m}} \frac{(-1)^k}{k!} \sum_{i+j=0}^{k} \mathrm{C}_k^i\, D^{\alpha}m_{ij}(\underline{x}) \left[ (\partial_x)^i (\partial_y)^j \right] u(\underline{x}) + D^{\alpha}\tilde{R}_{\tilde{m}+1}(\underline{\tilde{x}}) \tag{1.81}$$

avec :

$$D^{\alpha}\mathbf{m}(\underline{x}) = \int_{-\infty}^{+\infty} \underline{p}(\underline{x}-\underline{\xi}) \cdot \underline{p}^{T}(\underline{x}-\underline{\xi})\, D^{\alpha}W_{\xi}(\underline{x})\, d\Omega_{\xi} \tag{1.82}$$

Dans ce cas, les conditions complètes pour reproduire le champ solution et ses dérivées partielles jusqu'à l'ordre $\alpha$ s'écrivent :

$$D^{\beta}m_{ij}(\underline{x}) = (-1)^{i+j}\, i!\, j!\, \delta_{i\beta}\delta_{j\beta} \qquad 0 \le \beta \le \alpha,\ 0 \le i+j \le \tilde{m} \tag{1.83}$$

Si l'on utilise les moments modifiés, on doit résoudre le système matriciel suivant :

$$\begin{bmatrix} \mathbf{m} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ D\mathbf{m} & \mathbf{m} & \mathbf{0} & \cdots & \mathbf{0} \\ D^2\mathbf{m} & 2D\mathbf{m} & \mathbf{m} & & \vdots \\ \vdots & \vdots & & \ddots & \mathbf{0} \\ \mathrm{C}_{\alpha}^{0} D^{\alpha}\mathbf{m} & \mathrm{C}_{\alpha}^{1} D^{\alpha-1}\mathbf{m} & & \cdots & \mathrm{C}_{\alpha}^{\alpha}\mathbf{m} \end{bmatrix} \begin{bmatrix} \underline{b} \\ D\underline{b} \\ D^2\underline{b} \\ \vdots \\ D^{\alpha}\underline{b} \end{bmatrix} = \begin{bmatrix} \underline{p}(\underline{0}) \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{1.84}$$

Dans les deux sections suivantes, nous présentons les méthodes SPH et RKPM, formes discrétisées de l'approximation *RKM*, la méthode SPH correspondant à la forme continue non corrigée (1.63), et RKPM, à la forme corrigée (1.74).

## 1.4.2 La méthode SPH

### *1.4.2.1 Introduction*

La méthode SPH (« *Smoothed Particle Hydrodynamics* ») est l'une des méthodes meshfree les plus utilisées en pratique, essentiellement pour simuler les phénomènes stellaires et astrophysiques, les problèmes d'impacts à haute vitesse (projectiles), les problèmes de transferts de masse et de chaleur, et plus généralement les problèmes de Mécanique des Fluides (écoulements, surface libre, etc.).

La principale difficulté pour utiliser cette méthode et plus généralement toutes les méthodes de particules, dans le cadre conventionnel des problèmes aux limites rencontrés en Mécanique des Milieux Continus, réside dans l'imposition des conditions aux limites essentielles, cette méthode n'ayant pas été développée à l'origine pour ce type de problèmes. Ainsi, une portion de frontière ayant ce type de condition imposée, nécessite un traitement numérique spécifique. Une technique courante consiste à introduire des particules fictives situées à l'extérieur du domaine et appartenant au voisinage des particules réelles à la frontière [RAN 96]. Une autre technique consiste à utiliser des particules frontières de masse réduite [CHE 00].

Ci-après, nous rappelons le principe, ainsi que quelques unes des propriétés importantes de la méthode SPH.

### *1.4.2.2 Principe*

L'approximation SPH correspond à la forme discrétisée de l'intégrale (1.63), que l'on calcule sur un domaine fini $\Omega$, généralement en utilisant une méthode d'intégration nodale de type trapèzes :

$$u^h\left(\underline{x}\right) = \sum_{I=1}^{N} W_I\left(\underline{x}\right) \Delta V_I \, u_I \tag{1.85}$$

en définissant $W_I\left(\underline{x}\right)$ comme en (1.7) et où $\underline{u}^T = \left\{u_I = u\left(\underline{x}_I\right)\right\}_{I=1}^{N}$ représente le vecteur des valeurs nodales. $\Delta V_I$ représente l'aire ou le volume associé au nœud *I*. Dans le cas 1D et pour un arrangement nodal régulier, $\Delta V_I$ est en pratique calculé par la règle du trapèze. Ainsi, pour un nœud intérieur, on aura :

$$\Delta V_I = \Delta x_I = \left(x_{I+1} - x_{I-1}\right)/2 \tag{1.86}$$

La généralisation au cas multidimensionnel est plus délicate. Une expression typique est donnée par Bonet et Kulasegaram [BON 00] :

$$\Delta V_I = \frac{1}{\sum_{J=1}^{N} W_J\left(\underline{x}_I\right)} \tag{1.87}$$

Il est possible d'écrire la relation (1.85) sous la forme classique : $u^h(\underline{x}) = \sum_{I=1}^{N} \phi_I(\underline{x})\, d_I$ , en posant : $d_I = \Delta V_I\, u_I$, contribution nodale du nœud I, et en prenant pour les fonctions de forme :

$$\phi_I(\underline{x}) = W_I(\underline{x}) \tag{1.88}$$

On voit d'après (1.88) que l'approximation ainsi construite n'est pas interpolante (généralement, $\forall I, \forall J, W_I(\underline{x}_J) \neq \delta_{IJ}$ et $d_I \neq u_I$).

*1.4.2.3 Conditions de consistance*

Si l'on exprime la matrice des moments discrets **M** associée à la fenêtre initiale à partir de l'intégrale (1.69) et pour l'ensemble des nœuds du domaine, on obtient la relation suivante :

$$\mathbf{M}(\underline{x}) = \sum_{I=1}^{N} \underline{p}(\underline{x}-\underline{x}_I) \cdot \underline{p}^T(\underline{x}-\underline{x}_I)\, W_I(\underline{x}) \tag{1.89}$$

Dans ce cas, les conditions de reproduction (1.72) et (1.83) s'écrivent de manière identique :

$$M_{ij}(\underline{x}) = \delta_{0i}\delta_{0j} \qquad 0 \le i+j \le \tilde{m} \tag{1.90}$$

$$D^{\beta} M_{ij}(\underline{x}) = (-1)^{i+j}\, i!\, j!\, \delta_{i\beta}\delta_{j\beta} \qquad 0 \le \beta \le \alpha,\ 0 \le i+j \le \tilde{m} \tag{1.91}$$

On montre que ces conditions ne sont en général pas vérifiées aux frontières du domaine. Par exemple dans le cas 1D, on voit que la condition d'ordre zéro pour un point frontière devient : $M_{00}(\underline{x}) = 1/2$ (*cf.* Fig. 1.18 où on a en posé $\phi_I = W_I$), ce qui conduit à une valeur erronée de $u^h(\underline{x})$. En effet, pour une fonction constante $u(\underline{x}) = c$, on aura : $u^h(\underline{x}) = c/2$. On renvoie à [BEL 96] pour la démonstration dans le cas linéaire (ordre 1).

*1.4.2.4 Approximation des dérivées de la solution*

Dans l'approche conventionnelle de la méthode SPH, une hypothèse simplificatrice est utilisée pour estimer les dérivées de la solution dans le cas des problèmes aux limites. En effet, si l'on revient à la forme continue (1.63) de l'approximation *RKM* exprimée sur le domaine d'étude **Ω**, on obtient :

$$u^R(\underline{x}) = \int_{\Omega} u(\underline{\xi})\, W_{\xi}(\underline{x})\, d\Omega_{\xi} \tag{1.92}$$

Alors dans ce cas, l'approximation de la fonction dérivée de la solution, par exemple la dérivée partielle par rapport à *x*, peut s'écrire :

$$\partial_x u^R(\underline{x}) = \int_{\Omega} \partial_x u(\underline{\xi})\, W_{\xi}(\underline{x})\, d\Omega_{\xi} \tag{1.93}$$

En intégrant par parties l'expression précédente, on trouve :

$$\partial_x u^R(\underline{x}) = \int_\Gamma u(\underline{\xi}) W_\xi(\underline{x}) d\Gamma_\xi - \int_\Omega u(\underline{\xi}) \partial_x W_\xi(\underline{x}) d\Omega_\xi \tag{1.94}$$

où $\Gamma$ représente la frontière du domaine $\Omega$. Dans la pratique, on néglige le terme d'intégrale sur la frontière pour écrire :

$$\partial_x u^R(\underline{x}) \approx -\int_\Omega u(\underline{\xi}) \partial_x W_\xi(\underline{x}) d\Omega_\xi \tag{1.95}$$

Ce qui conduit finalement à la forme discrète suivante :

$$\partial_x u^h(\underline{x}) \approx -\sum_{I=1}^{N} \partial_x W_I(\underline{x}) d_I \tag{1.96}$$

Cette simplification, et plus généralement l'intégration directe par collocation, sont en fait à l'origine des problèmes numériques de convergence et de précision rencontrés par la méthode SPH.

### *1.4.2.5 Fonctions de lissage usuelles*

Les fonctions *Gauss* (1.21), *Spline2* (1.27) et polynomiale (1.28) définies pour les méthodes de type *MLS* constituent également des fonctions de lissage couramment utilisées en pratique avec la méthode SPH. Dans le cas de la fonction de *Gauss* cependant, le paramètre $c_I$ devient ici un coefficient constant indépendant de l'arrangement nodal.

Par ailleurs, de même que pour les méthodes *MLS*, il est possible de définir des fonctions à support radial ou rectangulaire.

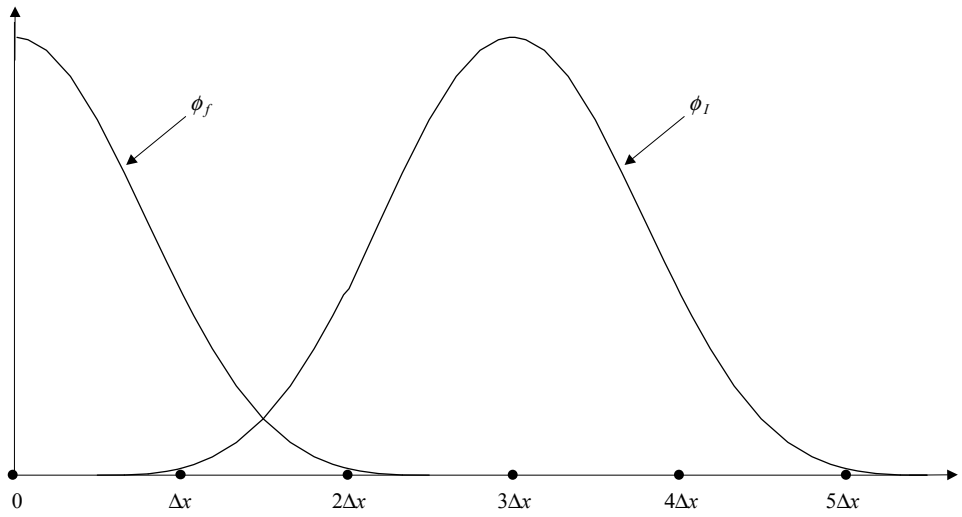



***Fig. 1.18 : Illustration du problème de consistance 1D aux frontières (nœuds f et I).***

### 1.4.3 La méthode RKPM

*1.4.3.1 Principe*

De même que pour la méthode SPH, l'approximation RKPM (« *Reproducing Kernel Particle Method* ») correspond à la forme discrétisée de l'intégrale (1.74) :

$$u^h(\underline{x}) = \sum_{I=1}^{N} \bar{W}_I(\underline{x})\,\Delta V_I\,u_I \tag{1.97}$$

avec :

$$\bar{W}_I(\underline{x}) = C_I(\underline{x}) \cdot W_I(\underline{x}) \tag{1.98}$$

et en utilisant (1.89) :

$$C_I(\underline{x}) = \underline{p}^T(\underline{x} - \underline{x}_I) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \underline{p}(\underline{0}) \tag{1.99}$$

La fonction de forme est dans ce cas définie par :

$$\phi_I(\underline{x}) = \bar{W}_I(\underline{x}) \tag{1.100}$$

Le calcul des dérivées partielles de $\phi_I(\underline{x})$ ( $j = 1, n$ ) s'effectue simplement par différentiation de (1.98) et (1.99) :

$$\phi_{I,j} = C_{I,j}(\underline{x})\,W_I(\underline{x}) + C_I(\underline{x})\,W_{I,j}(\underline{x}) \tag{1.101}$$

et :

$$C_{I,j}(\underline{x}) = \underline{p}_j{}^T(\underline{x} - \underline{x}_I) \cdot \underline{b}(\underline{x}) + \underline{p}^T(\underline{x} - \underline{x}_I) \cdot \underline{b}_{,j}(\underline{x}) \tag{1.102}$$

Remarque : De même que pour les méthodes *MLS*, il est possible d'optimiser le calcul des dérivées partielles de $\phi_I(\underline{x})$ en utilisant la décomposition *LU* de la matrice **M**.

*1.4.3.2 Conditions aux limites essentielles et fonction de dilatation*

De même que pour la méthode SPH, l'approximation RKPM ainsi construite n'est pas interpolante, ce qui pose problèmes pour imposer les conditions aux limites essentielles.

Gosz et Liu [GOS 96] montrent que pour vérifier la consistance aux frontières de Dirichlet, il est nécessaire de modifier soit les fonctions de forme ou bien les fonctions fenêtres calculées pour les particules proches, afin de les forcer à s'annuler sur ces frontières.

Dans la cas 1D, la condition d'annulation des fonctions de forme s'obtient en utilisant une propriété intéressante des fonctions à support compact : si le nombre de particules présentes dans le support de la fonction de forme est égal au nombre de monômes de la base polynomiale $\underline{p}$, alors on montre que les fonctions de forme sont interpolantes (voir la démonstration dans [GOS 96] pour la méthode RKPM).

Cette condition d'annulation est néanmoins difficile à mettre en œuvre dans le cas général 2D/3D (frontières courbes). Gosz et Liu montrent que ceci est par exemple possible si la frontière est rectiligne et que la distribution nodale est régulière (pas de discrétisation constants).

Une autre technique consiste à forcer l'annulation des fonctions fenêtres au voisinage de la frontière. Pour cela, Gosz et Liu proposent d'introduire une fonction de dilatation tenant compte de la géométrie réelle de la frontière. En effet, du fait du choix initial d'un coefficient de dilatation $h_\xi$ (forme continue de l'approximation *RKM*) ne dépendant que de la position des particules ou nœuds du domaine, la méthode RKPM standard (forme discrétisée) utilise un coefficient de dilatation $h_I$ constant par nœud. Le support résultant pour la fonction fenêtre associée possède alors une forme de type rectangulaire ou elliptique, mal adaptée pour vérifier les conditions de consistance pour des frontières à géométrie complexe. L'idée consiste donc à introduire une fonction de dilatation $h_\xi(\underline{x})$ pour la forme continue ou $h_I(\underline{x})$ pour la forme discrétisée, dépendant également de la position du point d'évaluation[2] $\underline{x}$, de façon à obtenir des supports de forme quelconque pour les nœuds proches de la frontière.

Après discrétisation, cela conduirait donc à calculer la fonction $h_I(\underline{x})$, telle que :

$$h_I(\underline{x}) = \underline{a}_I{}^T(\underline{x}) \cdot \Delta \underline{h}_I \tag{1.103}$$

et :

$$\underline{a}_I(\underline{x}) = \sum_{K=1}^{n_d} N_{IK}(\underline{x})\, \underline{a}_K \tag{1.104}$$

$\underline{a}_I(\underline{x})$ est un vecteur sans dimension associé au nœud *I*, qui permet de caractériser le raffinement local. La quantité $\Delta \underline{h}_I$ est représentative du poids du nœud dans la discrétisation spatiale et s'apparente au paramètre *h* de la forme continue (*cf.* §1.2.1).

Les $N_{IK}(\underline{x})$ représentent des fonctions d'interpolation (polynômes de Lagrange ou splines) utilisées pour générer des fonctions de dilatation suffisamment lisses pour $n_d$ valeurs de dilatation données.

Les composantes de $\underline{a}_K$ représentent donc les coefficients du polynôme choisi pour interpoler la portion de frontière coupant le support de la fonction de forme, l'ordre du polynôme étant fourni par le nombre de points situés sur la frontière (*cf.* Fig. 1.19).

En considérant *W*, la fonction fenêtre choisie, alors la fonction fenêtre modifiée s'écrit :

$$\tilde{W}_I(\underline{x}) = E(h_I)\, W\left(r\left(\underline{x} - \underline{x}_I, h_I(\underline{x})\right)\right) \tag{1.105}$$

[2] Suivant la méthode d'intégration utilisée, les points d'évaluation correspondent soit aux points d'intégration tels que définis par exemple, par la méthode de Gauss-Legendre, soit aux nœuds du problème (collocation).

$E(h_I)$ est un facteur de normalisation constant défini pour un coefficient de dilatation $h_I$ donné, de manière à obtenir une aire unité pour la fenêtre ainsi construite.

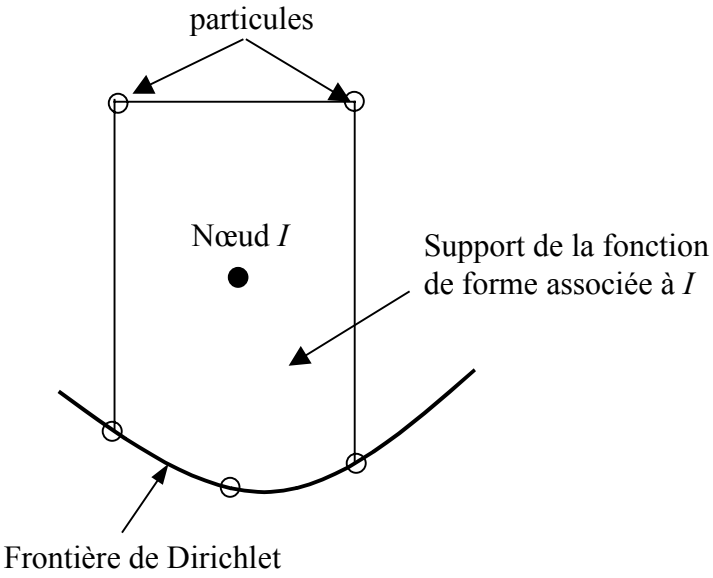



***Fig. 1.19 : Fonction de forme à support quelconque près d'une frontière de Dirichlet.***

Par exemple dans le cas 1D, si l'on pose $s=(x-x_I)/h_I(x)$, alors la dérivée de *s*, notée *ds*, est déterminée par la relation :

$$ds=\frac{1}{h_I(x)}-h_I'(x)\frac{x-x_I}{h_I^2(x)} \tag{1.106}$$

Le facteur de normalisation est alors défini par :

$$E(h_I)=\int ds \tag{1.107}$$

Remarques :

- D'après les auteurs [GOS 96], la méthode d'annulation de la fonction de forme fournit de meilleurs résultats que celle de la fonction fenêtre avec la fonction de dilatation.
- Pour notre part et suivant les applications, nous avons préférer utiliser les techniques utilisées avec les méthodes *MLS*, à savoir le couplage avec une rangée d'éléments finis au niveau des frontière du domaine ou l'utilisation de fonctions de pondération (fenêtre) singulières.

### *1.4.3.3 Exemples de fonctions fenêtres modifiées*

Nous donnons ci-après quelques exemples de fenêtres 1D modifiées ($\bar{W}(r), r \le 1$), une liste plus complète étant fournie dans [LIU 96a ; LIU 96b]. Les fenêtres initiales, notées $W(r)$, correspondent aux fonctions de pondération que nous avons définies pour les méthodes *MLS*.

*Fonctions modifiées pour reproduire la base* $\{1, x, x^2\}$ ($\tilde{m} = 2$) :

***Gauss :***

$$\bar{W}(r) = \left[\frac{3}{2} - 4r^2\right] W(r) \tag{1.108}$$

***Spline2 :***

$$\bar{W}(r) = \frac{1}{17}\left(27 - 120r^2\right) W(r) \tag{1.109}$$

*Fonctions modifiées pour reproduire la base* $\{1, x, x^2, x^3, x^4\}$ ($\tilde{m} = 4$) :

***Gauss :***

$$\bar{W}(r) = \left[\frac{15}{8} - 10r^2 + 8r^4\right] W(r) \tag{1.110}$$

***Spline2 :***

$$\bar{W}(r) = \frac{30}{80347}\left(5667 - 57260r^2 + 95088r^4\right) W(r) \tag{1.111}$$

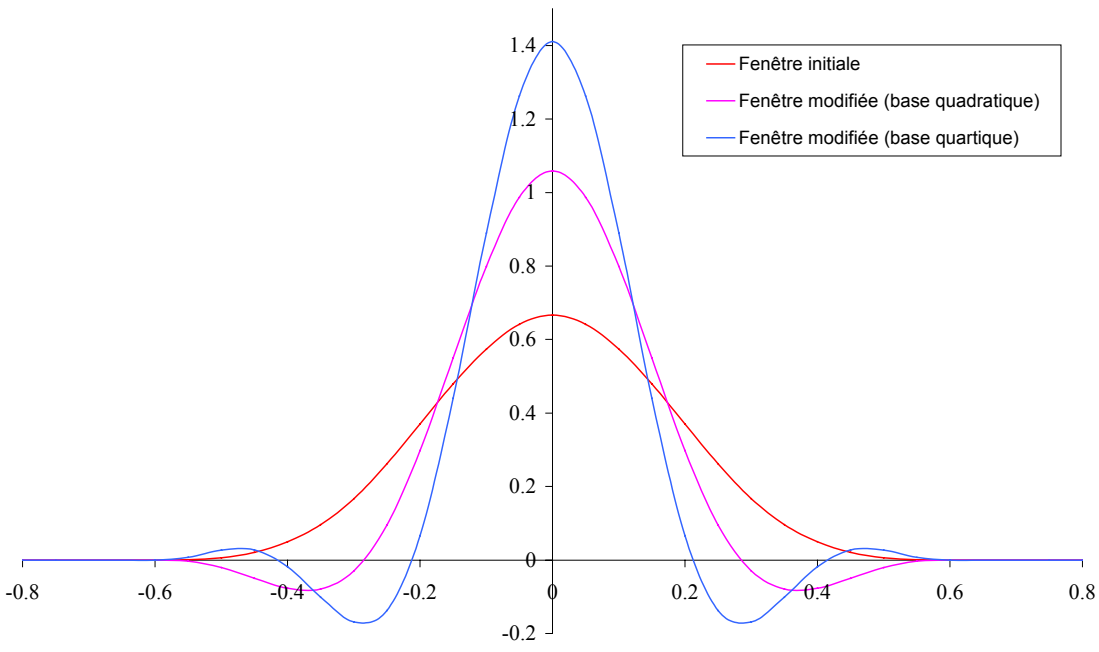



***Fig. 1.20 : Fonction Spline2 : fenêtres initiale et modifiées.***

### 1.4.4 La méthode RKI

Très récemment, Chen et co-auteurs [CHE 03] ont proposé une nouvelle méthode qu'ils ont appelée RKI (« *Reproducing Kernel Interpolation* ») et qui modifie la méthode RKPM, de façon à obtenir des fonctions de forme interpolantes. Pour ce faire, ils utilisent une technique d'enrichissement dont l'idée est proche de celle utilisée pour le couplage avec des éléments finis [BEL 95 ; BEL 96 ; HUE 00].

Ainsi, ils écrivent la fonction de forme modifiée $\phi_I$, comme la somme d'une fonction interpolante $N_I$ et d'une fonction d'enrichissement $\bar{\phi}_I$ :

$$\phi_I(\underline{x}) = N_I(\underline{x}) + \bar{\phi}_I(\underline{x}) \tag{1.112}$$

Si la fonction $N_I$ choisie correspond aux fonctions de forme des éléments finis, on retrouve la méthode d'enrichissement proposée par Huerta et Fernández-Méndez [HUE 00]. Chen propose quant à lui d'utiliser la fonction suivante :

$$N_I(\underline{x}) = \frac{\bar{W}_I(\underline{x})}{\bar{W}_I(\underline{0})} \tag{1.113}$$

en utilisant $2h_I$ comme mesure du support de $N_I$ (support compact) et en notant $\bar{W}_I(\underline{x})$, la fonction de pondération choisie. Pour notre part, nous proposons d'utiliser également la fonction de pondération (1.44), modifiée pour obtenir des fonctions de forme interpolantes pour l'approximation *MLS*.

$\bar{\phi}_I$ est construite de façon à ce que l'approximation atteigne l'ordre de consistance désiré, ce qui, après vérification des conditions de reproduction classiques conduit à la forme suivante :

$$\bar{\phi}_I(\underline{x}) = W_I(\underline{x})\, \underline{p}^T(\underline{x}-\underline{x}_I)\cdot \mathbf{M}^{-1}(\underline{x})\cdot\left[\underline{p}(\underline{0}) - \sum_{J=1}^{N} \underline{p}(\underline{x}-\underline{x}_J)\, N_J(\underline{x})\right] \tag{1.114}$$

où $W_I(\underline{x})$ est une fonction de pondération pouvant être prise égale ou non à $\bar{W}_I(\underline{x})$, et où $\mathbf{M}$ est définie par :

$$\mathbf{M}(\underline{x}) = \sum_{I=1}^{N} W_I(\underline{x})\, \underline{p}(\underline{x}-\underline{x}_I)\cdot \underline{p}^T(\underline{x}-\underline{x}_I) \tag{1.115}$$

## 1.4.5 Décomposition « multi-échelle » pour l'approximation *RKM*

### *1.4.5.1 Principe*

L'approche « multi-échelle » ou par ondelettes est une approche est ici proposée par Liu et co-auteurs [LIU 96a ; LIU 96b ; LIU 00a] pour permettre d'analyser dans le domaine fréquentiel, les conditions de reproduction de l'approximation *RKM* énoncées au départ dans le domaine transitoire.

En effet, si l'on considère le produit de convolution (1.74), la fonction fenêtre $\overline{W}$ se comporte en fait comme un filtre passe-bas appliqué à la solution $u$ et permettant ainsi d'obtenir la solution approchée ou « reproduite », $u^R$.

Dans l'approche « multi-échelle », $\overline{W}$ est effectivement appelé filtre. L'« échelle » dans ce cas correspond au contenu fréquentiel de « l'image reconstruite » $u^{Rh}$, correspondant au coefficient de dilatation $h$ :

$$u^{Rh}\left(\underline{x}\right) = P_h u\left(\underline{x}\right) \tag{1.116}$$

où $P_h$ peut être considéré comme un opérateur de projection à l'échelle $h$.

Du fait qu'un problème physique peut contenir plusieurs échelles, il est préférable de décomposer la solution de manière à pouvoir prendre en compte son contenu fréquentiel complet. Pour ce faire, il est possible de construire une série de filtres passe-bas ou projections par changements successifs d'échelles :

$$P_h u\left(\underline{x}\right) \supset P_{2h} u\left(\underline{x}\right) \supset P_{4h} u\left(\underline{x}\right) \supset \cdots \supset P_{2^i h} u\left(\underline{x}\right) \cdots \tag{1.117}$$

soit encore :

$$u^R\left(\underline{x}\right) = u^{Rh}\left(\underline{x}\right) + u^{R2h}\left(\underline{x}\right) + u^{R4h}\left(\underline{x}\right) + \ldots \tag{1.118}$$

Chaque échelle peut ensuite être étudiée séparément. Par ailleurs, la différence entre deux échelles permet d'obtenir un filtre passe-haut, utilisable notamment pour raffiner la solution localement.

En effet, si l'on définit la fonction ondelette $\psi\left(\underline{x}-\underline{\xi}, 2^i h\right)$ pour un entier naturel $i$ quelconque :

$$\psi\left(\underline{x}-\underline{\xi}, 2^i h\right) = W\left(\underline{x}-\underline{\xi}, 2^{i-1} h\right) - W\left(\underline{x}-\underline{\xi}, 2^i h\right) \tag{1.119}$$

où afin d'alléger les notations, nous avons noté $W\left(\underline{x}-\underline{\xi}, 2^i h\right)$ pour $W\left(r\left(\underline{x}-\underline{\xi}, 2^i h\right)\right)$.

Si l'on considère l'opérateur de projection complémentaire associé, $Q_{2^i h}$ :

$$Q_{2^i h}u(\underline{x}) = \sum_{I=1}^{N} \psi\left(\underline{x} - \underline{x}', 2^i h\right) u_I \, \Delta V_I \tag{1.120}$$

alors la solution projetée $P_{2^{i-1} h}u(\underline{x})$ s'écrit simplement :

$$P_{2^{i-1} h}u(\underline{x}) = P_{2^i h}u(\underline{x}) + Q_{2^i h}u(\underline{x}) \tag{1.121}$$

*1.4.5.2 Analyse en domaine fréquentiel*

L'approche « multi-échelle » associée à l'approximation *RKM* profite largement des propriétés de la convolution dans le domaine fréquentiel pour améliorer la compréhension et l'analyse. Pour ce faire, Liu et co-auteurs [LIU 96a ; LIU 96b] utilisent une transformée de Fourier semi-discrète (SDFT), retranscrite ici dans le cas 1D pour simplifier les notations :

$$\hat{u}(\zeta) = SF\left[u(x)\right] = \sum_{I=-\infty}^{+\infty} u(x_I) e^{-i\zeta x_I} \Delta x_I \tag{1.122}$$

Dans le cas d'un échantillonnage uniforme, la transformée SFDT devient périodique et s'écrit :

$$\hat{u}(\zeta) = \Delta x \sum_{n \in \mathbb{Z}} u(n\Delta x)\left[\cos(\zeta n\Delta x) - i\sin(\zeta n\Delta x)\right] \tag{1.123}$$

On montre que la transformée de Fourier du produit de convolution discret, noté $*_D$, s'écrit alors dans le cas d'une fenêtre simple (sans coefficient de dilatation) :

$$F\left[u(x) *_D \bar{W}(x)\right] = \int_{-\infty}^{+\infty} \left[ \sum_{I=-\infty}^{+\infty} u(x_I) \cdot \bar{W}_I(x) \Delta x_I \right] e^{-i\zeta x} dx \tag{1.124}$$

c'est-à-dire encore :

$$\hat{u}^R(\zeta) = \hat{u}(\zeta) \cdot \hat{\bar{W}}(\zeta) \tag{1.125}$$

en notant $\hat{u}^R(\zeta)$ et $\hat{\bar{W}}(\zeta)$, la transformée de Fourier continue des fonctions $u^R$ et $\bar{W}$.

L'expression pour une fenêtre avec paramètre de dilatation *h* devient :

$$\hat{u}^{Rh}(\zeta) = F\left[u^{Rh}(x)\right] = \sum_{I=-\infty}^{+\infty} \hat{\bar{W}}_h(\zeta) u(x_I) e^{-i\zeta x_I} \Delta x_I \tag{1.126}$$

avec :

$$\begin{cases} \hat{\bar{W}}_h(\zeta) = \int_{-\infty}^{+\infty} \bar{W}_h(s) e^{-i2h\zeta s} ds \\ s = (x - x_I)/2h \end{cases} \tag{1.127}$$

et :

$$\forall r \in \mathbb{R}^+, \bar{W}_h(r) = \frac{1}{h} \bar{W}\left(\frac{r}{h}\right) \tag{1.128}$$

Remarques :

- La généralisation au cas multidimensionnel se fera en écrivant : $\hat{\bar{W}}_h(\underline{\zeta}) = \prod_{j=1}^{n} \hat{\bar{W}}_h(\zeta_j)$, $n$ étant le nombre de dimensions spatiales.
- La transformée de Fourier de la fonction : $W_h(s)$, s'écrit simplement : $\hat{W}_h(\zeta) = \hat{W}(h\zeta)$.

Les conditions de reproduction 1D (1.72), exprimées à l'ordre $k$ dans le domaine fréquentiel, s'écrivent alors simplement :

$$m_k(x) = i^k \hat{W}_h^{(k)}(0) = i^k h^k \hat{W}^{(k)}(0) = \delta_{0k} \qquad 0 \le k \le n \tag{1.129}$$

Par exemple à l'ordre 0 et pour la forme continue de l'approximation, on aura la condition suivante :

$$\hat{W}_h(0) = 1 \Leftrightarrow \int_{-\infty}^{+\infty} W_h(x - \xi) d\xi = 1 \tag{1.130}$$

Le passage à une analyse dans le domaine fréquentiel permet selon les auteurs, de mieux mettre en évidence les paramètres pouvant affecter le contenu fréquentiel de la réponse d'un système mécanique, à savoir essentiellement la discrétisation spatiale, le type d'échantillonnage (uniforme ou non uniforme) et la méthode d'intégration numérique (ex : Galerkin), et ainsi à améliorer la précision des calculs par le choix des paramètres adéquats.

En effet, le domaine fréquentiel est utilisé dans ce cas pour construire des fonctions fenêtres modifiées vérifiant les conditions de reproduction (1.129), en remarquant que la fonction finale est en fait une combinaison linéaire de la fonction initiale et de ses dérivées jusqu'à l'ordre de reproduction considéré :

$$\hat{W}_h(\zeta) = \sum_{k=0}^{n} \alpha_k \hat{W}_h^{(k)}(\zeta) = \sum_{k=0}^{n} \alpha_k h^k \hat{W}^{(k)}(h\zeta) \tag{1.131}$$

Les coefficients $\alpha_k$ sont alors déterminés de façon à vérifier (1.129) :

$$\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} \hat{W}(0) & a\hat{W}'(0) & \cdots & a^n\hat{W}^{(n)}(0) \\ \hat{W}'(0) & a\hat{W}''(0) & \cdots & a^n\hat{W}^{(n+1)}(0) \\ \vdots & & \ddots & \vdots \\ \hat{W}^{(n)}(0) & a\hat{W}^{(n+1)}(0) & \cdots & a^n\hat{W}^{(2n)}(0) \end{bmatrix}^{-1} \cdot \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{1.132}$$

La fonction fenêtre modifiée s'exprime ensuite dans le domaine transitoire en utilisant (1.129) et (1.131) :

$$\bar{W}_h(x-\xi) = \sum_{k=0}^{n} \alpha_k i^{-k} (x-\xi)^k W_h(x-\xi) \tag{1.133}$$

### 1.4.6 Formalisme continu pour l'approximation MLS

Afin de fournir un cadre commun aux différentes méthodes meshfree (essentiellement *MLS* et *RKM*), Belytschko et co-auteurs [BEL 94 ; BEL 96] proposent d'utiliser une forme continue pour la relation (1.10), correspondant à une norme $L_2$ pondérée :

$$J(\underline{\alpha})\big|_{\underline{x}} = \int_{\underline{\Omega}} W_{\xi}(\underline{x})\left[u^R(\underline{x}) - u(\underline{\xi})\right]^2 d\Omega_{\xi} \tag{1.134}$$

où l'on a utilisé les mêmes notations que pour l'approximation *RKM* (*cf.* §1.4.1) : $u^R(\underline{x})$ correspond à l'approximation continue de la fonction solution au point quelconque $\underline{x}$, calculée sur tout voisinage $\Omega_{\xi}$ et $W$ correspond à la fonction de pondération associée au point d'évaluation $\underline{\xi}$ (voisin de $\underline{x}$). Après minimisation de (1.134) et en prenant $\underline{\xi} = \underline{x}$, on obtient la forme continue cherchée pour l'approximation globale :

$$u^R(\underline{x}) = \int_{\underline{\Omega}} C_{\xi}(\underline{x}) \cdot W_{\xi}(\underline{x}) \cdot u(\underline{\xi})\, d\Omega_{\xi} \tag{1.135}$$

avec :

$$C_{\xi}(\underline{x}) = \underline{p}^T(\underline{x}) \cdot \mathbf{m}^{-1}(\underline{x}) \cdot \underline{p}(\underline{\xi}) \tag{1.136}$$

et :

$$\mathbf{m}(\underline{x}) = \int_{\underline{\Omega}} \underline{p}(\underline{\xi}) \cdot \underline{p}^T(\underline{\xi})\, W_{\xi}(\underline{x})\, d\Omega_{\xi} \tag{1.137}$$

$C_{\xi}(\underline{x})$ est en fait assimilable à la fonction de correction de l'approximation *RKM*, et $\mathbf{m}(\underline{x})$, à la matrice des moments définie en (1.69)

### 1.4.7 La méthode MLS/RK

La construction de cette méthode [LIU 97] repose sur l'utilisation du formalisme continu de l'approximation *MLS* (*cf.* section §1.4.6). Cependant, contrairement à l'approximation *MLS*, l'approximation locale recherchée ici est de la forme suivante :

$$u^R(\underline{x}) = \underline{p}^T\left(\frac{\underline{x}-\underline{\xi}}{h}\right)\cdot\underline{\alpha}(\underline{\xi}) \qquad \forall \underline{x}\in\Omega_\xi \tag{1.138}$$

Après minimisation de la norme $L_2$ pondérée (1.134), on obtient une expression analogue à (1.135), avec pour $i,j=1,m$ :

$$C_\xi(\underline{x}) = \underline{p}^T\left(\frac{\underline{\xi}-\underline{x}}{h}\right)\cdot\mathbf{m}^{-1}(\underline{x})\cdot\underline{p}(\underline{0}) \tag{1.139}$$

et :

$$m_{ij}(\underline{x}) = \int_{\Omega} p_i\left(\frac{\underline{\xi}-\underline{x}}{h}\right)p_j\left(\frac{\underline{\xi}-\underline{x}}{h}\right)W_\xi(\underline{x})\,d\Omega_\xi \tag{1.140}$$

On voit que la matrice des moments (1.140) correspond en fait à celle définie en (1.137), mais pour une base primaire polynomiale locale.

### 1.4.8 Synthèse des formalismes continus pour les méthodes *MLS* et *RKM*

Nous proposons dans le tableau 1.1 ci-après, une synthèse des formalismes continus pour les principales méthodes meshfree *MLS* et *RKM*, présentées dans les sections précédentes.

| **Approximations** | **Principales Méthodes meshfree** | **Formalisme continu :** $u^R(\underline{x}) = \int_{\underline{\Omega}} \phi_\xi(\underline{x})\, u(\underline{\xi})\, d\Omega_\xi$ ; $\phi_\xi(\underline{x}) = C_\xi(\underline{x})\, W_\xi(\underline{x})$ |
|---|---|---|
| *MLS* | EFGM, DEM | $C_\xi(\underline{x}) = \underline{p}^T(\underline{x}) \cdot \mathbf{m}^{-1}(\underline{x}) \cdot \underline{p}(\underline{\xi})$ ; $\mathbf{m}(\underline{x}) = \int_{\underline{\Omega}} \underline{p}(\underline{\xi}) \cdot \underline{p}^T(\underline{\xi})\, W_\xi(\underline{x})\, d\Omega_\xi$ |
| | *Références* | [BEL 94] |
| *RKM* | SPH | $C_\xi(\underline{x}) = 1$ |
| | *Références* | [MON 82] |
| | RKPM | $C_\xi(\underline{x}) = \underline{p}^T(\underline{x}-\underline{\xi}) \cdot \mathbf{m}^{-1}(\underline{x}) \cdot \underline{p}(\underline{0})$ ; $\mathbf{m}(\underline{x}) = \int_{-\infty}^{+\infty} \underline{p}(\underline{x}-\underline{\xi}) \cdot \underline{p}^T(\underline{x}-\underline{\xi})\, W_\xi(\underline{x})\, d\Omega_\xi$ |
| | *Références* | [LIU 95a ; LIU 96a] |
| *Mixte MLS - RKM* | MLS/RK | $C_\xi(\underline{x}) = \underline{p}^T\left(\frac{\underline{\xi}-\underline{x}}{h}\right) \cdot \mathbf{m}^{-1}(\underline{x}) \cdot \underline{p}(\underline{0})$ ($h$ : par ex., rayon du support de $\phi_\xi(\underline{x})$) ; $\mathbf{m}(\underline{x}) = \int_{\underline{\Omega}} \underline{p}\left(\frac{\underline{\xi}-\underline{x}}{h}\right) \cdot p^T\left(\frac{\underline{\xi}-\underline{x}}{h}\right) W_\xi(\underline{x})\, d\Omega_\xi$ |
| | *Références* | [LIU 97] |

***Tabl. 1.1 : Formalismes continus pour les méthodes MLS et RKM.***

***Rappel des principales notations :***

| | |
|---|---|
| $\underline{x}$ | Vecteur des coordonnées spatiales d'un point quelconque |
| $u^R(\underline{x})$ | Approximation continue du champ solution calculée au point $\underline{x}$ |
| $N$ | Nombre total de nœuds (ou particules) sur le domaine d'étude $\Omega$ |
| $\underline{p}^T(\underline{x}) = \{p_j(\underline{x})\}_{j=1}^m$ | Base primaire de dimension $m$ (ex : polynôme ordre $\tilde{m}$), calculée au point $\underline{x}$ |
| $\mathbf{m}(\underline{x})$ | Matrice des moments continus (appellation initialement pour *RKM*) |
| $\underline{\xi}$ | Vecteur des coordonnées spatiales d'une particule quelconque |
| $C_\xi(\underline{x})$ | Fonction de correction associée à la particule $\underline{\xi}$ (appellation *RKM*) |
| $W_\xi(\underline{x})$ | Fonctions de pondération (*MLS*) ou fenêtre (*RKM*) ou fonction de lissage (SPH) associée à la particule $\underline{\xi}$ |

## 1.5 METHODE BASEE SUR LA PARTITION DE L'UNITE : LA METHODE C-PUM

Afin de contourner les problèmes inhérents aux méthodes meshfree *MLS*, à savoir notamment la complexité des calculs des fonctions de forme et de leurs dérivées partielles, leur propriété de non interpolation et donc la difficulté d'imposer les conditions aux limites essentielles, Krysl et Belytschko [KRY 00] proposent de modifier la méthode PUM (« *Partition of Unity Method* », [BAB 95 ; BAB 97 ; MEL 96]), afin d'obtenir des fonctions de forme interpolantes construites à partir des fonctions de Shepard [SHE 68] :

$$u^h\left(\underline{x}\right)=\sum_{I=1}^{N}\frac{\tilde{W}_I\left(\underline{x}\right)}{\sum_{J=1}^{N}\tilde{W}_J\left(\underline{x}\right)}u_I \tag{1.141}$$

où les fonctions de pondérations modifiées $\tilde{W}_I\left(\underline{x}\right)$, par exemple celles définies en (1.44), ont remplacé les fonctions initiales $W_I\left(\underline{x}\right)$ pour obtenir une approximation interpolante.

Cependant, l'approximation ainsi obtenue est uniquement consistante à l'ordre zéro. Pour corriger ce problème, Krysl et Belytschko proposent de modifier les fonctions de forme de manière à obtenir une consistance linéaire. Pour ce faire, ils remplacent la valeur nodale $u_I$ par une fonction nodale $V_I\left(\underline{x}\right)$ :

$$u^h\left(\underline{x}\right)=\sum_{I=1}^{N}\tilde{\phi}_I\left(x\right)V_I\left(\underline{x}\right) \tag{1.142}$$

où l'on a posé :

$$\tilde{\phi}_I\left(x\right)=\frac{\tilde{W}_I\left(\underline{x}\right)}{\sum_{J=1}^{N}\tilde{W}_J\left(\underline{x}\right)} \tag{1.143}$$

$V_I\left(\underline{x}\right)$ est une fonction définie sur un voisinage choisi de manière arbitraire et non nécessairement identique au domaine d'influence. Elle est calculée comme suit :

$$V_I\left(\underline{x}\right)=u_I+\sum_{J=1}^{N_{vI}}C_J^{(I)}\left(\underline{x}\right)\left(u_J-u_I\right) \tag{1.144}$$

avec $N_{vI}$, le nombre de nœuds voisins (« *star nodes* ») du nœud *I*, et $C_J^{(I)}\left(\underline{x}\right)$, une fonction de correction associée au nœud *I* et déterminée de façon à obtenir : $V_I\left(\underline{x}_I\right)=u_I$, soit encore : $C_J^{(I)}\left(\underline{x}_I\right)=0$.

Pour atteindre la consistance linéaire, Krysl et Belytschko proposent de définir la fonction $C_J^{(I)}(\underline{x})$ de la manière suivante :

$$C_J^{(I)}(\underline{x}) = \sum_{i=1}^{n} \alpha_{Ji}^{(I)} \frac{x_i - x_{Ii}}{h_J^{(I)}} \tag{1.145}$$

avec : $h_J^{(I)} = \left\| \underline{x}_J - \underline{x}_I \right\|_2$ et $n$, la dimension de l'espace euclidien.

Les coefficients $\alpha_{Ji}^{(I)}$ sont ensuite déterminés de manière à atteindre la consistance linéaire recherchée, qui se traduit par la condition de reproduction ci-après :

$$\sum_{J=1}^{N_{vI}} \frac{\alpha_{Ji}^{(I)}}{h_J^{(I)}} \left( x_{Jj} - x_{Ij} \right) = \delta_{ij} \qquad 1 \le i, j \le n \tag{1.146}$$

D'après la relation (1.146), il apparaît que la détermination de $C_J^{(I)}(\underline{x})$ et donc de $V_I(\underline{x})$, nécessite l'existence d'au moins $n$ nœuds dans le voisinage du nœud $I$, pour lesquels les vecteurs positions sont linéairement indépendants. Si plus de $n$ nœuds sont utilisés, Krysl et Belytschko proposent de résoudre le système (1.146) en considérant les $n$ premiers coefficients $\left\{ \alpha_{Ji}^{(I)} \right\}_{i,J=1}^{n}$ comme inconnus et donc, en fixant a priori la valeur pour les $(N_{vI} - n)$ coefficients restants.

Dans le cas le cas général, c'est-à-dire pour une distribution non symétrique ou irrégulière des nœuds appartenant au voisinage du nœud $I$, les auteurs proposent les valeurs initiales suivantes :

$$\alpha_{Ji}^{(I)} = \frac{1}{\sum_{K=1}^{N_{vI}} \left( \frac{x_{Ki} - x_{Ii}}{h_K^{(I)}} \right)^2} \frac{x_{Ji} - x_{Ii}}{h_J^{(I)}} \qquad n+1 \le J \le N_{vI}, 1 \le i \le n \tag{1.147}$$

En reportant (1.144) dans (1.142), et en regroupant les $d_I$, les fonctions de forme de la méthode PUM modifiée, méthode que nous avons choisi d'appeler C-PUM, s'écrivent finalement :

$$\phi_I(\underline{x}) = \tilde{\phi}_I(\underline{x}) \left[ 1 - \sum_{J=1}^{N_{vI}} C_J^{(I)}(\underline{x}) \right] + \sum_{J=1}^{A_{vI}} \tilde{\phi}_J(\underline{x}) C_J^{(I)}(\underline{x}) \tag{1.148}$$

où $A_{vI}$ représente le nombre de voisinages auxquels le nœud $I$ contribue.

**Notre contribution :**

La méthodologie proposée par Krysl et Belytschko vise uniquement à atteindre une consistance linéaire (ordre $\tilde{m}=1$). Nous présentons ci-après la généralisation au cas de consistance quelconque ($\tilde{m}\geq 1$).

Pour ce faire, nous définissons la fonction de correction générale suivante :

$$C_J^{(I)}\left(\underline{x}\right)=\underline{p}^T\left(\frac{\underline{x}-\underline{x}_I}{h_J^{(I)}}\right)\cdot\underline{\alpha}_J^{(I)} \tag{1.149}$$

en prenant : $\underline{\alpha}_J^{(I)\,T}=\left\{0\;\alpha_{J2}^{(I)}\ldots\alpha_{Jm}^{(I)}\right\}$.

Pour un nœud donné, les conditions de reproduction des termes $\left\{p_j\left(\underline{x}\right)\right\}_{j=2}^{m}$ de la base primaire polynomiale d'ordre $\tilde{m}$ (consistance d'ordre $\tilde{m}=0$ obtenue par construction), s'écrivent alors :

$$\sum_{J=1}^{N_{vI}} p_j\left(\frac{\underline{x}_J-\underline{x}_I}{h_J^{(I)}}\right)\alpha_{Ji}^{(I)}=\delta_{ij}\qquad 2\leq i,j\leq m \tag{1.150}$$

On définit le voisinage du nœud $I$ de façon à choisir $m-1$ nœuds « proches » dont les vecteurs positions sont linéairement indépendants et on fixe la valeur pour les coefficients supplémentaires de la façon suivante :

$$\alpha_{Ji}^{(I)}=\frac{1}{\sum_{K=1}^{N_{vI}} {p_i}^2\left(\frac{\underline{x}_K-\underline{x}_I}{h_J^{(I)}}\right)}\,p_i\left(\frac{\underline{x}_J-\underline{x}_I}{h_J^{(I)}}\right)\qquad m\leq J\leq N_{vI},\,2\leq i\leq m \tag{1.151}$$

En écrivant le système (1.150) sous forme matricielle, on obtient :

$$\mathbf{P}_I\cdot\underline{A}_i^{(I)}=\underline{B}_i^{(I)}\qquad 2\leq i\leq m \tag{1.152}$$

avec :

$$\mathbf{P}_I=\begin{bmatrix} p_2\left(\frac{\underline{x}_1-\underline{x}_I}{h_1^{(I)}}\right) & \cdots & p_2\left(\frac{\underline{x}_{m-1}-\underline{x}_I}{h_{m-1}^{(I)}}\right)\\ \vdots & \vdots & \vdots\\ p_m\left(\frac{\underline{x}_1-\underline{x}_I}{h_1^{(I)}}\right) & \cdots & p_m\left(\frac{\underline{x}_{m-1}-\underline{x}_I}{h_{m-1}^{(I)}}\right)\end{bmatrix}_{(m-1)\times(m-1)} \tag{1.153}$$

$$B_{ij}^{(I)} = \delta_{ij} - \sum_{J=m}^{N_{vI}} p_j\left(\frac{\underline{x}_1 - \underline{x}_I}{h_1^{(I)}}\right)\alpha_{Ji}^{(I)} \qquad 2 \le i, j \le m \tag{1.154}$$

Les coefficients cherchés s'écrivent finalement :

$$\alpha_{Ji}^{(I)} = \sum_{j=1}^{m-1}\left[\mathbf{P}_I^{-1}\right]_{Jj} B_{ij}^{(I)} \qquad 2 \le i \le m, 2 \le J \le m \tag{1.155}$$

Finalement, les dérivées partielles de $\phi_I$ sont calculées simplement par :

$$\begin{aligned}\phi_{I,j}(\underline{x}) = \tilde{\phi}_{I,j}(\underline{x})\left[1 - \sum_{J=1}^{N_{vI}} C_J^{(I)}(\underline{x})\right] - \tilde{\phi}_I(\underline{x})\sum_{J=1}^{N_{vI}} C_{J,j}^{(I)}(\underline{x}) \qquad 1 \le j \le n \\ + \sum_{J=1}^{A_{vI}}\left[\tilde{\phi}_{J,j}(\underline{x})\, C_J^{(I)}(\underline{x}) + \tilde{\phi}_J(\underline{x})\, C_{J,j}^{(I)}(\underline{x})\right]\end{aligned} \tag{1.156}$$

avec :

$$C_{J,j}^{(I)}(\underline{x}) = \underline{p}_{,j}^{T}\left(\frac{\underline{x} - \underline{x}_I}{h_J^{(I)}}\right)\cdot\underline{\alpha}_J^{(I)} \qquad 1 \le j \le n \tag{1.157}$$

Remarques :

La méthode C-PUM s'apparente à la méthode h-p Clouds, et présente certains avantages par rapport à cette dernière, essentiellement au niveau de la mise en œuvre pratique :

- de part leur construction, les fonctions de forme sont interpolantes ;
- contrairement à la méthode h-p Clouds, il n'est pas nécessaire de recourir à des contributions nodales additionnelles ;
- suivant l'ordre de consistance à atteindre, le calcul de la fonction de correction en un point d'évaluation donné nécessite simplement la connaissance des valeurs aux nœuds voisins (ou « *star nodes* »).

En revanche, contrairement à la méthode h-p Clouds, elle ne permet pas d'enrichir localement l'approximation et permet simplement de construire des fonctions de forme interpolantes. Par ailleurs, nous n'avons pas vraiment constaté de gain de temps au niveau de la procédure de détermination des fonctions de forme interpolantes pour cette méthode ou pour les autres méthodes meshfree.

## 1.6 METHODE BASEE SUR L'INTERPOLATION POLYNOMIALE : LA METHODE PIM

Afin de contourner les problèmes inhérents aux méthodes meshfree *MLS*, à savoir notamment la complexité des calculs des fonctions de forme et de leurs dérivées partielles, leur propriété de non interpolation et donc la difficulté d'imposer les conditions aux limites essentielles, Liu et Gu [LIU 99 ; LIU 01a] ont proposé une nouvelle méthode meshfree, qu'ils ont appelé « *Point Interpolation Method* », qui leur permet de construire directement des fonctions de forme interpolantes.

Pour cela, l'approximation locale $u^h$, déterminée sur le domaine d'influence $\Omega_\xi$ d'un point d'évaluation $\underline{\xi}$ donné, est écrite comme étant une combinaison linéaire des termes de la base polynomiale $\underline{p} = \{p_i\}_{i=1}^{N_\xi}$ d'ordre $\tilde{m}$ (équivalente de la base primaire polynomiale pour l'approximation *MLS*, avec $m = N_\xi$), où $N_\xi$ représente le nombre de nœuds voisins de $\underline{\xi}$ :

$$\forall \underline{x} \in \overline{\Omega}, \quad u^h(\underline{x}) = \sum_{I=1}^{N_\xi} \phi_I(\underline{x})\, u_I = \underline{p}^T(\underline{x}) \cdot \underline{\alpha}(\underline{\xi}) \tag{1.158}$$

Les composantes de $\underline{\alpha}$ sont déterminés de manière à vérifier la condition d'interpolation cherchée, c'est-à-dire :

$$u^h(\underline{x}_I) = u_I = \sum_{j=1}^{N_\xi} p_j(\underline{x}_I)\, \alpha_j(\underline{\xi}) \tag{1.159}$$

En notant $\underline{u}^{eT} = \{u_I\}_{I=1}^{N_\xi}$ et $\underline{\alpha}^{eT} = \{\alpha_j\}_{j=1}^{N_\xi}$, et en posant $m = N_\xi$, les vecteurs respectivement des valeurs nodales et des coefficients cherchés sur $\Omega_\xi$, alors on peut écrire (1.159) sous forme matricielle, ce qui permet d'obtenir l'expression de $\underline{\alpha}^e$ :

$$\underline{\alpha}^e = \mathbf{P}^{-1} \cdot \underline{u}^e \tag{1.160}$$

avec **P**, la matrice locale des moments identique à celle de l'approximation *MLS* (voir la relation (1.11)) pour $m = N$.

Ainsi, le vecteur des fonctions de forme PIM sur $\Omega_\xi$, noté $\underline{\phi}^{eT}(\underline{x}) = \{\phi_I(\underline{x})\}_{I=1}^{m}$, s'écrit simplement, à condition que **P** ne soit pas singulière :

$$\underline{\phi}^{eT}(\underline{x}) = \underline{p}^T(\underline{x}) \cdot \mathbf{P}^{-1} \tag{1.161}$$

Soit également en explicitant au nœud $I$ :

$$\phi_I(\underline{x}) = \sum_{j=1}^{m} p_j(\underline{x}) \left[\mathbf{P}^{-1}\right]_{jI} \tag{1.162}$$

Les fonctions de forme ainsi construites sont donc interpolantes ($\phi_I(\underline{x}_J) = \delta_{IJ}$), ce qui permet d'imposer les conditions aux limites essentielles directement. Par ailleurs, elles forment une partition de l'unité sur $\Omega_\xi$ : $\sum_{I=1}^{m} \phi_I(\underline{x}) = 1, \forall \underline{x} \in \Omega_\xi$. Enfin, les dérivées partielles sont directement calculables :

$$\partial_i \phi_I(\underline{x}) = \sum_{j=1}^{m} \partial_i p_j(\underline{x}) \left[\mathbf{P}^{-1}\right]_{jI} \qquad 1 \le i \le n \tag{1.163}$$

La technique de construction des fonctions de forme PIM est en fait identique à celle utilisée dans le cas des éléments finis isoparamétriques, simplement l'évaluation est réalisée sur des supports pouvant se chevaucher, au lieu de l'être sur des éléments prédéfinis et juxtaposés. Néanmoins, nous avons constaté que cette méthode est difficile d'utilisation dans les cas 2D et 3D, du fait des contraintes rencontrées pour la construction de la base primaire c'est-à-dire la recherche des voisins adéquats (nombre et positions).

Remarque :

De la même manière que pour l'approximation *MLS*, $\mathbf{P}_e$ peut devenir non inversible lorsque l'arrangement spatial des nœuds sur $\Omega_\xi$ est singulier (*cf.* Fig. 1.21). Nous verrons au chapitre 2, les différentes techniques proposées par Liu *et al.* pour y remédier [LIU 01a ; LIU 03], et leur incidence sur la précision numérique des résultats.

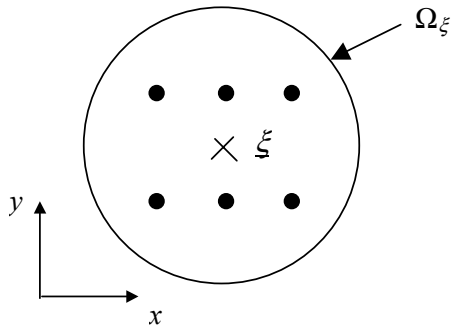



***Fig. 1.21 : Arrangement nodal 2D singulier pour*** $\underline{p}^T(\underline{x}) = \{1, x, y, x^2, xy, y^2\}$.

## 1.7 METHODES UTILISANT LES FONCTIONS DE BASES RADIALES

### 1.7.1 La méthode RPIM

#### *1.7.1.1 Principe de l'interpolation avec des fonctions de bases radiales à support global*

Les fonctions de bases radiales à support global (GSRBF) sont classiquement utilisées depuis plus d'une vingtaine d'années pour l'interpolation de données spatiales dispersées [FRA 82 ; POW 90 ; FAS 98]. Ainsi, l'on cherche à construire une interpolation de la fonction $u^h(\underline{x})$ à partir des couples $(\underline{x}_I, u_I) \in \mathbb{E} \times \mathbb{R}$ pour $1 \le I \le N$ ($\mathbb{E} = \mathbb{R}^n$), où $N$ représente le nombre de points (encore appelés « centres » ou nœuds) de vecteur position $\underline{x}_I$ et $u_I$, la valeur connue de $u^h(\underline{x})$ au nœud $\underline{x}_I$. Dans ce cas, l'approximation globale cherchée sera exprimée par :

$$u^h(\underline{x}) = \sum_{I=1}^{N} \alpha_I B\left(\left\|\underline{x} - \underline{x}_I\right\|_2\right) \tag{1.164}$$

où $B$ représente la fonction de base radiale à valeurs réelles, définie sur $\mathbb{E}$, et $\{\alpha_I\}_{I=1}^{N}$, l'ensemble des coefficients cherchés de manière à satisfaire les conditions d'interpolation : $\forall I, u^h(\underline{x}_I) = u_I$.

Le principal intérêt des fonctions de bases radiales est qu'elles permettent de simplifier une analyse au départ multidimensionnelle, par une transformation simple sur un espace à une dimension.

Depuis les années 90, ces fonctions sont également utilisées en association avec les techniques de collocation pour résoudre les équation aux dérivées partielles (EDP) [KAN 90a ; KAN 90b ; FRA 98a ; FRA 98b ; FAS 97 ; FAS 99 ; FAS 01 ; ZER 00 ; ZHA 00], notamment dans les problèmes de mécanique des fluides. Cette approche présente l'avantage de conduire à une approximation globale directe de la solution. L'inconvénient majeur est qu'elle conduit à une matrice dense pour le système à résoudre et donc à des performances de calculs moindres. Différentes techniques de résolution ont été envisagées pour cet inconvénient, notamment la partition par domaines [DUB 94 ; ISK 02].

Par ailleurs, le choix des GSRBF pour la résolution des EDP est déterminant. En effet, certaines fonctions présentent l'avantage d'être inconditionnellement définies positives (*cf.* §1.5.2.3), tandis que d'autres le sont sous certaines conditions. Dans ce cas, hormis les problèmes de mauvais conditionnement de la matrice de rigidité du système, cette dernière peut également devenir singulière (non définie positive) et/ou non symétrique. Il est alors nécessaire d'utiliser un polynôme additionnel $\underline{p}^T(\underline{x}) = \{p_j(\underline{x})\}_{j=1}^{m}$ d'ordre au plus $(\tilde{m}-1)$ sur $\mathbb{E}$, permettant ainsi d'obtenir des fonctions radiales définies positives [WU 93 ; WEN 01 ; CHE 02a].

Dans ce cas, l'approximation globale (1.164) sera exprimée par :

$$u^h(\underline{x}) = \sum_{I=1}^{N} \alpha_I B\left(\left\|\underline{x} - \underline{x}_I\right\|_2\right) + \sum_{j=1}^{m} \beta_j p_j(\underline{x}) \tag{1.165}$$

On parle dans ce cas de fonctions radiales conditionnellement (ou strictement) définies positives d'ordre $\tilde{m}$. Par ailleurs, en plus des conditions d'interpolation, le problème admet une solution à condition de vérifier les contraintes supplémentaires :

$$\sum_{I=1}^{N} \alpha_I p_j(\underline{x}_I) = 0 \qquad \text{pour } j = 1,...,m \tag{1.166}$$

C'est la technique adoptée notamment par Zerroukat et co-auteurs [ZER 00] pour résoudre les EDP par collocation dans le cas de problèmes d'advection - diffusion.

C'est également la méthode utilisée par Wang et co-auteurs [WAN 00 ; WAN 02a ; WAN 02b ; LIU 03] pour une résolution de type Galerkin notamment dans le cas des problèmes de consolidation. Les auteurs nomme cette méthode RPIM, car ils sont en fait partis de la base polynomial de la méthode PIM, à laquelle ils ont ajouté une base utilisant des fonctions radiales GSRBF. Cette technique présente l'intérêt d'être interpolante, et contrairement à la méthode PIM, la construction de la base primaire se fait ici de manière simple et rapide.

### *1.7.1.2 Fonctions de forme RPIM*

Si l'on reprend les notations des sections précédentes et en partant de la relation (1.165), le champ approché $u^h(\underline{x})$ en tout point $\underline{x}$ du domaine $\overline{\Omega}$, s'exprime selon la relation :

$$u^h(\underline{x}) = \sum_{I=1}^{N} \alpha_I B_I(\underline{x}) + \sum_{j=1}^{m} \beta_j p_j(\underline{x}) \tag{1.167}$$

en notant : $B_I(\underline{x}) = B(r)$ et $r = \left\|\underline{x} - \underline{x}_I\right\|_2$.

Les coefficients $\alpha_I$ et $\beta_I$ sont déterminés de manière à satisfaire les $N$ conditions d'interpolation : $u^h(\underline{x}_I) = u_I$, auxquelles s'ajoutent les $m$ équations suivantes :

$$\sum_{I=1}^{N} \alpha_I p_j(\underline{x}_I) = 0 \qquad 1 \le j \le m \tag{1.168}$$

Le système linéaire à résoudre est alors le suivant :

$$\begin{bmatrix} \mathbf{B} & \mathbf{P} \\ \mathbf{P^T} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \underline{\alpha} \\ \underline{\beta} \end{Bmatrix} = \begin{Bmatrix} \underline{U} \\ \underline{0} \end{Bmatrix} \tag{1.169}$$

avec : $\underline{\alpha}^T = \{\alpha_I\}_{I=1}^N$, $\underline{\beta}^T = \{\beta_I\}_{I=1}^N$ et $\underline{U}^T = \{u_I\}_{I=1}^N$.

Par ailleurs, **P** est donnée en (1.11) , et l'on note :

$$\mathbf{B}=\begin{bmatrix} B_1(\underline{x}_1) & \cdots & B_N(\underline{x}_1) \\ \vdots & \vdots & \vdots \\ B_1(\underline{x}_N) & \cdots & B_N(\underline{x}_N) \end{bmatrix}_{N\times N} \tag{1.170}$$

Soit la matrice **G** définie par :

$$\mathbf{G}=\begin{bmatrix} \mathbf{B} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \tag{1.171}$$

Alors le système (1.169) admet une solution unique à condition de supposer que : $m \le N$ et si $\mathbf{B}^{-1}$ (donc $\mathbf{G}^{-1}$) existe. Dans ce cas, si l'on écrit $u^h(\underline{x})=\sum_{I=1}^{N}\phi_I(\underline{x})\,u_I$ , alors les fonctions de forme $\phi_I(\underline{x})$ sont données par :

$$\phi_I(\underline{x})=\sum_{J=1}^{N}B_J(\underline{x})\left[\mathbf{G}^{-1}\right]_{JI}+\sum_{\substack{j=1\\k=j+N}}^{m}p_j(\underline{x})\left[\mathbf{G}^{-1}\right]_{kI} \tag{1.172}$$

où $\left[\mathbf{G}^{-1}\right]_{kI}$ représente le terme d'indices $(k,I)$ de la matrice $\mathbf{G}^{-1}$.

La matrice **G** étant constante, les dérivées spatiales des fonctions de forme s'écrivent simplement :

$$\phi_{I,i}(\underline{x})=\sum_{J=1}^{N}B_{J,i}(\underline{x})\left[\mathbf{G}^{-1}\right]_{JI}+\sum_{\substack{j=1\\k=j+N}}^{m}p_{j,i}(\underline{x})\left[\mathbf{G}^{-1}\right]_{kI} \tag{1.173}$$

### *1.7.1.3 Fonctions GSRBF usuelles*

Nous rappelons ci-après les fonctions de bases radiales à support global les plus utilisées en pratique. Les fonctions inconditionnellement définies positives correspondent en fait à des fonctions définies positives d'ordre $\tilde{m}=0$.

**Fonctions inconditionnellement définies positives :**

***Gauss :***

$$B(r)=e^{-br^2},\ b>0 \tag{1.174}$$

***Multi-Quadrics inverse :***

$$B(r)=\left(r^2+c^2\right)^{-\beta/2} \qquad c>0,\beta\in\mathbb{R}^+\setminus 2\mathbb{N} \tag{1.175}$$

Généralement, on prend $\beta = 0$. $c$ est une constante de lissage dépendant du nombre de nœuds et de l'arrangement spatial [CAR 91].

**Fonctions conditionnellement définies positives d'ordre $\tilde{m}$ :**

***Multi-Quadrics :***

$$B(r) = \left(r^2 + c^2\right)^{\beta/2} \qquad c > 0, \beta \in \mathbb{R}^+ \setminus 2\mathbb{N}, \tilde{m} \geq \lceil \beta/2 \rceil$$

***Thin-Plate spline :***

$$B(r) = \begin{cases} (-1)^{\beta+1} r^{2\beta} \log r & \beta \in \mathbb{N}, \tilde{m} > \beta \\ (-1)^{\lceil \beta/2 \rceil} r^{\beta} & \beta \in \mathbb{R}^+ \setminus 2\mathbb{N}, \tilde{m} \geq \lceil \beta/2 \rceil \end{cases} \tag{1.176}$$

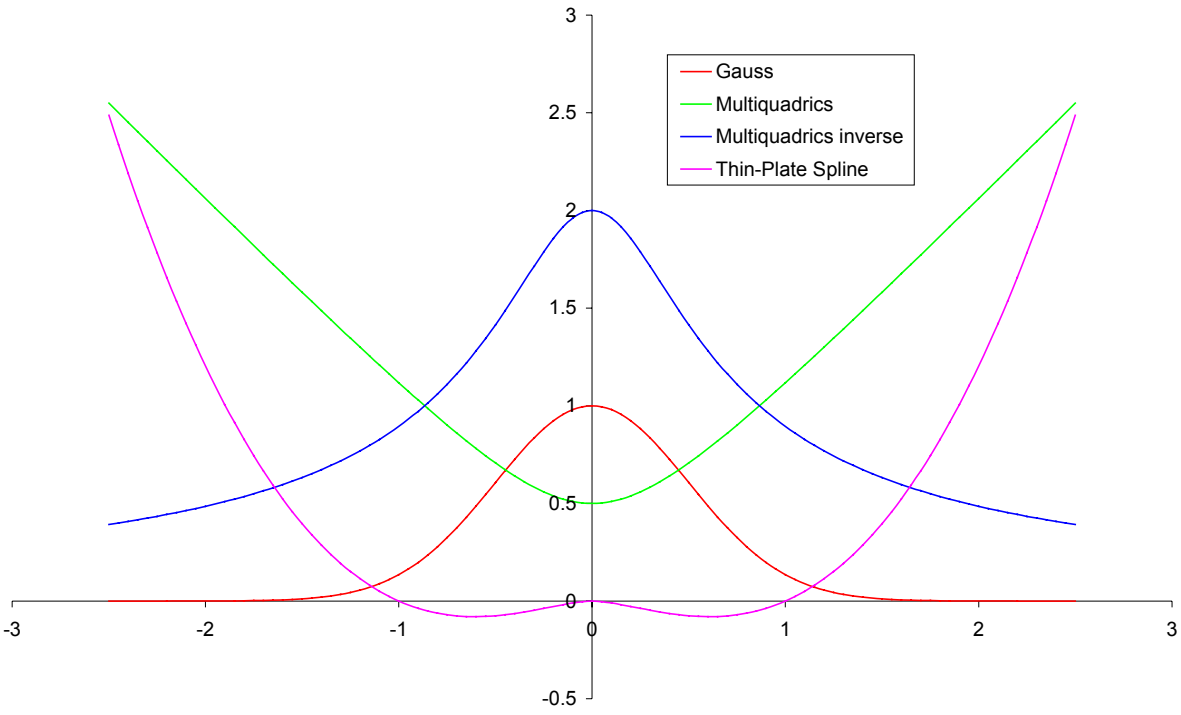



***Fig. 1.22 : Fonctions GSRBF calculées pour le point d'abscisse nulle : Multiquadrics ( $\beta = 1, c = 0.5$ ), Thin-Plate Spline ( $\beta = 1$ ), Gauss ( $b = 2$ )***

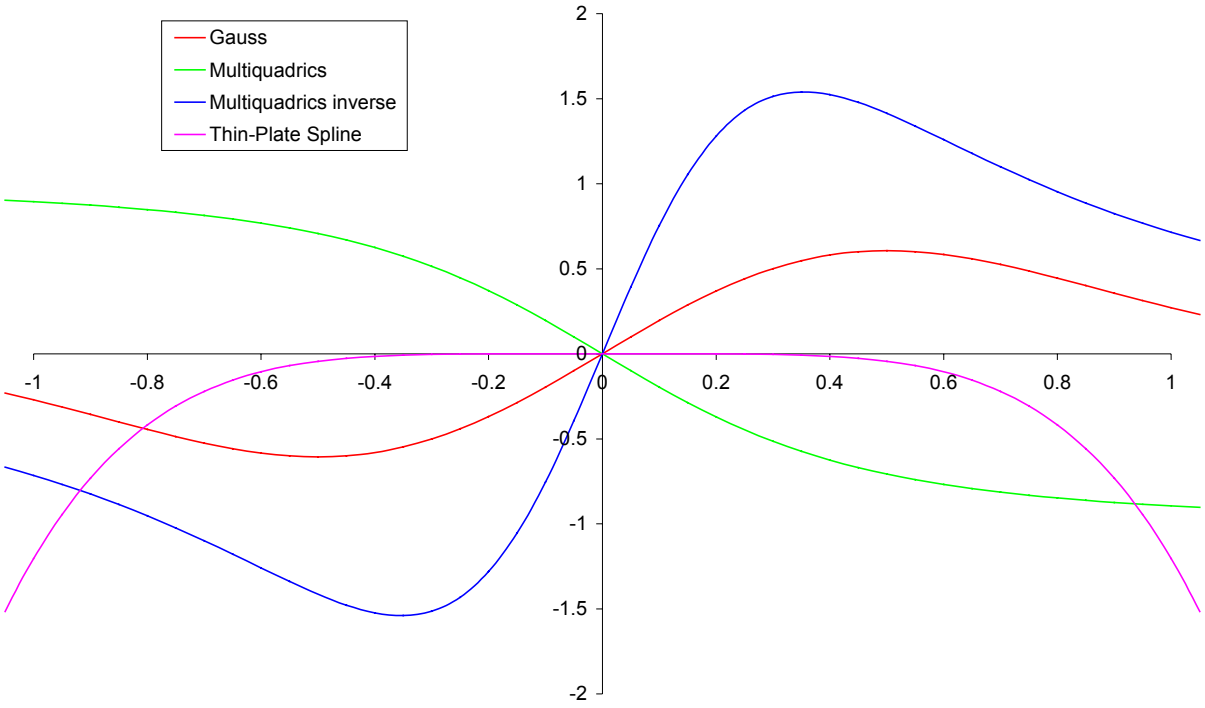



***Fig. 1.23 : Dérivées des fonctions GSRBF calculées pour le point d'abscisse nulle : Multiquadrics ( $\beta = 1, c = 0.5$ ), Thin-Plate Spline ( $\beta = 1$ ), Gauss ( $b = 2$ )***

**Notre contribution :**

Wang et co-auteurs [WAN 00 ; WAN 02a ; WAN 02b ; LIU 03] utilisent la méthode RPIM uniquement avec des fonctions GSRBF, ce qui à notre avis présente l'inconvénient majeur de générer des matrices globales pleines pour la résolution des problèmes aux limites de taille importante. De manière générale, nous avons donc préféré utiliser des fonctions de base radiale à support compact (CSRBF), dont le principe est expliqué dans la section §1.7.2.1. ci-après.

## 1.7.2 La méthode MLS/RBF

### *1.7.2.1 Principe de l'interpolation avec des fonctions de bases radiales à support compact*

Une autre classe de fonctions définies positives, à savoir les fonctions de bases radiales à supports compacts (CSRBF), est également apparue à partir du milieu des années 90 [WU 95 ; WEN 95 ; WEN 99], pour essayer de pallier les inconvénients des fonctions GSRBF.

De même que pour les fonctions de pondération usuelles pour les autres méthodes meshfree, les fonctions CSRBF vérifient :

$$B_I\left(\underline{x}\right)\begin{cases}>0 & \text{si } \underline{x}\in\Omega_I\\ =0 & \text{si } \underline{x}\notin\Omega_I\end{cases} \tag{1.177}$$

en notant également :

$$\begin{cases}B_I\left(\underline{x}\right)=B\left(r\right)\\ r=r\left(\underline{x}-\underline{x}_I,\underline{h}_I\right)\end{cases} \tag{1.178}$$

où $r$ est défini de la même manière que pour les méthodes *MLS* (*cf.* §1.3.1).

L'approximation ainsi obtenue s'écrit de la même façon que celles utilisant des fonctions GSRBF, à savoir :

$$u^h\left(\underline{x}\right)=\sum_{I=1}^{N}\alpha_I B_I\left(r\right) \tag{1.179}$$

Sachant que la matrice **B** donnée en (1.170), est définie positive par construction, les fonctions de forme $\phi_I(\underline{x})$ sont finalement données par :

$$\phi_I\left(\underline{x}\right)=\sum_{J=1}^{N}B_J\left(\underline{x}\right)\left[\mathbf{B}^{-1}\right]_{JI} \tag{1.180}$$

Remarque :

Cette méthode n'est pas interpolante et nécessite donc d'utiliser l'une des techniques particulières mentionnées précédemment pour le devenir (couplage avec des éléments finis ou multiplicateurs de Lagrange). A notre avis, il est plus avantageux d'utiliser la méthode RPIM avec des fonctions CSRBF, car elle est d'une part interpolante par construction, et d'autre part, elle profite des propriétés intéressantes des fonctions CSRBF (matrice global non pleine, définie positive).

### *1.7.2.2 Fonctions de forme MLS/RBF*

Très récemment, un certain nombre de chercheurs se sont penchés sur l'utilisation de fonctions CSRBF dans le cadre des méthodes de partition de l'unité [FAS 02 ; WEN 02]. Ainsi, Fasshauer [FAS 02] construit une approximation *MLS*, à partir des fonctions CSRBF. Dans ce mémoire, cette méthode sera nommée MLS/RBF. De façon analogue à la démarche adoptée pour la méthode C-PUM, il construit tout d'abord ses fonctions de forme à partir des fonctions de Shepard [SHE 68], qu'il modifie ensuite en utilisant des fonctions CSRBF pour obtenir un ordre de consistance supérieur.

En effet, du fait des propriétés des CSRBF, il montre que pour une fonction solution $u(\underline{x})$ continûment dérivable de classe $C^{\tilde{m}+1}(\bar{\Omega})$, et en utilisant des fonctions de forme de type Shepard :

$$\phi_I(\underline{x}) = \frac{W_I(\underline{x})}{\sum_{J=1}^{N} W_J(\underline{x})} \qquad \forall \underline{x} \in \bar{\Omega} \tag{1.181}$$

Alors l'approximation globale $u^h(\underline{x})$ est d'ordre $(\tilde{m}+1)$, à condition d'écrire les fonctions de pondération sous la forme suivante :

$$W_I(\underline{x}) = B_I(\underline{x})\, C_I(\underline{x}) \quad \forall \underline{x} \in \bar{\Omega} \tag{1.182}$$

où $C_I(\underline{x})$ représente un polynôme de degré $\tilde{m}$, et à condition également de vérifier les conditions de reproduction suivantes :

$$\begin{cases} M_k(\underline{x}) = \sum_{I=1}^{N} W_I(\underline{x}) \prod_{j=1}^{n} \left(x_{Ij} - x_j\right)^{k_j} = \prod_{j=1}^{n} \delta_{0k_j} & 0 \le k \le \tilde{m} \\ \sum_{j=1}^{n} k_j = k \end{cases} \tag{1.183}$$

On voit en fait que les conditions de reproduction (1.183) et (1.89) sont identiques et donc que $M_k(\underline{x})$ correspond au moment discret d'ordre *k* introduit dans l'approximation *RKM*.

Par ailleurs, le polynôme $C_I(\underline{x})$ correspond à la fonction de correction introduite dans la méthode RKPM.

La condition $M_0(\underline{x}) = 1$ assure que les fonctions $W_I$ forment une partition de l'unité, ce qui en simplifiant dans (1.181), permet finalement d'écrire les fonctions de forme MLS/RBF :

$$\phi_I(\underline{x}) = B_I(\underline{x})\, C_I(\underline{x}) \quad \forall \underline{x} \in \bar{\Omega} \tag{1.184}$$

Cependant, Fasshauer [FAS 99 ; FAS 01] montre que l'utilisation des CSRBF, en tout cas dans le cadre d'une résolution par collocation, requiert généralement la mise en œuvre d'algorithmes spécifiques pour maintenir l'équilibre entre les niveaux de précision et de

performances numériques à atteindre (exemple : largeur de bande relativement faible et/ou constante pour la matrice de rigidité). Il préconise notamment d'utiliser un algorithme hiérarchique ou « *multilevel algorithm* » [FLO 96 ; FAS 01]. Dans cet algorithme, une suite de grilles imbriquées (ou des ensembles de nœuds) est utilisée pour actualiser la solution selon un processus itératif de type Newton. Fasshauer utilise cet algorithme indifféremment avec une approximation de type *MLS* ou avec une méthode de type fonction de base radiale (*cf*. Tabl. 1.2).

Enfin, Schaback et Wendland [SCH 00] préfèrent quant à eux, utiliser un algorithme adaptatif de type « *greedy algorithm* », qui leur permet une résolution sur des sous-ensembles de données restreints successifs, pour lesquels la solution optimale est recherchée.

Pour notre part, nous avons implémenté cette méthode dans le cadre d'une résolution de type Galerkin avec quadrature de Gauss-Legendre (*cf*. applications numériques du chapitre 2).

- *Création de K ensembles imbriqués de noeuds* :

$$S_1 \subset \cdots \subset S_K = \left\{ \underline{x}_1, \cdots, \underline{x}_{N^K} \right\} \subset \mathbb{R}^n$$

- *Initialisation* : $u_0^h\left(\underline{x}\right) = 0$
- *Pour* $k = 1, 2, \cdots, K$ *faire* $\left(sur\ S_k\right)$

1. *Résoudre le problème d'interpolation* : $r_k\left(\underline{x}\right) = u_k^h\left(\underline{x}\right) - u_{k-1}^h\left(\underline{x}\right)$

$$CSRBF : \quad r_k\left(\underline{x}\right) = \sum_{I=1}^{N^k} \alpha_I B_I\left(r\right)$$

$$MLS : \qquad r_k\left(\underline{x}\right) = \sum_{I=1}^{N^k} r_k\left(\underline{x}_I\right) \cdot B_I\left(r\right)$$

2. *Mettre à jour* : $u_k^h\left(\underline{x}\right) = u_{k-1}^h\left(\underline{x}\right) + r_k\left(\underline{x}\right)$

***Tabl. 1.2 : Algorithme hiérarchique***

### *1.7.2.3 Fonctions CSRBF usuelles*

Dans les fonctions présentées ci-après, nous utilisons la notation $(r)_+$ pour tout $r$ de $\mathbb{R}^+$, définie par la relation :

$$(r)_+ = \begin{cases} r & \text{si } 0 \le r \le 1 \\ 0 & \text{si } r > 1 \end{cases} \tag{1.185}$$

***Wendland :***

Les fonctions introduites par Wendland [WEN 95], notées $B_{l,s}$, sont sans doute les fonctions CSRBF les plus utilisées en pratique. Elles sont définies positives sur $\mathbb{R}^d$ ($d \le n$) sous certaines conditions et s'expriment sous forme explicite en fonction des paramètres $s$ et $l = \lfloor d/2 \rfloor + s + 1$, où $2s$ représente le degré de continuité de la fonction ($C^{2s}\left(\mathbb{R}^d\right)$). Elles sont par ailleurs définies à un facteur multiplicatif constant près :

$$B_{l,0}(r) = (1-r)_+^l \tag{1.186}$$

$$B_{l,1}(r) = (1-r)_+^{l+1}\left[1+(l+1)r\right] \tag{1.187}$$

$$B_{l,2}(r) = (1-r)_+^{l+2}\left[1+(l+2)r+\frac{1}{3}\left(l^2+4l+3\right)r^2\right] \tag{1.188}$$

$$B_{l,3}(r) = (1-r)_+^{l+3}\left[1+(l+3)r+\frac{1}{5}\left(2l^2+12l+15\right)r^2+\frac{1}{15}\left(l^3+9l^2+23l+15\right)r^3\right] \tag{1.189}$$

Les dérivées par rapport à la variable $r$ s'expriment simplement par :

$$B'_{l,0}(r) = -l(1-r)_+^{l-1} \tag{1.190}$$

$$B'_{l,1}(r) = -(l+1)(l+2)rB_{l,0}(r) \tag{1.191}$$

$$B'_{l,2}(r) = -\frac{(l+4)(l+3)r}{3}B_{l,1}(r) \tag{1.192}$$

$$B'_{l,3}(r) = -\frac{(l+6)(l+5)r}{5}B_{l,2}(r) \tag{1.193}$$

***Buhmann [BUH 01] :***

$$B(r) = \frac{1}{6} - 2r^2 + \frac{16}{3}r^3 - \frac{7}{2}r^4 + 2r^4 \log r \tag{1.194}$$

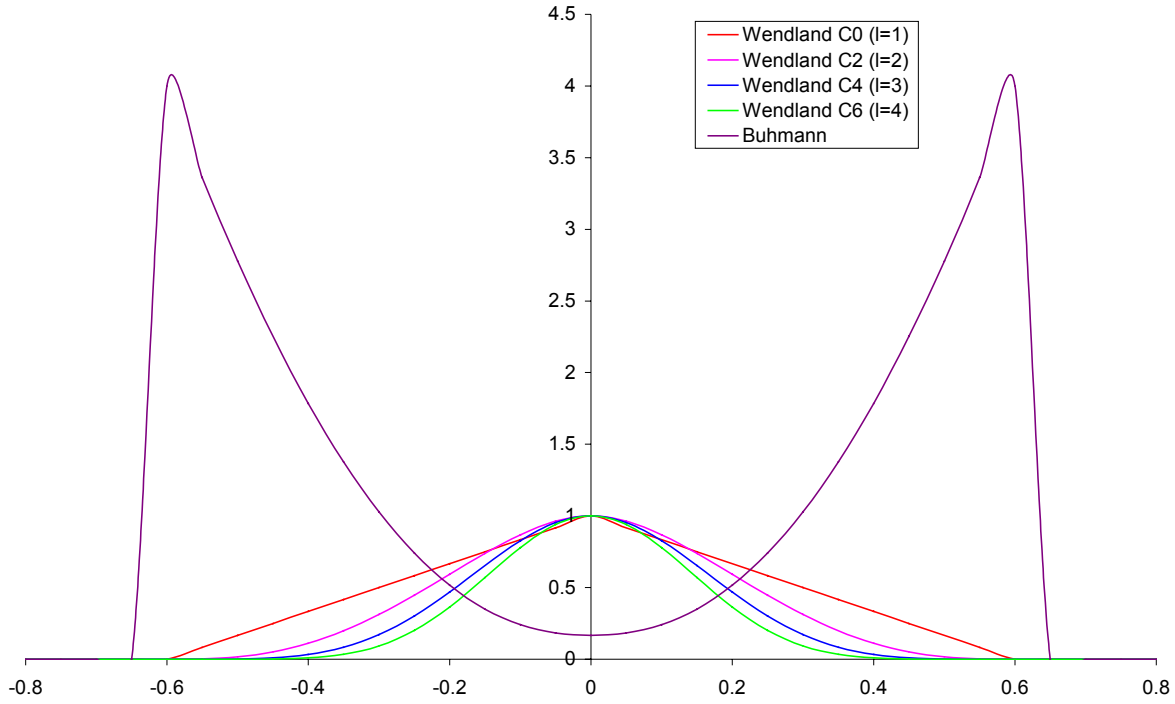



***Fig. 1.24 : Fonctions CSRBF calculées pour le point d'abscisse nulle : Wendland ($d=1$) et Buhmann***

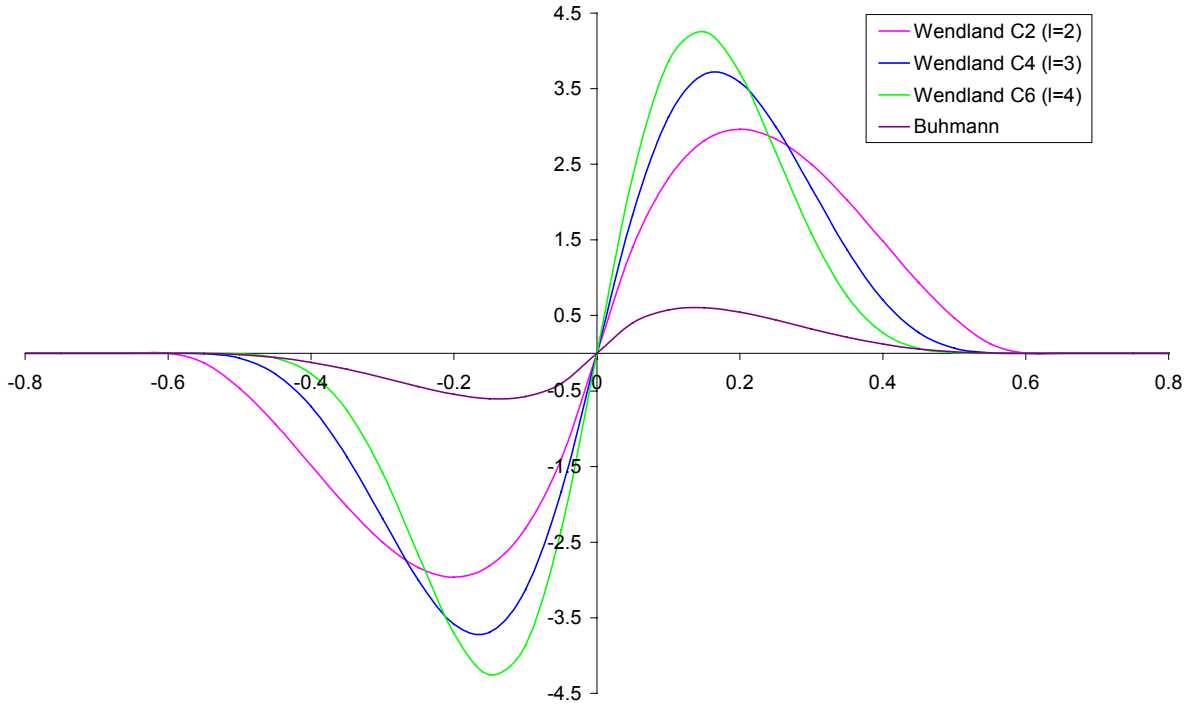



***Fig. 1.25 : Dérivées des fonctions CSRBF calculées pour le point d'abscisse nulle : Wendland ($d=1$) et Buhmann***

## 1.8 CONCLUSIONS ET SYNTHESE DES FORMALISMES MESHFREE

Dans ce chapitre, nous avons présenté le formalisme des principales familles de méthodes meshfree actuellement les plus utilisées en pratique, sachant que la liste n'est pas exhaustive, du fait d'une littérature extrêmement abondante sur le sujet. Nous avons choisi de regrouper les différentes méthodes arbitrairement par type d'approximation, à savoir les approximations *MLS* (moindres carrés mobiles) et *RKM* (« reproducing kernel »), l'interpolation polynomiale et les approximations *RBF* utilisant des fonctions de bases radiales. Ce regroupement est arbitraire, car un certain nombre de méthodes sont en fait à l'intersection entre deux groupes (exemple : MLS/RK, RPIM, MLS/RBF). Par ailleurs, la plupart des méthodes meshfree présentées, à l'exception des méthodes *RBF* (méthode MLS/RBF non comprise), s'inscrivent dans le cadre de la partition de l'unité de par la construction de leurs fonctions de forme. Nous proposons ci-après la synthèse des formalismes discrets présentés dans ce chapitre (*cf*. Tabl. 1.3, 1.4 et 1.5).

Pour les approximations *MLS* et *RKM*, l'ordre de consistance (ou reproduction) est conditionné par l'ordre de dérivabilité des fonctions de pondération. Par ailleurs, les fonctions de forme sont généralement non interpolantes, sauf sous certaines conditions, ce qui nécessite d'utiliser de techniques particulières pour permettre d'imposer les conditions aux limites de Dirichlet. Dans le cas d'une résolution de type Galerkin, une technique pratique consiste à coupler avec des éléments finis près de la frontière et de mettre en œuvre une procédure de transition entre les domaines meshfree et éléments finis. Une autre technique proposée récemment pour l'approximation *RKM* consiste à enrichir les fonctions de forme pour qu'elles deviennent interpolantes (sans utiliser les éléments finis). Ces deux techniques présentent l'avantage de ne pas augmenter le nombre de degrés de liberté du système, contrairement à la technique classique qui consiste à employer des multiplicateurs de Lagrange.

L'approximation polynomiale (méthode PIM), qui peut être considérée comme une généralisation des éléments finis, construit des fonctions de forme interpolantes, ce qui permet d'imposer les conditions aux limites directement aux nœuds. L'inconvénient est que la détermination de l'ordre de la base polynomiale est en pratique difficile, car variable suivant l'arrangement spatial des nœuds (nombre de nœuds voisins du point d'évaluation et positions).

Les méthodes utilisant des fonctions de bases radiales à support global (GSRBF) sont intéressantes, car elles sont interpolantes et sont en outre faciles à mettre en œuvre dans un code numérique. Les méthodes utilisant des fonctions à support global possèdent des ordres de convergence élevés, mais en revanche, produisent des matrices de rigidité pleines ou mal conditionnées, voire non définies positives. Elles nécessitent de ce fait l'emploi d'une algorithmique spécifique pour y remédier (méthodes de pré-conditionnement, partition par domaines, etc.). Les méthodes avec fonctions de bases radiales à support compact (CSRBF) ne présentent pas les mêmes inconvénients, mais conduisent généralement à une précision numérique moins élevée. Un moyen possible d'y remédier est d'utiliser un algorithme hiérarchique et/ou de construire des fonctions de forme de type Shepard, modifiées pour atteindre l'ordre de consistance recherché.

Par ailleurs, nous avons implanté dans un même outil de calculs, la plupart des méthodes meshfree présentées dans ce chapitre. Le second chapitre de ce mémoire est consacré à la comparaison des performances numériques (convergence, précision, etc.) des différentes méthodes pour quelques applications standards.

| **Approximations** | **Principales Méthodes meshfree** | **Formalisme discret :** $u^h(\underline{x}) = \sum_{I=1}^{N} \phi_I(\underline{x})\, d_I$ ; $\phi_I(\underline{x}) = C_I(\underline{x})\, W_I(\underline{x})$ |
|---|---|---|
| *MLS* | DEM<br>EFGM<br>*Références* | $C_I(\underline{x}) = \underline{p}^T(\underline{x}) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \underline{p}(\underline{x}_I)$<br>$\mathbf{M}(\underline{x}) = \sum_{I=1}^{N} W_I(\underline{x})\, \underline{p}(\underline{x}_I) \cdot \underline{p}^T(\underline{x}_I)$<br>[NAY 91a ; NAY 91b ; NAY 92 ; BEL 94] |
| *RKM* | SPH<br>*Références* | $C_I(\underline{x}) = 1$<br>[LUC 77 ; GIN 77 ; MON 82] |
| | RKPM<br>*Références* | $C_I(\underline{x}) = \underline{p}^T(\underline{x} - \underline{x}_I) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \underline{p}(\underline{0})$<br>$\mathbf{M}(\underline{x}) = \sum_{I=1}^{N} \underline{p}(\underline{x} - \underline{x}_I) \cdot \underline{p}^T(\underline{x} - \underline{x}_I)\, W_I(\underline{x})$<br>[LIU 95a ; LIU 95b ; LIU 96a ; LIU 96b] |
| | RKI<br>*Références* | $\phi_I(\underline{x}) = N_I(\underline{x}) + W_I(\underline{x}) \left[ C_I(\underline{x}) + \underline{p}^T(\underline{x} - \underline{x}_I) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \sum_{J=1}^{N} \underline{p}(\underline{x} - \underline{x}_J) N_J(\underline{x}) \right]$<br>$C_I(\underline{x})$ et $\mathbf{M}(\underline{x})$ : idem RKPM<br>[CHE 03] |
| *Mixte MLS - RKM* | MLS/RK<br>*Références* | $C_I(\underline{x}) = \underline{p}^T\left(\frac{\underline{x}_I - \underline{x}}{h}\right) \cdot \mathbf{M}^{-1}(\underline{x}) \cdot \underline{p}(\underline{0})$<br>( $h$ : par ex., rayon du support de $\phi_I(\underline{x})$ )<br>$\mathbf{M}(\underline{x}) = \sum_{I=1}^{N} \underline{p}\left(\frac{\underline{x}_I - \underline{x}}{h}\right) \cdot \underline{p}^T\left(\frac{\underline{x}_I - \underline{x}}{h}\right) W_I(\underline{x})$<br>[LIU 97 ] |
| *Mixte MLS – RBF* | MLS/RBF<br>*Références* | $C_I(\underline{x})$ : idem RKPM<br>$W_I(\underline{x}) = B_I(\underline{x})$ (CSRBF)<br>[FAS 02] |

***Tabl. 1.3 : Formalismes discrets pour les méthodes MLS et RKM.***

| **Approximations** | **Principales Méthodes meshfree** | **Formalisme discret :** $u^h(\underline{x}) = \sum_{I=1}^{N} \phi_I(\underline{x})\, d_I$ |
|---|---|---|
| *RBF* | RBF | $\phi_I(\underline{x}) = \sum_{J=1}^{N} B_J(\underline{x}) \left[\mathbf{B}^{-1}\right]_{JI}$ <br> $\mathbf{B}_{IJ} = B_J(x_I) \quad I,J = 1,N$ <br> $B_J(\underline{x})$ : CSRBF |
| | *Références* | [WU 95 ; WEN 95 ; WEN 99] |
| *Polynomiale* | PIM | $\phi_I(\underline{x}) = \underline{p}^T(\underline{x}) \cdot \mathbf{P}^{-1} \quad (m = N)$ <br> $\mathbf{P}_{Ij} = p_j(\underline{x}_I) \quad I = 1,N$ et $j = 1,m$ |
| | *Références* | [LIU 99 ; LIU 01a] |
| *Polynomiale- RBF* | RPIM | $\phi_I(\underline{x}) = \sum_{J=1}^{N} B_J(\underline{x}) \left[\mathbf{G}^{-1}\right]_{JI} + \sum_{\substack{j=1 \\ k=j+N}}^{m} p_j(\underline{x}) \left[\mathbf{G}^{-1}\right]_{kI}$ |
| | *Références* | $\mathbf{G} = \begin{bmatrix} \mathbf{B} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix}$, $B_J(\underline{x})$ : CSRBF ou GSRBF <br> [WAN 00 ; WAN 02a ; WAN 02b] |

***Tabl. 1.4 : Formalismes discrets pour les méthodes polynomiales et RBF.***

***Rappel des principales notations des tableaux 3 et 4 :***

| | |
|---|---|
| $\underline{x}$ | Vecteur des coordonnées spatiales d'un point du domaine d'étude $\Omega$. |
| $u^h(\underline{x})$ | Approximation discrète du champ solution, calculée au point $\underline{x}$. |
| $N$ | Nombre total de nœuds (ou particules) sur $\Omega$. |
| $\underline{p}^T(\underline{x}) = \{p_j(\underline{x})\}_{j=1}^{m}$ | Base primaire de dimension *m* (ex : polynôme d'ordre $\tilde{m}$). |
| $\mathbf{M}(\underline{x})$ | Matrice des moments discrets (appellation initialement pour *RKM*). |
| $\underline{x}_I$ | Vecteur des coordonnées spatiales d'un nœud *I* quelconque ($I = 1,N$). |
| $\phi_I(\underline{x})$ | Fonctions de forme meshfree associée au nœud *I*. |
| $C_I(\underline{x})$ | Fonction de correction associée au nœud *I* (appellation *RKM*), calculée de façon à atteindre un ordre de consistance donné. |
| $W_I(\underline{x})$ | Fonctions de pondération (*MLS*) ou fenêtre (*RKM*) ou fonction de lissage (SPH) associée au nœud *I*. |
| $d_I$ | Contribution nodale associée au nœud *I*. |
| $N_I(\underline{x})$ | Fonction de forme interpolante (FEM, etc.) associée au nœud *I*. |
| $B_I(\underline{x})$ | Fonction de base radiale à support global (GSRBF) ou compact (CSRBF), associée au nœud *I*. |

| **Approximations** | **Principales Méthodes meshfree** | **Formalisme discret :** $u^h(\underline{x}) = \sum_{I=1}^{N} \phi_I(\underline{x}) V_I(\underline{x})$ |
|---|---|---|
| *MLS* | h-p Clouds | $V_I(\underline{x}) = d_I + \sum_{k=1}^{m_I} L_{Ik}(\underline{x}) d_{Ik}$<br>$\phi_I(\underline{x})$ : fonction de forme *MLS* |
| | *Références* | [DUA 95 ; DUA 96a ; DUA 96b] |
| *PU* | C-PUM | $\phi_I(\underline{x}) = \frac{W_I(\underline{x})}{\sum_{J=1}^{N} W_J(\underline{x})}$<br>$V_I(\underline{x}) = d_I + \sum_{J=1}^{N_{vI}} C_J^{(I)}(\underline{x})(d_J - d_I),\ I = 1, N$ |
| | *Références* | [BAB 95 ; BAB 97 ; KRY 00] |

***Tabl. 1.5 : Formalismes discrets pour les méthodes h-p Clouds et C-PUM.***

***Rappel des notations spécifiques au tableau 5 :***

| | |
|---|---|
| $V_I(\underline{x})$ | Fonction nodale définie sur un voisinage du nœud *I*. |
| $\underline{L}_I^T(\underline{x}) = \{L_{Ii}(\underline{x})\}_{i=1}^{m_I}$ | Base secondaire de dimension $m_I$ (ex : polynôme ordre $\tilde{m}_I$ ), associée au nœud *I* et calculée au point $\underline{x}$. |
| $\{d_{Ik}\}_{k=1}^{m_I}$ | Contributions nodales complémentaires associées au nœud *I*. |
| $N_{vI}$ | Nombre de nœuds voisins (« *star nodes* ») du nœud *I*. |
| $C_J^{(I)}(\underline{x})$ | Fonction de correction associée aux nœuds *I* et *J* et permettant d'atteindre l'ordre de consistance souhaité pour l'approximation. |

# 2. Mise en œuvre des Méthodes Meshfree

## 2.1 INTRODUCTION

Nous nous sommes attachés au niveau du premier chapitre de ce mémoire, à faire la synthèse d'un grand nombre de méthodes meshfree proposées actuellement dans la littérature. Cette synthèse en fournissant une écriture commune, visait en fait à faciliter la mise en œuvre numérique de ces différentes méthodes. Les difficultés pratiques se posant lors de cette mise en œuvre, quelles soient d'ordre algorithmique, ou bien au niveau du choix des techniques les mieux appropriées pour simuler un problème donné, sont à notre avis souvent insuffisamment traitées dans la littérature. Notre motivation en abordant dans ce chapitre un certain nombre d'aspects pratiques, est de fournir des éléments en vue d'une utilisation plus courante des méthodes meshfree. Au niveau de la démarche, nous avons tout d'abord choisi de présenter différents aspects liés au développement, puis de tester les méthodes présentées au premier chapitre sur des applications numériques simples permettant ainsi d'évaluer leurs performances numériques et de fournir des éléments de comparaison.

Du point de vue du développement, l'une des principales difficultés réside dans la détermination et éventuellement la gestion des domaines d'influence des nœuds de la discrétisation. On verra en effet dans les exemples numériques de ce chapitre, que la taille et la forme du domaine d'influence sont des paramètres essentiels au niveau de la convergence et de la précision numérique qu'il est possible d'atteindre. Par ailleurs, il est généralement indispensable d'utiliser une technique de recherche rapide des nœuds appartenant aux différents domaines d'influence. C'est notamment le cas lorsque la taille du problème à simuler est importante (nombre de nœuds élevé) ou lorsque les domaines d'influence peuvent évoluer au cours de la simulation et doivent donc être régulièrement réévalués (exemple : raffinement adaptatif ou nœuds mobiles pour simuler des champs évolutifs avec forts gradients, grandes transformations en formulation lagrangienne réactualisée, etc.).

Du fait que l'on utilise généralement un formalisme discret pour résoudre le problème aux limites mécanique, un autre choix essentiel concerne la méthode à adopter pour intégrer numériquement sur le domaine d'étude, les équations de conservation classiques (mouvement, moments, masse, énergie). Pour ce faire, deux méthodes sont généralement utilisées en pratique :

- la collocation : les équations sont résolues au sens fort en intégrant sur un ensemble de points qui coïncident en général avec les nœuds utilisés pour l'approximation ;

- les méthodes de type Galerkin : les équations sont résolues au sens faible sur un ensemble de points d'intégration ne coïncidant pas avec les nœuds.

En pratique, la collocation est souvent utilisée avec les approximations *RKM* (essentiellement la méthode SPH) et *RBF*, tandis que les méthodes de type Galerkin sont souvent préférées pour les approximations *MLS* et polynomiales.

Enfin, une autre difficulté importante rencontrée lors de la mise en œuvre pratique des méthodes meshfree, réside dans le traitement des discontinuités, celles-ci pouvant être matérielles (contrastes de propriétés physiques et/ou mécaniques) ou numériques (ordres ou méthodes d'approximation différents). Elles peuvent également être liées à une discontinuité du milieu physique à modéliser (fissures, etc.). Dans le cadre de ce mémoire, ce dernier cas ne sera pas envisagé et nous nous intéresserons uniquement au cas des domaines convexes.

Dans la première partie de ce chapitre, nous allons tout d'abord rappeler les diverses techniques numériques communes aux méthodes meshfree et relatives aux difficultés énoncées plus avant, à savoir :

- la définition du domaine d'influence d'un nœud donné : détermination du rayon « optimal » et recherche rapide des nœuds appartenant au domaine ;

- l'intégration spatiale dans le cadre d'une formulation de type Galerkin : intégration nodale ou par une quadrature de type Gauss-Legendre avec utilisation de grilles ;

- le traitement des discontinuités matérielles ou numériques.

Nous présenterons également les techniques numériques plus spécifiques concernant la construction des fonctions de forme dans le cas d'un arrangement nodal singulier. Ces techniques ont initialement été proposées par Liu et co-auteurs pour la méthode PIM (voir notamment [LIU 03]).

Dans la seconde partie du chapitre, nous présenterons les résultats issus des simulations réalisées pour des problèmes mécaniques simples, avec les méthodes meshfree présentées dans le premier chapitre de ce mémoire.

Nous avons fait le choix ici de rendre interpolantes toutes les méthodes *MLS* et *RKM* testées, pour en évaluer les performances. Ce choix est par ailleurs motivé par les applications visées, à savoir des simulations en transformations finies avec des lois de comportement complexes, simulations fort consommatrices en temps de calcul. Suivant les applications, l'interpolation sera réalisée soit sur la totalité du domaine d'étude en utilisant pour ce faire, des fonctions de pondération ou des fonctions de forme modifiées, soit uniquement au niveau des frontières avec conditions aux limites imposées (Dirichlet ou Neumann), en les couplant avec la méthode des éléments finis. Dans le cas où la solution analytique n'est pas connue pour le test considéré, la méthode des éléments finis est utilisée comme référence.

Nous avons par ailleurs choisi de ne pas utiliser les méthodes SPH et DEM, la première constituant en fait un cas particulier de la méthode RKPM, et la seconde, un cas particulier de la méthode EFGM avec dérivées tronquées des fonctions de forme (*cf.* chapitre 1). De même, nous avons abandonné les méthodes C-PUM et RBF (raisons évoquées au chapitre 1).

## 2.2 TECHNIQUES NUMERIQUES POUR LES METHODES MESHFREE

### 2.2.1 Domaine d'influence et recherche des nœuds voisins

Deux stratégies existent habituellement pour déterminer les domaines d'influence pour tous les nœuds d'une discrétisation.

La première technique consiste à fixer de manière uniforme les dimensions du domaine d'influence des nœuds de la discrétisation. Cette méthode est applicable pour des arrangements nodaux réguliers, dont le pas de discrétisation dans chaque direction spatiale est fixe et connu a priori.

La seconde technique consiste tout d'abord à déterminer les *m* nœuds les plus proches d'un point d'évaluation donné (par exemple un nœud ou un point d'intégration), de manière à obtenir une matrice des moments non singulière (*cf.* chapitre 1). Il s'agit ensuite d'augmenter éventuellement la taille des domaines d'influence de ces nœuds voisins, de façon à ce qu'ils contiennent le point d'évaluation considéré. Cette technique est mieux adaptée lorsque la répartition des nœuds de la discrétisation est très irrégulière. Elle nécessite néanmoins un algorithme spécifique pour optimiser la recherche des nœuds voisins.

Une approche couramment utilisée en pratique consiste à subdiviser le domaine d'étude en sous-domaines rectangulaires, appelés « *buckets* », pour lesquels est définie un nombre maximal de nœuds pouvant être contenu à l'intérieur. Cette limite est déterminée a priori et est fonction de la taille du problème et du nombre maximal de nœuds admissibles pour un domaine d'influence donné. On utilise ensuite une procédure récursive pour déterminer l'ensemble des « *buckets* » nécessaires pour couvrir le domaine d'étude et vérifiant la condition sur le nombre limite de nœuds (*cf.* Tabl. 2.2). Finalement, au lieu de s'effectuer sur le domaine global, la recherche des nœuds voisins s'effectue alors simplement sur les « *buckets* » recoupant le domaine d'influence en cours de construction.

(1) *Déterminer le nombre maximal* $N_l$ *de noeuds autorisés par "bucket".*

(2) *Créer une liste* $\mathcal{L}_{\mathcal{B}}$ *pour gérer les "buckets".*

(3) *Initialiser : le premier "bucket" contient tout le domaine.*

(4) *Pour tout "bucket"* $\mathcal{B}$ *:*

- *a. Evaluer le nombre N de noeuds effectivement inclus*
- *b. Si* $N > N_l$ *alors :*
  - *Diviser en deux sous - buckets* $\mathcal{SB}_1$ *et* $\mathcal{SB}_2$ *de taille égale;*
  - *Pour* $i = 1, 2,$ $\mathcal{B} = \mathcal{SB}_i$ *et aller en (4).*
- *c. Si* $N \leq N_l$ *alors : ajouter* $\mathcal{B}$ *à* $\mathcal{L}_{\mathcal{B}}$ *et aller en* (5).

(5) *Fin*

***Tabl. 2.1 : Algorithme récursif de construction des « buckets » (« bucket algorithm »).***

## 2.2.2 Intégration spatiale

### *2.2.2.1 Quadrature de Gauss - Legendre*

La technique d'intégration de Gauss-Legendre est très utilisée en pratique pour résoudre les problèmes aux limites en éléments finis (voir par exemple [ZIE 91]). Cette technique est également couramment utilisée avec les méthodes meshfree et nécessite de construire une grille permettant de définir les points de Gauss. Cette grille peut être soit un maillage conforme à la géométrie du modèle comme pour la méthode des éléments finis classique, soit un ensemble de cellules indépendantes formant un domaine couvrant le modèle (voir par exemple [BEL 94 ; BEL 96 ; DOL 98]). Dans ce dernier cas, on utilise généralement une grille régulière. Belytschko et co-auteurs [BEL 94] proposent même une règle empirique pour définir le nombre de points de Gauss $n_Q$ à utiliser par cellule et le nombre de cellules $n_c$ à générer automatiquement pour couvrir le domaine d'étude. Ainsi, pour une application 2D, ils préconisent d'utiliser les valeurs suivantes :

$$\begin{cases} n_c = \sqrt{N} \\ n_Q = \alpha + \sqrt{n_c} \qquad \alpha = 2 \text{ ou } 3 \end{cases} \tag{2.1}$$

où $N$ représente le nombre total de nœuds de la discrétisation.

Nous verrons dans les applications numériques de la section §2.3, l'influence du nombre de points d'intégration, ainsi que de la forme des supports d'intégration sur la qualité des résultats obtenus.

### *2.2.2.2 Intégration nodale*

Certains auteurs [BEI 96 ; NAG 99 ; CHE 01,02b] préfèrent utiliser la technique d'intégration nodale, car elle permet d'obtenir une méthode réellement meshfree. En revanche, comme on le verra ci-après, cette technique ne fait pas obligatoirement gagner du temps au niveau des calculs par rapport à une quadrature de Gauss classique, du fait que certaines procédures de correction sont ensuite nécessaires pour maintenir la précision numérique.

D'une manière générale, pour intégrer numériquement la fonction $f(\underline{x})$ quelconque sur les $N$ nœuds du domaine d'étude $\Omega$, on écrira simplement :

$$\int_{\Omega} f(\underline{x})\, d\Omega = \sum_{I=1}^{N} \int_{\Omega_I} f(\underline{x})\, d\Omega = \sum_{I=1}^{N} f(\underline{x}_I)\, \Delta V_I \tag{2.2}$$

où $\Omega_I$ représente le domaine d'intégration du nœud $I$ quelconque et $\Delta V_I = \int_{\Omega_I} d\Omega$, le volume ou poids d'intégration associé. Tout le problème consiste alors à déterminer $\Delta V_I$.

***Intégration mixte***

La méthode utilisée par Nagashima [NAG 99] dans le cadre de calculs de structures peut être qualifiée de méthode mixte, car elle repose sur la définition d'un ensemble de « *buckets* », dont les centres seront utilisés pour l'intégration spatiale. Comme on l'a mentionné précédemment, ces « *buckets* » permettent par ailleurs une recherche rapide des nœuds voisins d'un point donné, car seules celles directement contiguës ou voisines du point et donc seuls les nœuds situés dans ces cellules sont examinés (structure arborescente).

Tout d'abord, on définit l'ensemble $\{C_k\}_{k=1}^{N_c}$ des $N_c$ cellules rectangulaires couvrant le domaine d'étude. Soit $N_{vk}$, le nombre de nœuds appartenant au domaine d'influence $\Omega_k$ du centre $\underline{x}_k$ de la cellule $C_k$.

Par ailleurs, on définit la fonction poids associée à $\underline{x}_k$ sur $\Omega_k$ par la relation :

$$w_k(\underline{x}) = \begin{cases} \|\underline{x} - \underline{x}_k\| & \underline{x} \in \Omega_k \\ 0 & \underline{x} \notin \Omega_k \end{cases} \tag{2.3}$$

Enfin, si $A_k$ représente l'aire de la cellule $C_k$, alors l'aire $a_k(\underline{x}_I)$, correspondant à l'aire distribuée de $C_k$ sur tout nœud $I$ (position $\underline{x}_I$) appartenant au domaine d'influence $\Omega_k$ est donnée par :

$$a_k(\underline{x}_I) = \frac{w_k(\underline{x}_I) A_k}{\sum_{J=1}^{N_v} w_k(\underline{x}_J)} \qquad \text{pour } I = 1, N_v \text{ et } k = 1, N_c \tag{2.4}$$

Le volume d'intégration associé au nœud $I$ quelconque est finalement donné par :

$$\Delta V_I = \sum_{k=1}^{N_c} a_k(\underline{x}_I) \tag{2.5}$$

***Intégration purement nodale***

Dans le cadre de la méthode EFGM, Beissel et Belytschko [BEI 96] proposent d'intégrer la formulation variationnelle du problème uniquement aux nœuds du problème, ce qui en fait revient à utiliser une méthode de collocation. Pour évaluer les intégrales de volume, ils proposent par exemple de calculer le poids nodal correspondant par la relation suivante :

$$\Delta V_I = \frac{f_{\Omega,I}\, {h_I}^2}{\sum_{J=1}^{N} f_{\Omega,J}\, {h_J}^2} A_\Omega \tag{2.6}$$

où $A_\Omega$ représente le volume total de $\Omega$ ; $h_I$ correspond au rayon du domaine d'influence associé au nœud $I$ et $f_{\Omega,I}$, la fraction du support radial (nœud $I$) incluse dans $\Omega$. Par

exemple, on aura : $f_{\Omega,I} = 1$ pour des nœuds intérieurs ; $f_{\Omega,I} = 0.5$ pour les nœuds situés sur une frontières rectiligne et $f_{\Omega,I} = 0.25$ pour des nœuds situés sur un coin à angle droit.

Dans le cas des intégrales sur la frontière $\Gamma$ du domaine d'étude, ils proposent de calculer le poids nodal par l'expression suivante :

$$\Delta V_I = \frac{f_{\Gamma,I}\, h_I}{\sum_{J=1}^{N_\Gamma} f_{\Gamma,J}\, h_J} L_\Gamma \tag{2.7}$$

où $L_\Gamma$ représente la longueur totale de la frontière considérée et $f_{\Gamma,I}$, la fraction de support couverte par la frontière.

### *2.2.2.3 Techniques de stabilisation utilisées pour l'intégration nodale*

Le principal inconvénient rencontré dans l'utilisation des méthodes d'intégration nodale (ou par collocation) est que ces dernières fournissent généralement des erreurs importantes, notamment au niveau des contraintes et déformations calculées, et qu'elles sont de ce fait, à l'origine d'instabilités purement numériques (voir par exemple [BEI 96 ; BEL 00]). Ceci s'explique par le fait que le nombre de nœuds ou points utilisés pour l'intégration est généralement insuffisant (sous-intégration), comparé à celui utilisé pour une intégration de type Gauss.

Nous allons présenter ci-après quelques techniques de stabilisation permettant de remédier à ces inconvénients.

***Utilisation de particules esclaves (« stress points »)***

Une première technique consiste à utiliser des particules additionnelles, appelées « *stress points* », permettant essentiellement de mieux intégrer les équations de comportement et de ce fait, de mieux calculer les déformations et les contraintes sur le domaine d'étude [RAN 96 ; DYK 97 ; BEL 00]. Suivant la terminologie utilisée en éléments finis, ces points sont également appelés « nœuds esclaves », par opposition aux « nœuds maîtres » de la discrétisation initiale sur lesquels sont calculés les champs inconnus (déplacements, vitesses, etc.). Les valeurs des champs inconnus aux nœuds esclaves sont alors déduites de celles calculées aux nœuds maîtres voisins, ce qui ne rajoute pas de degrés de liberté supplémentaires. Les efforts intérieurs sont par ailleurs calculés sur l'ensemble des nœuds maîtres et esclaves.

Par exemple, le vecteur déplacement $\underline{u}_I^E$ calculé au nœud esclave *I* quelconque s'écrira :

$$\underline{u}_I^E = \sum_{J=1}^{N^M} \phi_J\left(\underline{x}_I^E\right) \underline{u}_J^M \tag{2.8}$$

où $N^M$ représente le nombre de nœuds maîtres voisins du point $\underline{x}_I^E$ considéré (nœud esclave), $\underline{u}_J^M$ est le vecteur déplacement calculé au nœud maître *J* par résolution des

équations du problème aux limites et $\phi_J\left(\underline{x}_I^E\right)$, la fonction de forme correspondante calculée au point $\underline{x}_I^E$ .

Belytschko et al. [BEL 00] proposent de construire une structure de cellules de Voronoï pour définir les nœuds esclaves par rapport aux nœuds maîtres de la discrétisation (*cf*. Fig. 2.1), cette structure étant ensuite utilisée pour déterminer les supports d'intégration des nœuds esclaves et pour intégrer la formulation variationnelle.

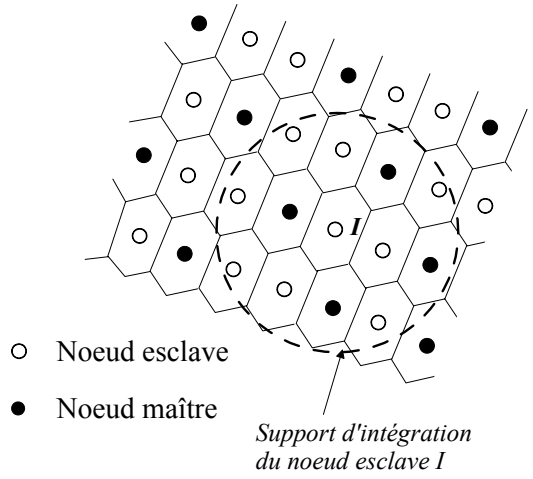



***Fig. 2.1 : Définition des nœuds maîtres et esclaves dans une structure de Voronoï.***

Cette technique est bien adaptée pour un problème avec arrangement nodal régulier. Par ailleurs, Belytschko *et al.* [BEL 00] ont montré qu'un arrangement hexagonal permettait d'obtenir de meilleurs résultats en termes de stabilité et de précision (*cf*. Fig. 2.2) .

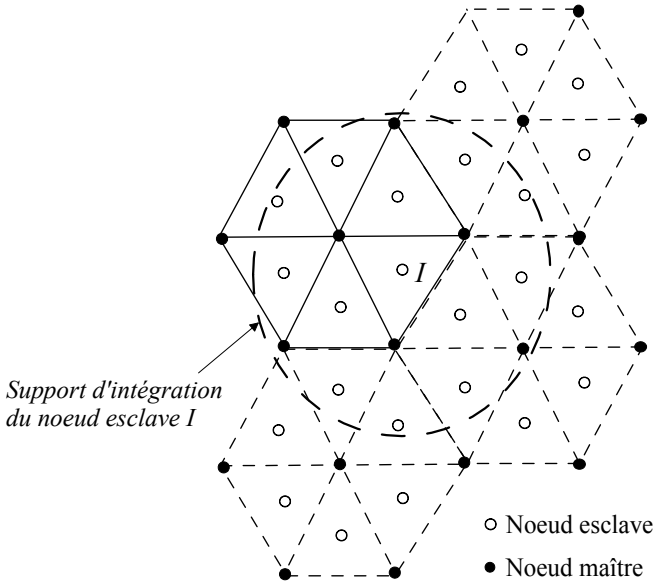



***Fig. 2.2 : Arrangement nodal hexagonal.***

***Utilisation de termes de stabilisation au niveau des équations du problème***

D'autres techniques possibles consistent à ajouter des termes de stabilisation directement au niveau des équations du problème à résoudre.

Concernant l'intégration purement nodale appliquée à des problèmes de mécanique des solides (par exemple, sans couplage avec une phase fluide), Beissel et Belytschko [BEI 96] proposent de rajouter le carré du résidu de l'équation de conservation de la quantité de mouvement au niveau de la formulation faible du problème. Par exemple, dans le cas d'un problème statique en transformations infinitésimales, on écrira :

$$\int_{\Omega}\left[\underline{\underline{\sigma}}(\underline{u}):\underline{\underline{\varepsilon}}(\underline{v}_u)+\underline{b}\cdot\underline{v}_u\right]d\Omega+\frac{\alpha_s l_c^2}{E}\int_{\Omega}\underline{div\underline{\underline{\sigma}}(\underline{v}_u)}\cdot\left(\underline{div\underline{\underline{\sigma}}(\underline{u})}+\underline{b}\right)d\Omega=0 \tag{2.9}$$

où $\underline{b}$ représente les forces de volume et $\underline{\underline{\sigma}}$, $\underline{\underline{\varepsilon}}$, les tenseurs de contraintes de Cauchy et déformations infinitésimales ; $\underline{u}$ et $\underline{v}_u$ représentent respectivement le champ de déplacement inconnu et le champ virtuel cinématiquement admissible associé.

Par ailleurs, $l_c$ est une longueur caractéristique dépendant de la discrétisation (ou de l'arrangement nodal), $\alpha_s$, un paramètre adimensionnel de stabilisation et $E$, le module d'Young.

Néanmoins, les auteurs ont montré que cette technique d'intégration nodale avec stabilisation demeure bien inférieure à une intégration avec points de Gauss, tant au niveau de la convergence, que de la précision numérique.

***Utilisation d'un opérateur de dérivation modifié***

Nagashima [NAG 99] propose quant à lui, d'utiliser un développement de Taylor à l'ordre 1 permettant d'améliorer le calcul de la rigidité nodale dans le cas de l'intégration mixte.

Par exemple en 2D, si l'on note $\boldsymbol{K}_I$, la rigidité élastique calculée au nœud $I$ quelconque, on aura :

$$\boldsymbol{K}_I=\int_{\Omega_I}\left[\boldsymbol{B}^I(\underline{x})\right]^T\cdot\underline{\underline{\underline{\underline{D}}}}\cdot\boldsymbol{B}^I(\underline{x})\,d\Omega \tag{2.10}$$

en notant $\underline{\underline{\underline{\underline{D}}}}$, le tenseur d'élasticité.

En 2D, la sous-matrice $\boldsymbol{B}^I$ associée au nœud $I$, s'écrit par exemple sous la forme suivante :

$$\boldsymbol{B}^I(\underline{x})=\begin{bmatrix}\partial_x\phi_I & 0\\ 0 & \partial_y\phi_I\\ \partial_y\phi_I & \partial_x\phi_I\end{bmatrix} \tag{2.11}$$

L'expression de $\boldsymbol{B}^I(\underline{x})$ est alors remplacée dans (2.10) par son développement de Taylor à l'ordre 1 autour de $\underline{x}_I$ (position du nœud *I*) :

$$\boldsymbol{B}^I(\underline{x}) \approx \boldsymbol{B}_I^I + \sum_{i=1}^{n}\left[(\underline{x}-\underline{x}_I)\cdot\underline{e}_i\right]\partial_i\boldsymbol{B}_I^I \tag{2.12}$$

avec *n*, la dimension de l'espace euclidien et en notant pour simplifier : $\boldsymbol{B}_I^I = \boldsymbol{B}^I(\underline{x}_I)$ et $\partial_i\boldsymbol{B}_I^I = \partial_i\boldsymbol{B}^I(\underline{x})\Big|_{\underline{x}=\underline{x}_I}$.

Ainsi, la rigidité calculée pour un nœud *I* quelconque s'écrira :

$$\begin{aligned}
\boldsymbol{K}_I = & \left[\boldsymbol{B}_I^I\right]^T\cdot\underline{\underline{\underline{D}}}\cdot\boldsymbol{B}_I^I V_I \\
& + \frac{h_I^3}{3}(\sin\alpha-\sin\beta)\left(\left[\partial_x\boldsymbol{B}_I^I\right]^T\cdot\underline{\underline{\underline{D}}}\cdot\boldsymbol{B}_I^I + \left[\boldsymbol{B}_I^I\right]^T\cdot\underline{\underline{\underline{D}}}\cdot\partial_x\boldsymbol{B}_I^I\right) \\
& + \frac{h_I^3}{3}(\cos\alpha-\cos\beta)\left(\left[\partial_y\boldsymbol{B}_I^I\right]^T\cdot\underline{\underline{\underline{D}}}\cdot\boldsymbol{B}_I^I + \left[\boldsymbol{B}_I^I\right]^T\cdot\underline{\underline{\underline{D}}}\cdot\partial_y\boldsymbol{B}_I^I\right) \\
& + \frac{h_I^4}{8}\left(\sin^2\alpha-\sin^2\beta\right)\sum_{\substack{i,j\\ i\neq j}}\partial_i\left[\boldsymbol{B}_I^I\right]^T\cdot\underline{\underline{\underline{D}}}\cdot\partial_j\boldsymbol{B}_I^I \\
& + \frac{h_I^4}{8}\left[\alpha-\beta+\frac{1}{2}(\sin 2\alpha-\sin 2\beta)\right]\sum_i\partial_i\left[\boldsymbol{B}_I^I\right]^T\cdot\underline{\underline{\underline{D}}}\cdot\partial_i\boldsymbol{B}_I^I
\end{aligned} \tag{2.13}$$

$\alpha$ et $\beta$ sont les angles utilisés pour définir la portion de support (radial) dans le cas d'un nœud frontière (*cf.* Fig. 2.3) et $h_I = \sqrt{\dfrac{2V_I}{\alpha-\beta}}$ est le rayon du support.

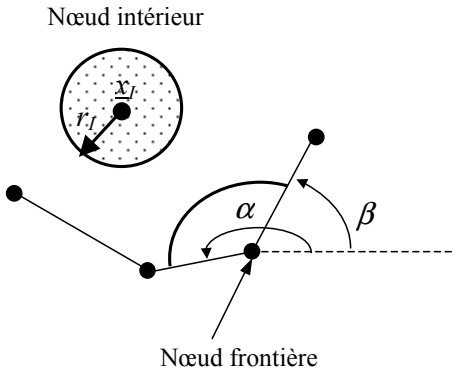



***Fig. 2.3 : Définition des domaines d'intégration nodaux pour les termes de stabilisation.***

Pour un nœud intérieur ($\alpha = 2\pi$ , $\beta = 0$), l'expression (2.13) s'écrira plus simplement :

$$\boldsymbol{K}_I = \left( \left[ \boldsymbol{B}_I^I \right]^T \cdot \underline{\underline{\underline{\underline{D}}}} \cdot \boldsymbol{B}_I^I + \frac{V_I}{4\pi} \sum_i \partial_i \left[ \boldsymbol{B}_I^I \right]^T \cdot \underline{\underline{\underline{\underline{D}}}} \cdot \partial_i \boldsymbol{B}_I^I \right) V_I \tag{2.14}$$

Enfin, Chen et co-auteurs [CHE 01 ; CHE 02b] proposent d'utiliser une technique de lissage des déformations, nécessitant également de modifier l'opérateur de dérivation. Pour ce faire, ils cherchent à déterminer les conditions d'intégration nécessaires pour atteindre l'ordre de consistance souhaité, en l'occurrence linéaire.

De même que pour la technique des nœuds esclaves vue précédemment, les auteurs utilisent une structure de cellules de Voronoï (ou de manière équivalente, une triangulation de Delaunay) pour définir un domaine représentatif $\tilde{\Omega}_I$ associé à chaque nœud $I$ de la discrétisation, ce domaine étant distinct du domaine d'influence (*cf.* Fig. 2.4). Ils déterminent ensuite la déformation en un nœud $I$ quelconque, par lissage des déformations sur le domaine représentatif associé :

$$\underline{\underline{\underline{\underline{\tilde{\varepsilon}}}}}^h \left( \underline{x}_I \right) = \sum_{J \in G_I} \tilde{\boldsymbol{B}}^J \left( \underline{x}_I \right) \cdot \underline{d}_J \tag{2.15}$$

où $\underline{x}_I$ représente le vecteur position du nœud $I$ et $G_I$, le groupe de nœuds dont le domaine d'influence contient $I$. $\tilde{\boldsymbol{B}}^J \left( \underline{x}_I \right)$ est l'opérateur de dérivation lissé associé au nœud $I$, défini par l'expression suivante :

$$\tilde{\boldsymbol{B}}^J \left( \underline{x}_I \right) = \frac{1}{\tilde{A}_I} \begin{bmatrix} \int_{\tilde{\Gamma}_I} \phi_I \left( \underline{x} \right) n_x \left( \underline{x} \right) d\Gamma & 0 \\ 0 & \int_{\tilde{\Gamma}_I} \phi_I \left( \underline{x} \right) n_y \left( \underline{x} \right) d\Gamma \\ \int_{\tilde{\Gamma}_I} \phi_I \left( \underline{x} \right) n_y \left( \underline{x} \right) d\Gamma & \int_{\tilde{\Gamma}_I} \phi_I \left( \underline{x} \right) n_x \left( \underline{x} \right) d\Gamma \end{bmatrix} \tag{2.16}$$

$\tilde{\Gamma}_I$ et $\tilde{A}_I$ représentent respectivement la frontière et le volume (ou l'aire) du domaine représentatif associé au nœud $I$ (*cf.* Fig. 2.4), et $\underline{n}\left( \underline{x} \right)$, la normale extérieure à $\tilde{\Gamma}_I$.

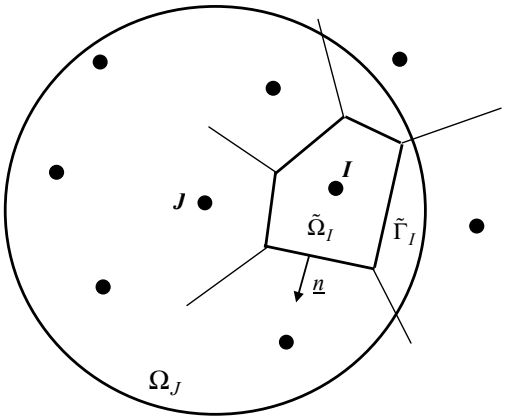



***Fig. 2.4: Relation géométrique 2D entre le domaine représentatif du nœud I et le domaine d'influence du nœud J.***

Les auteurs montrent que pour des domaines représentatifs de type cellules de Voronoï, la condition de consistance linéaire se traduit par la relation :

$$\sum_{I=1}^{N_{ip}} \tilde{\boldsymbol{B}}^J \left(\underline{x}_I\right) \tilde{A}_I = \mathbf{0} \tag{2.17}$$

Soit encore :

$$\sum_{I=1}^{N} \int_{\tilde{\Gamma}_I} \phi_I \left(\underline{x}\right) n_j \left(\underline{x}\right) d\Gamma = 0 \qquad 1 \le j \le n \tag{2.18}$$

Les auteurs montrent alors que la condition (2.18) est automatiquement vérifiée pour les nœuds intérieurs, quelles que soient le type de conditions aux limites utilisées.

Dans le cas des nœuds dont le support de fonction de forme recoupe la frontière, la condition (2.18) s'écrit également :

$$\int_{\Gamma} \phi_I \left(\underline{x}\right) n_j \left(\underline{x}\right) d\Gamma = 0 \quad 1 \le j \le n \tag{2.19}$$

où cette fois, l'intégration est réalisée sur la frontière $\Gamma$ du domaine d'étude où $\underline{n}$ représente la normale extérieure à cette frontière.

A notre avis, cette méthode de stabilisation des déformations s'apparente à la technique d'intégration utilisant des nœuds esclaves [BEL 00]. En effet, dans les deux méthodes, l'objectif est en fait de réduire les erreurs numériques dues à une mauvaise évaluation des efforts intérieurs aux nœuds de la discrétisation.

D'une façon générale, les techniques de stabilisation semblent peu avantageuses pour une mise en œuvre en pratique, du fait qu'elles nécessitent un nombre d'opérations plus importants. Elles ont été utilisées par leurs auteurs exclusivement pour des problèmes mécaniques purs avec des lois de comportement simples, le plus souvent élastiques. Elles sont à notre avis peu envisageables pour des problèmes géomécaniques couplés, pour lesquels les

lois de comportement sont généralement complexes et déjà fortement consommatrices de temps de calcul.

Néanmoins, la technique d'intégration avec particules esclaves nous semblent la plus intéressante d'un point de vue pratique, à condition d'utiliser un arrangement nodal régulier de type hexagonal.

### 2.2.3 Traitement des discontinuités

La prise en compte de discontinuités matérielles (contrastes de propriétés physiques et/ou mécaniques) ou numériques (ordres ou méthodes d'approximation différents), nécessite généralement un traitement spécifique avec les méthodes meshfree, du fait de la propriété de non interpolation de la plupart de ces méthodes.

Au niveau pratique, la simulation d'un problème physique avec une méthode meshfree requiert éventuellement de « découper » le domaine d'étude en sous-domaines représentatifs des différentes propriétés matérielles et/ou numériques du problème. Les nœuds héritent ensuite des propriétés du sous-domaine auquel ils appartiennent. Dans le cas d'une approximation non interpolante, toute technique usuelle pour traiter les conditions aux limites (*cf*. chapitre 1) est ensuite envisageable au niveau de l'interface entre les sous-domaines. Il est par exemple possible d'introduire des multiplicateurs de Lagrange pour assurer la continuité du champ solution au sens faible.

La difficulté réside dans le fait que les nœuds situés sur l'interface doivent pouvoir hériter et contribuer simultanément aux deux sous-domaines adjacents. Une solution consiste à dupliquer les contributions nodales et éventuellement les multiplicateurs de Lagrange (cas non interpolant) sur l'interface [AUB 97]. Pour notre part, nous avons préféré créer automatiquement des nœuds « fictifs » additionnels, coïncidant avec chacun des nœuds « principaux » de l'interface. Les nœuds principaux sont alors affectés à l'un des sous-domaines, et les nœuds fictifs, à l'autre. Cette technique présente à notre avis, l'avantage de pouvoir traiter ultérieurement un saut du champ principal, les nœuds ainsi créés ayant alors leurs propres degrés de liberté et étant libres de se déplacer indépendamment.

### 2.2.4 Techniques spécifiques dans le cas d'un arrangement nodal singulier

Dans le premier chapitre de ce mémoire, on a vu que la détermination des fonctions de forme des méthodes *MLS* et PIM nécessite de pouvoir inverser la matrice des moments ($\mathbf{M}(\underline{x})$ pour les méthodes *MLS* et $\mathbf{P}$ pour la méthode PIM). Cette matrice devient en fait singulière, lorsque son rang est différent du nombre de termes $m$ de la base primaire. Ceci résulte généralement d'un arrangement spatial singulier ou encore d'un nombre inapproprié de nœuds voisins utilisés pour calculer les fonctions de forme au point d'évaluation (nœud ou point d'intégration).

Dans le cas de la méthode PIM, Liu et co-auteurs [LIU 01a ; LIU 03] proposent de remédier à ce problème en utilisant les quatre techniques présentées ci-après.

### *Utilisation de fonctions de base radiales*

La première solution préconisée par les auteurs consiste en fait à utiliser la méthode RPIM, plutôt que la méthode PIM initiale. En effet, comme on a pu le voir au chapitre 1, les fonctions de base radiales additionnelles garantissent l'inversibilité de la matrice des moments résultante.

### *Déplacement aléatoire des nœuds voisins*

Une autre technique très simple consiste à faire bouger chaque nœud $I$ voisin du point d'évaluation $\underline{\xi}$, dans une direction aléatoire (*cf*. Fig. 2.5) et d'une distance $d_{mI}$ calculée de la manière suivante :

$$d_{mI} = \varepsilon \cdot \underset{J=1,m \text{ et } J \neq I}{Min} \left( \left\| \underline{x}_I - \underline{x}_J \right\|_2 \right) \tag{2.20}$$

avec $\varepsilon$, un coefficient réel fixé petit (dans leurs exemples : $-0.2 \leq \varepsilon \leq 0.2$).

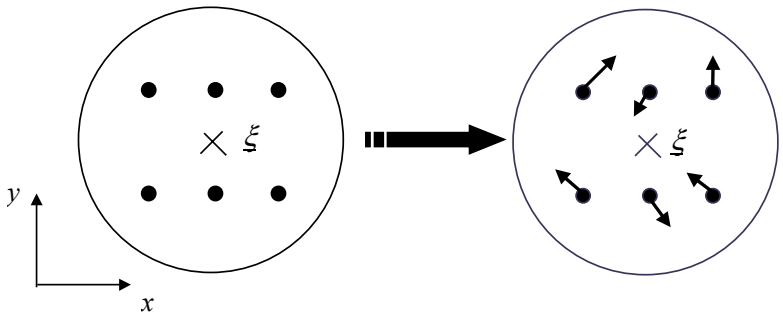



***Fig. 2.5 : Exemple 2D de modification aléatoire d'un arrangement nodal singulier***

### *Transformation du système de coordonnées initial*

Une autre technique envisageable consiste à calculer la matrice dans un nouveau système de coordonnées locales, obtenu par rotation du système initial et dont l'origine se situe par exemple au point d'évaluation (*cf*. Fig. 2.6).

La matrice **P** et donc les fonctions de forme sont alors calculées dans ce nouveau système de coordonnées. De manière classique, les dérivées des fonctions de forme sont calculées ensuite en utilisant la matrice jacobienne de la transformation (rotation).

On voit que le choix de l'angle de rotation est déterminant pour la réussite de cette technique. Liu [LIU 03] suggère d'utiliser un angle aléatoire en vérifiant que le déterminant de la matrice **P** correspondante ne s'annule pas, mais aucune méthode fiable n'existe à ce jour.

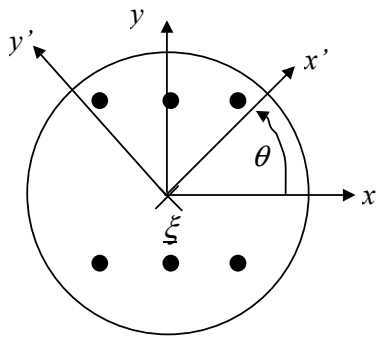



***Fig. 2.6 : Exemple 2D de rotation du système de coordonnées locales***

***Méthode de triangularisation de la matrice des moments***

La dernière technique proposée par les auteurs consiste à mettre en œuvre un algorithme systématique de triangularisation de **P**. Cette procédure qu'ils nomment MTA (« *Matrix Triangularization Technique* »), permet à la fois de choisir les nœuds à inclure ou non dans le support de la fonction de forme, et de déterminer pour un ordre $\tilde{m}$ fixé (base primaire polynomiale), les monômes de la base à conserver pour que le nombre de termes de la base soit en adéquation avec le nombre de nœuds effectivement inclus dans le support (*cf.* Tabl. 2.2).

L'incidence de ces différentes techniques sur la précision numérique des résultats est analysée dans la section §2.3.

*Pour chaque point d'évaluation* $\underline{\xi}$ *et* $\tilde{m}$ *étant fixé :*

(1) *Déterminer les* $N_\xi = m$ *noeuds inclus initialement dans le support de* $\underline{\xi}$ (*rayon* $h_\xi$)

(2) *Pour chaque noeud I du support :*

*Calculer les coordonnées locales normalisées :* $\tilde{\underline{x}}_I = \left(\underline{x}_I - \underline{x}_\xi\right) \Big/ h_{max}$

*avec :* $h_{max} = \max_I \left( \left\| \underline{x}_I - \underline{x}_\xi \right\| \right)$ *ou* $h_{max} = h_\xi$

(3) *Trianguler* $\mathbf{P}$ *à partir des* $\tilde{\underline{x}}_I$ *et enregistrer les échanges de lignes*

(4) *Déterminer le nombre r de lignes nulles* $(r \leq m)$

(5) *Si* $r < m$ *alors :*

*a. Déterminer les* $(m - r)$ *noeuds à exclure du support et correspondant aux lignes nulles* (*historique des échanges de lignes*)

*b. Trianguler* $\mathbf{P}^T$ *et enregistrer les échanges de lignes* ( = *colonnes de* $\mathbf{P}$)

*c. Déterminer les* $(m - r)$ *monomes de degré supérieur à exclure de la base (lignes nulles de* $\mathbf{P}^T$ *triangulée)* $\Rightarrow$ *base* $\underline{p'}\left(\underline{x}\right)$

*d. Eliminer les lignes et colonnes correspondantes de* $\mathbf{P}$

$\Rightarrow$ *matrice réduite* $\mathbf{P}'$ *calculée avec les* $\underline{x}_I$

*e. Aller en* (6).

*Sinon :* $\mathbf{P}' = \mathbf{P}$ *calculée avec les* $\underline{x}_I$ *et aller en* (6).

(6) *Calculer les fonctions de forme finales en utilisant* $\mathbf{P}'$ *:*

*vecteur aux noeuds :* $\underline{\phi}^{eT}\left(\underline{x}\right) = \underline{p}'^{T}\left(\underline{x}\right) \cdot \mathbf{P}'^{-1}$

***Tabl. 2.2 : Algorithme MTA***

## 2.3 APPLICATIONS NUMERIQUES

### 2.3.1 Introduction

Toutes les applications numériques de cette section ont été réalisées avec le logiciel *Movefree*. Cet outil numérique que nous avons développé dans le cadre de ce travail, utilise un langage orienté objet (C++) et intègre la majorité des méthodes meshfree développées au chapitre 1, ainsi que la méthode des éléments finis. Une présentation détaillée de l'outil (structures de données, architecture et possibilités) est fournie en annexe.

Par ailleurs, le choix des applications présentées ci-après visent essentiellement à atteindre les objectifs suivants :

- évaluer et valider les différentes méthodes meshfree implantées dans le logiciel ;
- évaluer et comparer les performances numériques entre les différentes méthodes à notre disposition, à savoir les méthodes meshfree et la méthode des éléments finis, et suivant le choix des techniques numériques utilisées (intégration, interpolation ou couplage avec des éléments finis, etc.).

### 2.3.2 Patch Tests linéaires

#### *2.3.2.1 Rappel*

D'une manière générale, le « patch test » consiste à étudier l'évolution des erreurs relatives calculées pour le champ solution et ses dérivées partielles en fonction de la discrétisation choisie, c'est-à-dire essentiellement en faisant varier le nombre de nœuds utilisés sur la frontière et à l'intérieur du domaine, ainsi que leurs positions (arrangement nodal régulier ou non).

Nous rappelons ci-après rapidement le principe de ce test mécanique standard [ZIE 91], réalisé en élasticité linéaire isotrope avec l'hypothèse des transformations infinitésimales. On se place par ailleurs dans le cadre des déformations planes.

Le problème consiste simplement à résoudre l'équation d'équilibre statique ( $\underline{\underline{div}}\,\underline{\underline{\sigma}} = \underline{0}$ ) sur un domaine carré homogène $\Omega$, en imposant des conditions aux limites linéaires en déplacements ( $\underline{u} = \underline{\overline{u}}$ ) sur la frontière de Dirichlet, notée $\Gamma_u$ :

$$\begin{cases} \overline{u}_x(\underline{x}) = x \\ \overline{u}_y(\underline{x}) = y \end{cases} \quad \forall \underline{x} \in \Gamma_u \tag{2.21}$$

La solution exacte $\underline{u}(\underline{x})$ pour tout point du domaine d'étude est alors la même fonction linéaire des coordonnées : $u_x(\underline{x}) = x$ et $u_y(\underline{x}) = y$.

On discrétise ensuite la formulation variationnelle de ce problème sur un ensemble de $N$ nœuds répartis sur $\overline{\Omega}$, en exprimant le champ de déplacement approché $\underline{u}^h(\underline{x})$ sous la forme :

$$\underline{u}^h(\underline{x}) = \sum_{I=1}^{N} \phi_I(\underline{x})\, d_{Ii}\, \underline{e}_i \tag{2.22}$$

$\phi_I(\underline{x})$ représente la fonction de forme associée au nœud $I$ pour l'approximation choisie (meshfree ou éléments finis), et $\{\underline{d}_I\}_{I=1}^{N}$, le vecteur des contributions nodales.

En posant $\boldsymbol{K}$, la matrice de rigidité globale et $\underline{U}$, le vecteur des contributions nodales pour l'ensemble des nœuds de $\overline{\Omega}$, le système final à résoudre (résolution de type Galerkin), peut s'écrire :

$$\boldsymbol{K} \cdot \underline{U} = \underline{0} \tag{2.23}$$

avec :

$$\boldsymbol{K} = \int_{\Omega} \boldsymbol{B}^T \cdot \underline{\underline{\underline{\underline{D}}}}^e \cdot \boldsymbol{B}\, d\Omega \tag{2.24}$$

$$\boldsymbol{B} = \left[ \boldsymbol{B}^1 \cdots \boldsymbol{B}^I \cdots \boldsymbol{B}^N \right] \tag{2.25}$$

et $\underline{\underline{\underline{\underline{D}}}}^e$, le tenseur élastique exprimé en déformations planes :

$$\underline{\underline{\underline{\underline{D}}}}^e = \begin{bmatrix} \lambda+2\mu & \lambda & 0 \\ \lambda & \lambda+2\mu & 0 \\ 0 & 0 & \mu \end{bmatrix} \tag{2.26}$$

où $\lambda$ et $\mu$ sont les coefficients de Lamé.

Les indicateurs d'erreurs relatives utilisés pour tous les tests sont les suivants :

$$e(\underline{u}) = \frac{\left\| \underline{u} - \underline{u}^h \right\|_{L_2(\Omega)}}{\left\| \underline{u} \right\|_{L_2(\Omega)}} \tag{2.27}$$

$$e_{u_i} = \frac{\sqrt{\int_{\Omega} \left( u_i - u_i^h \right)^2 d\Omega}}{\sqrt{\int_{\Omega} u_i^{\,2} d\Omega}} \tag{2.28}$$

$$e_{\partial_j \underline{u}} = \frac{\left\| \partial_j \underline{u} - \partial_j \underline{u}^h \right\|_{L_2(\Omega)}}{\left\| \partial_j \underline{u} \right\|_{L_2(\Omega)}} \tag{2.29}$$

avec pour la norme $L_2$ : $\left\| \underline{u} \right\|_{L_2(\Omega)} = \sqrt{\int_\Omega \underline{u}^T \cdot \underline{u} \, d\Omega}$ .

*2.3.2.2 Effet du choix du domaine d'influence*

Ce premier test avait pour objectif d'évaluer l'effet de la forme et la taille du domaine d'influence sur la précision des résultats (domaine radial ou rectangulaire, rayon fixe ou variable).

Nous avons considéré ici le domaine $\Omega = [0,1] \times [0,1]$ avec 4 nœuds répartis sur la frontière et un nœud intérieur (*cf.* Fig. 2.7). La base primaire était une base polynomiale linéaire. Nous avons également pris : $\lambda = 0.75\,Pa$ et $\mu = 0.5\,Pa$ .

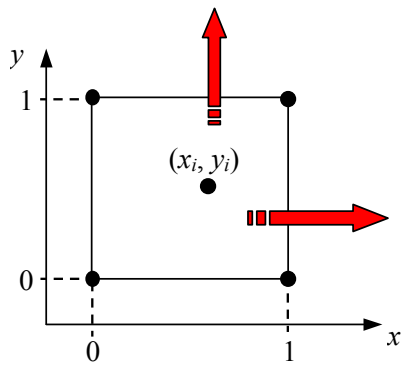



***Fig. 2.7: Schéma du patch test linéaire à 5 nœuds.***

Par ailleurs, nous avons uniquement considéré des fonctions de forme rendues interpolantes en utilisant les fonctions de pondérations modifiées (1.44). L'effet de l'utilisation des différentes techniques d'interpolation sera étudié dans le test à 36 nœuds (*cf.* la section §2.3.2.4).

De plus, l'intégration réalisée est de type Gauss avec une seule cellule (dimensions identiques au domaine) comprenant 2x2 points d'intégration.

Les tableaux 2.3 à 2.5 ci-après regroupent les erreurs obtenues pour différentes positions du nœud intérieur dans le quart supérieur droit de $\overline{\Omega}$, en utilisant les principales méthodes meshfree, et en considérant soit un domaine d'influence radial (*cf.* Tabl. 2.3) ou rectangulaire (*cf.* Tabl. 2.4). Pour obtenir les résultats fournis dans ces tableaux, nous avons utilisé la fonction de pondération *Spline2* (*cf.* chapitre 1) pour toutes les méthodes *MLS – RKM*, et la fonction CSRBF de Wendland $C^2$ (paramètres $d = 1$ et $l = 2$) pour les méthodes RPIM et MLS/RBF.

| **Méthode meshfree** | **Position du nœud intérieur ($x_i$, $y_i$)** | Erreur $e_u$ **(%)** | Erreur $e_{u_x}$ (%) | Erreur $e_{u_y}$ (%) | Erreur $e_{\partial_x u}$ (%) | Erreur $e_{\partial_y u}$ (%) |
|---|---|---|---|---|---|---|
| EFGM | (0.5 ; 0.5) | $3.9\ 10^{-14}$ | $3.8\ 10^{-14}$ | $4.1\ 10^{-14}$ | $9.8\ 10^{-14}$ | $1.1\ 10^{-13}$ |
| | (0.5 ; 0.6) | 0.763 | $7.8\ 10^{-14}$ | 1.08 | $1.3\ 10^{-13}$ | 1.85 |
| | (0.75 ;0.75) | 0.374 | 0.374 | 0.374 | 0.538 | 0.538 |
| | (0.5 ; 0.9) | 18.7 | $2.9\ 10^{-14}$ | 26.4 | $4.3\ 10^{-14}$ | 27.1 |
| RKPM | (0.5 ; 0.5) | $3.4\ 10^{-15}$ | 0 | $4.8\ 10^{-15}$ | $1.1\ 10^{-14}$ | $1.9\ 10^{-14}$ |
| | (0.5 ; 0.6) | 0.763 | $9.9\ 10^{-15}$ | 1.08 | $3.9\ 10^{-14}$ | 1.85 |
| | (0.75 ;0.75) | 0.374 | 0.374 | 0.374 | 0.538 | 0.538 |
| | (0.5 ; 0.9) | 18.7 | $2.2\ 10^{-14}$ | 26.4 | $2.5\ 10^{-14}$ | 27.1 |
| RKI | (0.5 ; 0.5) | $1.9\ 10^{-14}$ | $1.9\ 10^{-14}$ | $1.9\ 10^{-14}$ | $1.1\ 10^{-14}$ | $1.9\ 10^{-14}$ |
| | (0.5 ; 0.6) | 0.763 | $1.1\ 10^{-14}$ | 1.08 | $3.9\ 10^{-14}$ | 1.85 |
| | (0.75 ;0.75) | 0.374 | 0.374 | 0.374 | 0.538 | 0.538 |
| | (0.5 ; 0.9) | 18.7 | $1.1\ 10^{-14}$ | 26.4 | $2.5\ 10^{-14}$ | 27.1 |
| MLS/RK | (0.5 ; 0.5) | $1.4\ 10^{-14}$ | $2.0\ 10^{-14}$ | 0 | $1.9\ 10^{-14}$ | $2.7\ 10^{-14}$ |
| | (0.5 ; 0.6) | 0.763 | $1.4\ 10^{-14}$ | 1.08 | $4.8\ 10^{-14}$ | 1.85 |
| | (0.75 ;0.75) | 0.374 | 0.374 | 0.374 | 0.538 | 0.538 |
| | (0.5 ; 0.9) | 18.7 | $1.3\ 10^{-14}$ | 26.4 | $4.7\ 10^{-14}$ | 27.1 |
| RPIM | (0.5 ; 0.5) | $5.4\ 10^{-15}$ | $5.4\ 10^{-15}$ | $5.4\ 10^{-15}$ | $1.7\ 10^{-14}$ | $1.7\ 10^{-14}$ |
| | (0.5 ; 0.6) | 2.21 | $2.2\ 10^{-14}$ | 3.12 | $1.1\ 10^{-14}$ | 4.79 |
| | (0.75 ;0.75) | 2.98 | 2.98 | 2.98 | 3.8 | 3.8 |
| | (0.5 ; 0.9) | 22.3 | $1.6\ 10^{-14}$ | 31.6 | $3.1\ 10^{-14}$ | 15.5 |
| MLS/RBF | (0.5 ; 0.5) | $1.1\ 10^{-13}$ | $8.8\ 10^{-14}$ | $1.2\ 10^{-13}$ | $1.1\ 10^{-13}$ | $1.2\ 10^{-13}$ |
| | (0.5 ; 0.6) | 37.3 | $2.0\ 10^{-14}$ | 52.8 | $4.8\ 10^{-14}$ | 59.1 |
| | (0.75 ;0.75) | 3.1 | 3.1 | 3.1 | 7.8 | 7.8 |
| | (0.5 ; 0.9) | 23.9 | $8.1\ 10^{-14}$ | 33.8 | $2.2\ 10^{-13}$ | 79.3 |

***Tabl. 2.3 : Evolution de l'erreur en fonction de la position du point intérieur ($h_{max}$ = 1.2).***

On constate au niveau du tableau 2.3, que dans le cas du domaine d'influence radial, toutes les méthodes fournissent des résultats imprécis lorsque l'arrangement nodal est dissymétrique, voire même très mauvais pour la méthode MLS/RBF. En fait, il semble que cette méthode soit très sensible à l'intégration numérique. En effet, si l'on reprend le cas radial avec par exemple le point (0.5 ; 0.6) et 2x4 points de Gauss au lieu de 2x2, on obtient de bien meilleurs résultats : $e_u$ = 0.88%, $e_{u_y}$ = 1.2%, $e_{\partial_y u}$ = 3.7% ( $e_{u_x}$ et $e_{\partial_x u}$ restant du même ordre).

Au niveau du tableau 2.4, nous avons choisi de présenter les résultats obtenus pour une position de nœud intérieur médiane, variant uniquement suivant l'axe des ordonnées. Cela nous a permis d'évaluer la précision que l'on peut attendre lorsque l'on définit des domaines d'influence rectangulaires adaptés à la dissymétrie de l'arrangement nodal. En l'occurrence, nous avons utilisé un domaine de dimensions $h_x = 0.5$, $h_y = 1$, donc parfaitement adapté à la configuration des nœuds. On voit dans ce cas qu'il est possible d'obtenir une très bonne précision.

Pour les configurations dissymétriques (nœud non milieu), on constate que les méthodes *MLS* fournissent généralement des résultats très comparables entre eux, la méthode RKPM étant peut-être légèrement plus précise que les autres dans ce test.

| **Méthode meshfree** | **Position du nœud intérieur ($x_i$, $y_i$)** | Erreur $e_u$ **(%)** | Erreur $e_{u_x}$ (%) | Erreur $e_{u_y}$ (%) | Erreur $e_{\partial_x u}$ (%) | Erreur $e_{\partial_y u}$ (%) |
|---|---|---|---|---|---|---|
| EFGM | (0.5 ; 0.6) | $1.3\ 10^{-13}$ | $1.2\ 10^{-13}$ | $1.5\ 10^{-13}$ | $1.3\ 10^{-13}$ | $4.9\ 10^{-14}$ |
| | (0.5 ; 0.9) | $2.3\ 10^{-13}$ | $3.0\ 10^{-13}$ | $1.3\ 10^{-13}$ | $6.4\ 10^{-13}$ | $5.4\ 10^{-14}$ |
| RKPM | (0.5 ; 0.6) | $2.0\ 10^{-14}$ | $2.2\ 10^{-14}$ | $1.9\ 10^{-14}$ | $4.7\ 10^{-14}$ | $2.3\ 10^{-14}$ |
| | (0.5 ; 0.9) | $4.8\ 10^{-14}$ | $5.1\ 10^{-14}$ | $4.4\ 10^{-14}$ | $5.9\ 10^{-13}$ | $1.1\ 10^{-13}$ |
| RKI | (0.5 ; 0.6) | $1.5\ 10^{-14}$ | $1.1\ 10^{-14}$ | $1.7\ 10^{-14}$ | $4.7\ 10^{-14}$ | $2.3\ 10^{-14}$ |
| | (0.5 ; 0.9) | $7.2\ 10^{-14}$ | $8.9\ 10^{-14}$ | $4.8\ 10^{-14}$ | $5.9\ 10^{-13}$ | $1.1\ 10^{-13}$ |
| MLS/RK | (0.5 ; 0.6) | $2.3\ 10^{-14}$ | $1.4\ 10^{-14}$ | $2.9\ 10^{-14}$ | $5.3\ 10^{-14}$ | $3.0\ 10^{-14}$ |
| | (0.5 ; 0.9) | $2.3\ 10^{-13}$ | $3.1\ 10^{-13}$ | $7.6\ 10^{-14}$ | $1.1\ 10^{-12}$ | $9.8\ 10^{-14}$ |
| RPIM | (0.5 ; 0.6) | $1.4\ 10^{-14}$ | $2.0\ 10^{-14}$ | 0 | $7.91\ 10^{-15}$ | $1.1\ 10^{-14}$ |
| | (0.5 ; 0.9) | $3.0\ 10^{-14}$ | $1.0\ 10^{-14}$ | $4.0\ 10^{-14}$ | $1.4\ 10^{-14}$ | $5.6\ 10^{-15}$ |
| MLS/RBF | (0.5 ; 0.6) | $7.0\ 10^{-14}$ | $5.2\ 10^{-14}$ | $8.4\ 10^{-14}$ | $8.4\ 10^{-13}$ | $2.5\ 10^{-13}$ |
| | (0.5 ; 0.9) | $1.5\ 10^{-12}$ | $2.0\ 10^{-12}$ | $5.7\ 10^{-13}$ | $8.0\ 10^{-12}$ | $1.7\ 10^{-12}$ |

***Tabl. 2.4 : Evolution de l'erreur en fonction de la position du point intérieur ($h_x$=0.5, $h_y$=1).***

On voit que l'utilisation d'un domaine d'influence rectangulaire dont les dimensions sont choisies de manière à suivre la position du nœud intérieur, permet cette fois d'obtenir une très bonne précision. En revanche, les résultats ne sont pas améliorés lorsque la position du nœud intérieur est très excentrée ( $x_i = y_i = 0.75$ ).

***Cas particulier de la méthode PIM :***

Dans le cas de la méthode PIM et pour une base primaire linéaire, l'arrangement nodal de ce test conduit à une matrice des moments singulière au niveau des points d'intégration. Nous avons donc comparé les résultats obtenus en utilisant les trois techniques les plus simples présentées dans la section §2.2.3, à savoir la méthode RPIM, le déplacement aléatoire des nœuds voisins et la transformation du système de coordonnées. Nous avons préféré ces techniques du fait que contrairement à l'algorithme MTA, elles ne nécessitent pas ou très peu d'opérations supplémentaires.

Nous avons choisi d'effectuer notre comparaison en utilisant un support d'intégration rectangulaire de dimensions fixes $h_x = 0.6$ et $h_y = 0.8$ (*cf.* Tabl. 2.5).

D'une manière générale, on constate que les résultats sont plus précis avec la méthode RPIM, l'utilisation des fonctions de base radiales à support global (GSRBF) fournissant les meilleurs résultats. Néanmoins, la précision de cette dernière méthode varie de manière importante avec le nombre de nœuds inclus dans le voisinage du point d'intégration, les performances optimales étant atteintes lorsque le voisinage couvre complètement le domaine d'étude (*cf.* section §2.3.2.3).

La technique de transformation du système de coordonnées, dont l'angle était choisi de manière aléatoire, est également plus efficace ici que celle avec déplacement aléatoire des nœuds voisins. Nous avons par ailleurs constaté que pour cette dernière technique, les résultats sont très sensibles à la valeur du coefficient $\varepsilon$ choisi.

| **Méthode PIM** | **Position du nœud intérieur ($x_i$, $y_i$)** | Erreur $e_u$ **(%)** | Erreur $e_{u_x}$ (%) | Erreur $e_{u_y}$ (%) | Erreur $e_{\partial_x u}$ (%) | Erreur $e_{\partial_y u}$ (%) |
|---|---|---|---|---|---|---|
| *RPIM - CSRBF (Wendland $C^2$)* | (0.5 ; 0.5) | $1.4\ 10^{-14}$ | $1.5\ 10^{-14}$ | $1.2\ 10^{-14}$ | $1.9\ 10^{-14}$ | 0 |
| | (0.5 ; 0.6) | $3.1\ 10^{-14}$ | $3.5\ 10^{-14}$ | $2.8\ 10^{-14}$ | $1.6\ 10^{-14}$ | $1.1\ 10^{-14}$ |
| | (0.6 ; 0.8) | 7.25 | 10.3 | $2.4\ 10^{-14}$ | 28 | $1.6\ 10^{-14}$ |
| *RPIM - GSRBF (Multiquadrics $\beta = 1, c = 0.5$)* | (0.5 ; 0.5) | 0 | 0 | 0 | 0 | 0 |
| | (0.5 ; 0.6) | 0 | 0 | 0 | 0 | 0 |
| | (0.6 ; 0.8) | 7.25 | 10.3 | $3.2\ 10^{-14}$ | 28 | 0 |
| *Transformation des coordonnées* | (0.5 ; 0.5) | $2.3\ 10^{-14}$ | $1.4\ 10^{-14}$ | $2.9\ 10^{-14}$ | 0 | 0 |
| | (0.5 ; 0.6) | $3.5\ 10^{-14}$ | $4\ 10^{-14}$ | $2.9\ 10^{-14}$ | $1.6\ 10^{-14}$ | 0 |
| | (0.6 ; 0.8) | 7.25 | 10.3 | $2.6\ 10^{-14}$ | 28 | $1.6\ 10^{-14}$ |
| *Déplacement des nœuds* $\varepsilon = 10^{-15}$ | (0.5 ; 0.5) | $8.6\ 10^{-14}$ | $10^{-13}$ | $6.3\ 10^{-14}$ | $1.1\ 10^{-13}$ | $6.7\ 10^{-14}$ |
| | (0.5 ; 0.6) | $10^{-13}$ | $1.1\ 10^{-13}$ | $8.7\ 10^{-14}$ | $1.2\ 10^{-13}$ | $6.7\ 10^{-14}$ |
| | (0.6 ; 0.8) | 7.25 | 10.3 | $8.8\ 10^{-14}$ | 28 | $3.9\ 10^{-14}$ |

***Tabl. 2.5 : Evolution de l'erreur pour la méthode PIM ($h_x$=0.6, $h_y$=0.8).***

### *2.3.2.3 Influence de l'intégration spatiale*

Ce patch test vise à évaluer l'influence de l'intégration spatiale sur les résultats obtenus pour un arrangement nodal irrégulier.

Nous avons considéré ici le domaine $\Omega = [0,2] \times [-1,1]$ avec 15 nœuds au total, dont 8 nœuds répartis sur la frontière (*cf.* Fig. 2.20), et 2x2 cellules d'intégration. Les autres caractéristiques (base primaire, matériau, conditions aux limites) sont identiques à celles du patch test à 5 nœuds. Nous avons par ailleurs utilisé la méthode RKPM avec une interpolation sur tout le domaine, une fonction de pondération sinusoïdale et suivant les cas testés, un domaine d'influence rectangulaire ou radial.

Nous nous sommes tout d'abord intéressés à l'évolution de l'erreur relative sur les déplacements et ses dérivées, lorsque le nombre de points de Gauss varie, pour un support d'intégration rectangulaire de dimensions fixées au départ à : $h_x = h_y = 1$.

Au niveau de ce test, la règle empirique (2.1) conduirait à adopter la configuration de 4 ou 5 points de Gauss par cellule, avec un total de 4 cellules (2x2). Néanmoins, on voit que la précision continue à augmenter lorsque le nombre de points d'intégration utilisés est plus important, avec une stabilisation à partir de 6x6 points de Gauss (*cf.* Fig. 2.9).

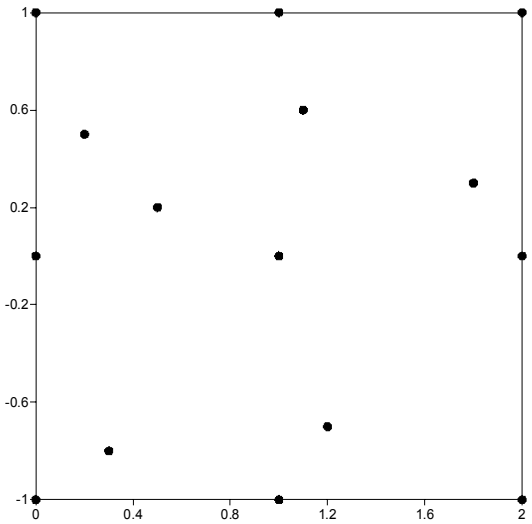



***Fig. 2.8 : Patch test linéaire à 15 nœuds (arrangement nodal irrégulier).***

Ensuite pour un nombre de points de Gauss donné, nous nous sommes intéressés à l'évolution de l'erreur relative sur les déplacements et ses dérivées, lorsque la forme du support d'intégration varie (support rectangulaire ou radial) et que le rayon du domaine est choisi fixe ou variable (*cf*. Fig. 2.10 et Fig. 2.11). Pour le support radial, le rayon maximal choisi était : $h_{max} = 1.4$, et pour le support rectangulaire : $h_x = h_y = 1$.

De même que pour le test à 5 nœuds (cf. §2.3.2.2), on constate que l'utilisation d'un support rectangulaire améliore généralement la précision par rapport à celle du support radial. C'est également le cas avec l'utilisation d'un rayon variable, du fait d'une meilleure adaptation au cas simulé ici, à savoir un arrangement nodal irrégulier.

Il faut cependant noter que d'une manière générale dans les patch tests réalisés, l'erreur obtenue pour le champ dérivé avec l'interpolation globale sur tout le domaine, c'est-à-dire en modifiant les fonctions de pondération, demeure relativement forte par rapport notamment à une technique utilisant les multiplicateurs de Lagrange [AUB 97]. Cette erreur est d'autant plus forte que l'arrangement nodal est irrégulier, ce qui induit de ce fait une erreur non négligeable sur le champ solution lui-même.

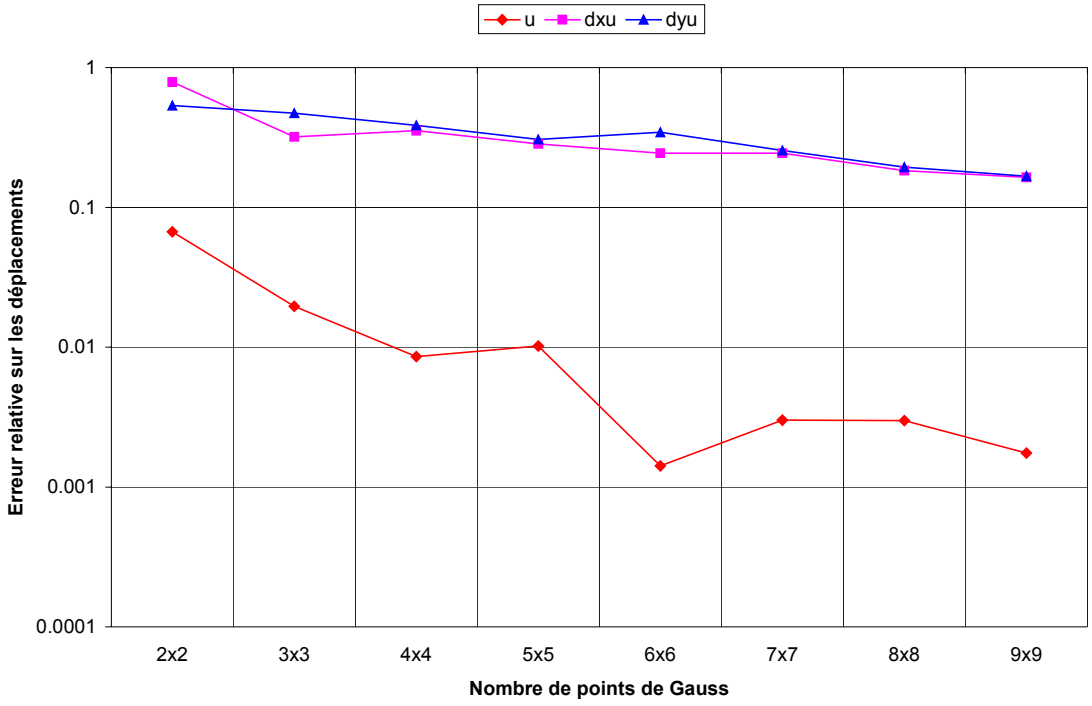



*Fig. 2.9 : Méthode RKPM : erreur relative sur les déplacements en fonction du nombre de points d'intégration (domaine rectangulaire de dimensions fixes)*

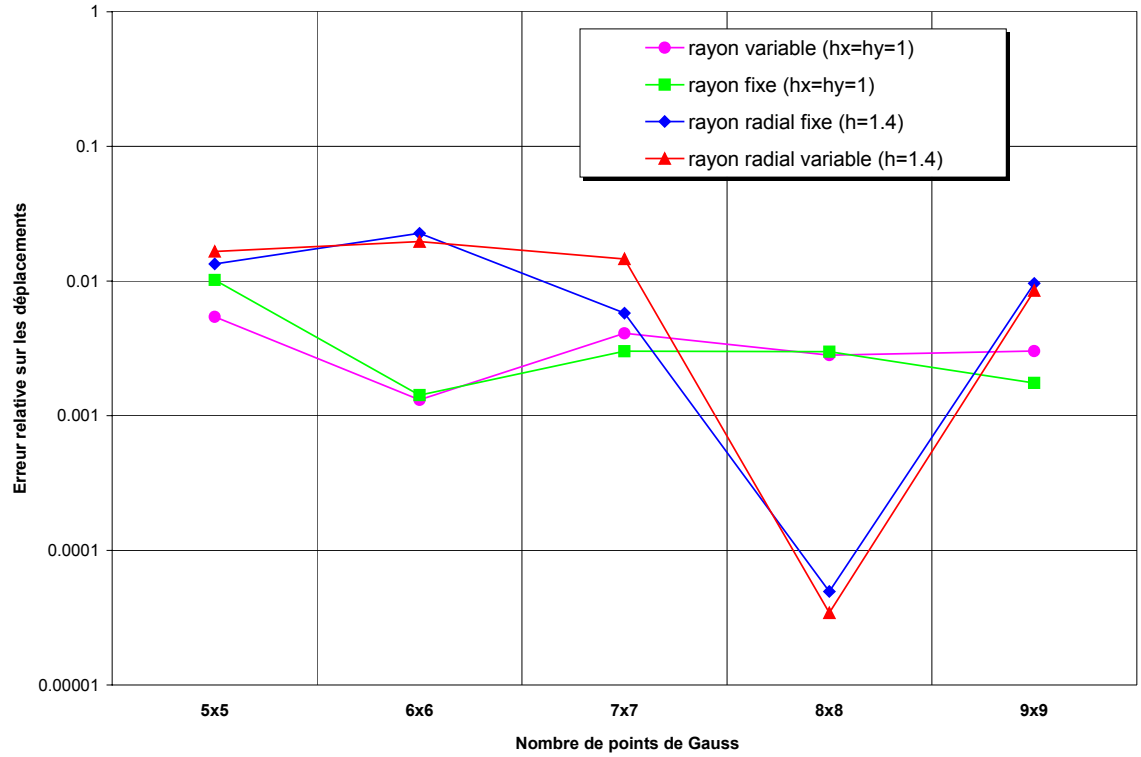



*Fig. 2.10 : Méthode RKPM : erreur relative sur les déplacements suivant la forme et le type de rayon de domaine d'influence*

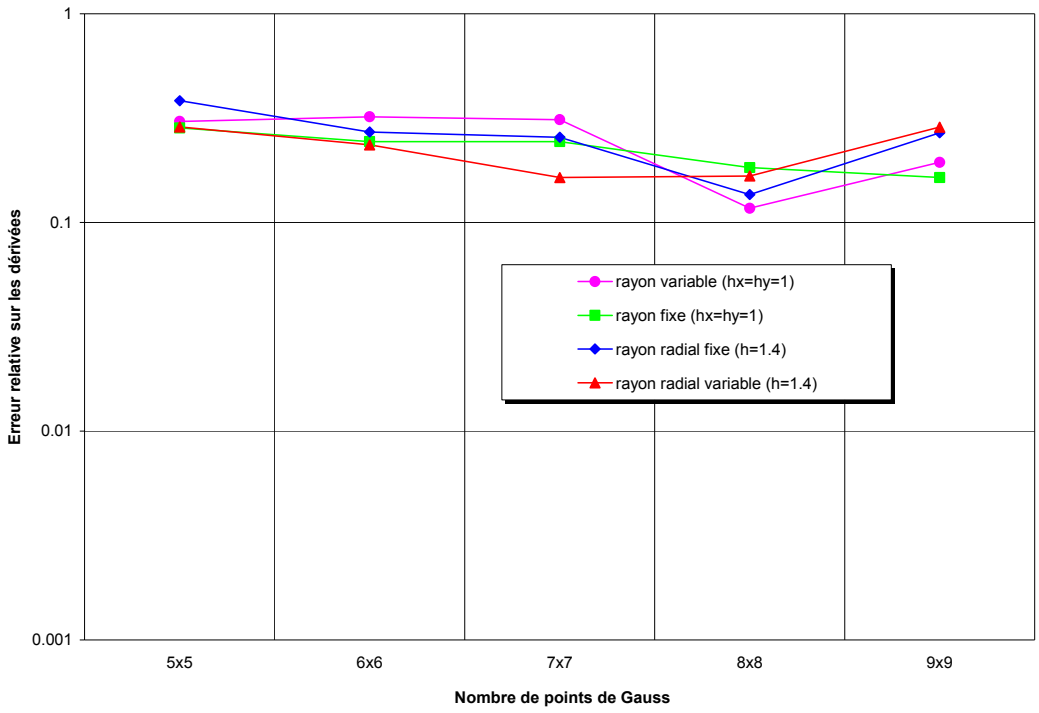



***Fig. 2.11 : Méthode RKPM : erreur relative sur la dérivée des déplacements suivant la forme et le type de rayon de domaine d'influence***

### *2.3.2.4 Influence de la technique d'interpolation*

Dans ce test, nous nous sommes intéressés à l'effet de l'utilisation des différentes techniques d'interpolation sur la précision numérique. Pour ce faire, nous avons considéré un domaine $\Omega = [0,2] \times [0,2]$ avec 36 nœuds au total, dont 20 nœuds répartis sur la frontière, pour un arrangement nodal régulier ou irrégulier (*cf.* Fig. 2.12). Les autres caractéristiques (base primaire, matériau, conditions aux limites) sont identiques à celles des tests précédents. Les domaines d'influence et supports des fonctions de forme étaient par ailleurs rectangulaires de dimensions constantes.

***Arrangement nodal régulier :***

L'intégration spatiale comprenaient 5x5 points de Gauss pour une structure d'intégration composée de 5x5 cellules régulières (*cf.* Fig. 2.13). Dans ce cas, le domaine d'influence considéré avait pour dimensions : $h_x = h_y = 0.4$ .

Les résultats comparés entre les méthodes EFGM et RKPM pour une interpolation sur tout le domaine et en utilisant différentes fonctions de pondération montrent que quelle que soit la fonction de pondération utilisée, on obtient toujours de meilleurs résultats avec la méthode RKPM (*cf.* Fig. 2.14 à Fig. 2.16). On notera également que pour une méthode donnée, la variabilité des résultats suivant la fonction de pondération utilisée est relativement importante.

Les résultats comparés entre les méthodes RPIM et MLS/RBF en utilisant des fonctions CSRBF et GSRBF montrent que quelle que soit la fonction de base radiale utilisée, on obtient toujours de meilleurs résultats avec la méthode RPIM (*cf*. Fig. 2.17 à Fig. 2.19).

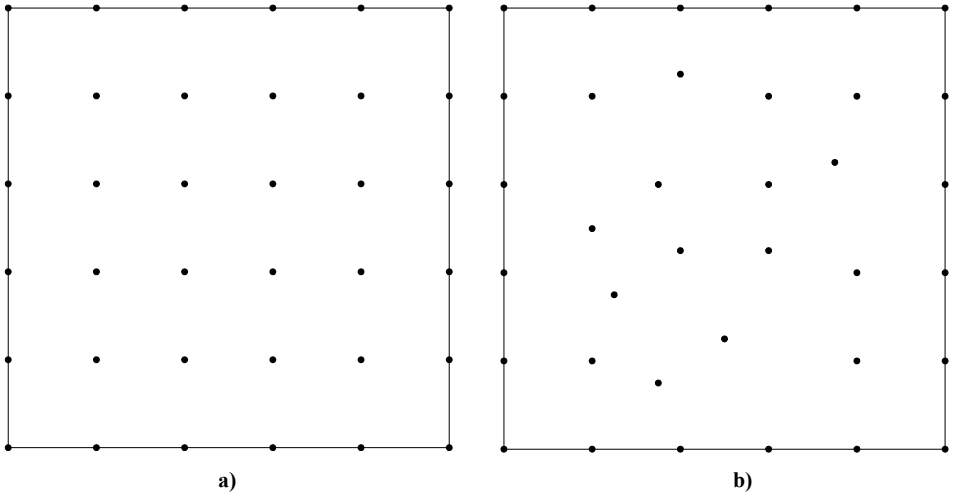



***Fig. 2.12: Patch test à 36 nœuds : arrangement nodal a) régulier ; b) irrégulier.***

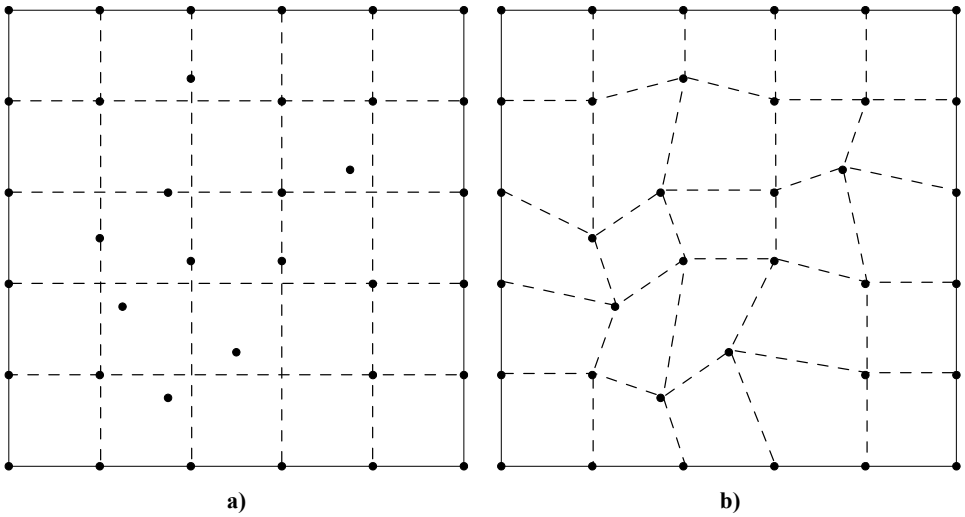
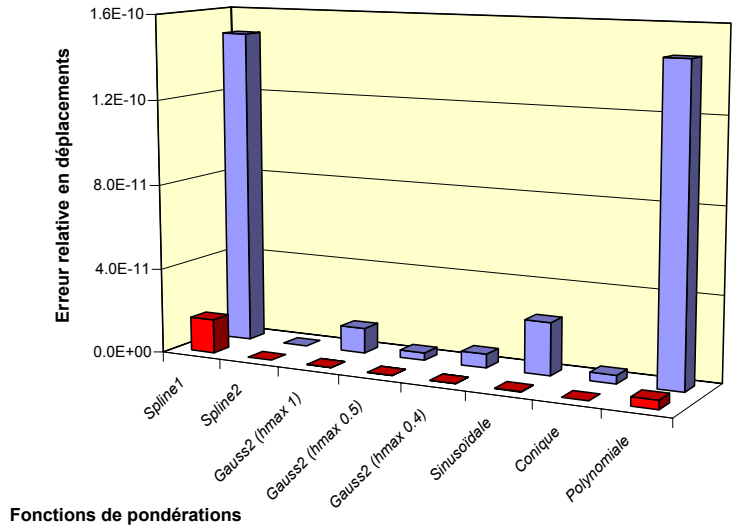
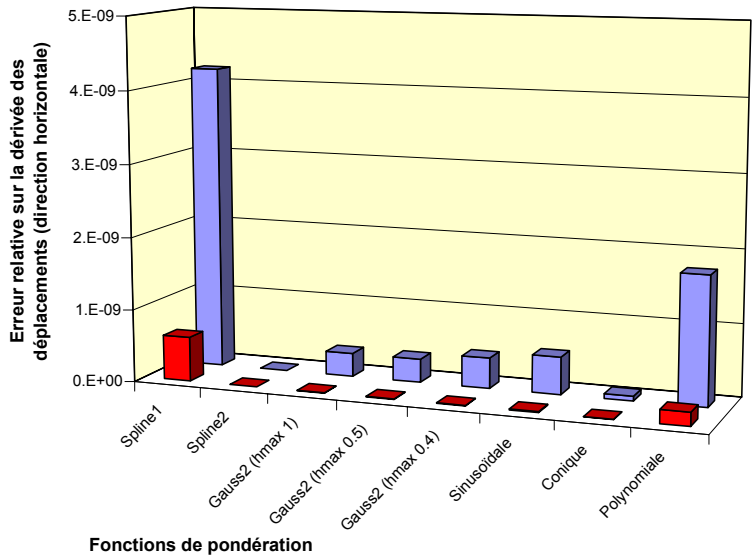



***Fig. 2.13: Structures d'intégration : a) grille régulière ; b) « maillage ».***



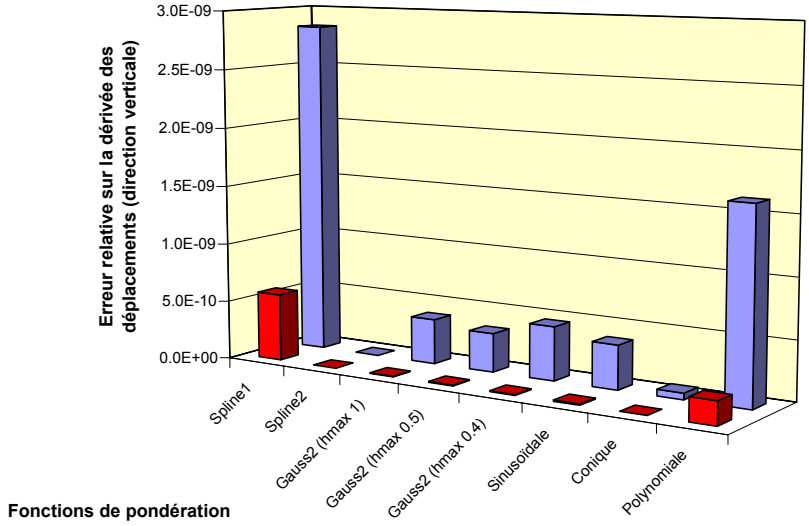

| | Spline1 | Spline2 | Gauss2 (hmax 1) | Gauss2 (hmax 0.5) | Gauss2 (hmax 0.4) | Sinusoïdale | Conique | Polynomiale |
|---|---|---|---|---|---|---|---|---|
| RKPM | 1.59E-11 | 0.00E+00 | 6.20E-14 | 6.86E-14 | 5.60E-14 | 2.64E-13 | 3.53E-14 | 4.19E-12 |
| EFGM | 1.49E-10 | 0.00E+00 | 1.18E-11 | 3.57E-12 | 6.48E-12 | 2.46E-11 | 3.95E-12 | 1.46E-10 |

***Fig. 2.14 : Méthodes EFGM et RKPM interpolantes : erreur relative sur les déplacements.***



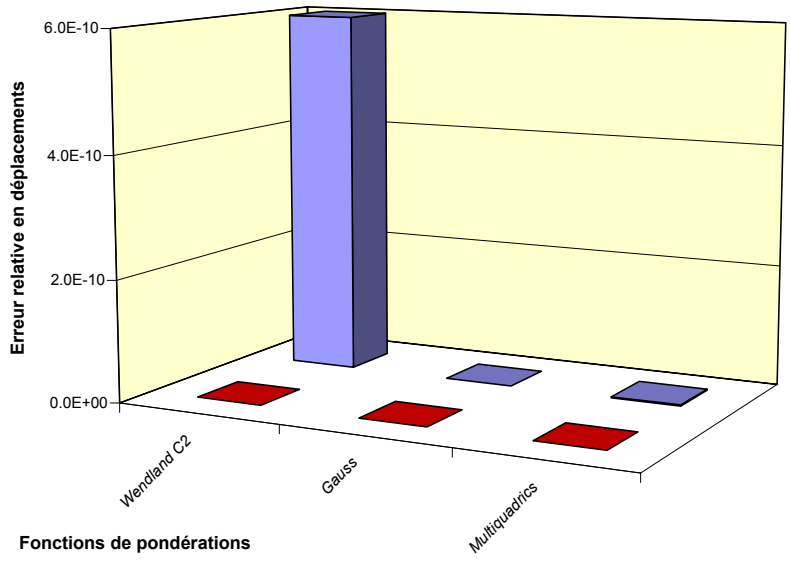

| | Spline1 | Spline2 | Gauss2 (hmax 1) | Gauss2 (hmax 0.5) | Gauss2 (hmax 0.4) | Sinusoïdale | Conique | Polynomiale |
|---|---|---|---|---|---|---|---|---|
| RKPM | 6.10E-10 | 0.00E+00 | 3.88E-12 | 7.33E-12 | 7.43E-12 | 1.47E-11 | 1.12E-12 | 1.87E-10 |
| EFGM | 4.20E-09 | 1.25E-14 | 3.25E-10 | 3.24E-10 | 4.20E-10 | 5.13E-10 | 6.48E-11 | 1.76E-09 |

***Fig. 2.15 : Méthodes EFGM et RKPM interpolantes : erreur relative*** $e_{\partial_x u}$.



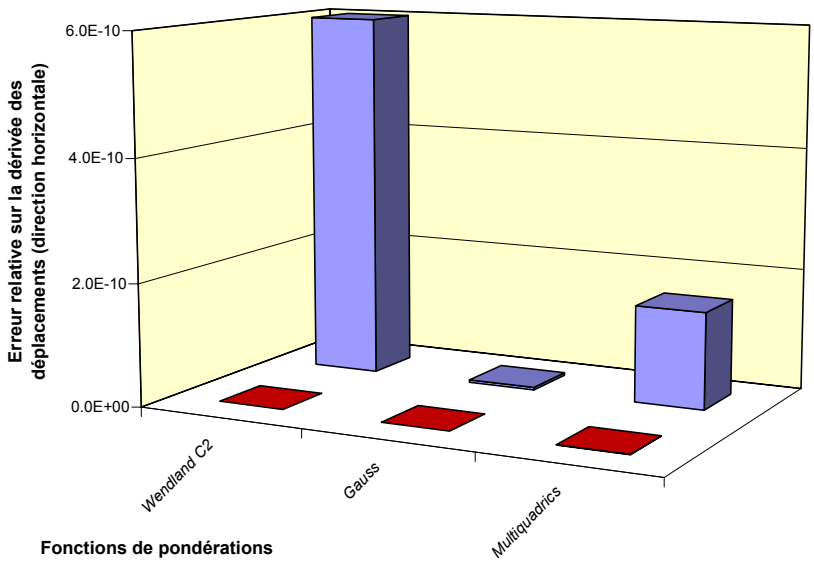

| | Spline1 | Spline2 | Gauss2 (hmax 1) | Gauss2 (hmax 0.5) | Gauss2 (hmax 0.4) | Sinusoïdale | Conique | Polynomiale |
|---|---|---|---|---|---|---|---|---|
| RKPM | 5.70E-10 | 0.00E+00 | 5.19E-12 | 7.06E-12 | 7.61E-12 | 1.09E-11 | 8.91E-13 | 1.97E-10 |
| EFGM | 2.84E-09 | 1.15E-14 | 3.82E-10 | 3.29E-10 | 4.56E-10 | 3.73E-10 | 5.46E-11 | 1.62E-09 |

***Fig. 2.16 : Méthodes EFGM et RKPM interpolantes : erreur relative*** $e_{\partial_y u}$.




| | Wendland C2 | Gauss | Multiquadrics |
|---|---|---|---|
| RPIM | 6.63E-14 | 7.18E-14 | 9.24E-14 |
| MLS/RBF | 3.28E-08 | 1.90E-13 | 1.49E-12 |

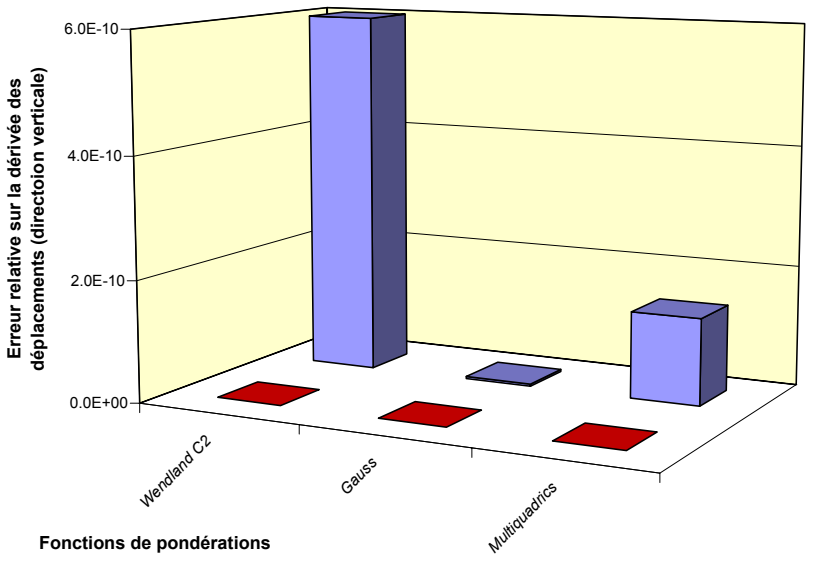
***Fig. 2.17 : Méthodes RPIM et MLS/RBF : erreur relative sur les déplacements (Gauss avec*** $b = 2$ ***et Multiquadrics avec*** $\beta = 1, c = 0.5$***).***




| | Wendland C2 | Gauss | Multiquadrics |
|---|---|---|---|
| RPIM | 1.13E-13 | 2.29E-13 | 3.52E-13 |
| MLS/RBF | 1.61E-06 | 4.45E-12 | 1.58E-10 |

***Fig. 2.18 : Méthodes RPIM et MLS/RBF : erreur relative*** $e_{\partial_x u}$

***(Gauss avec*** $b = 2$ ***et Multiquadrics avec*** $\beta = 1, c = 0.5$ ***).***




| | Wendland C2 | Gauss | Multiquadrics |
|---|---|---|---|
| RPIM | 1.48E-13 | 2.15E-13 | 2.95E-13 |
| MLS/RBF | 1.51E-06 | 3.82E-12 | 1.43E-10 |

***Fig. 2.19 : Méthodes RPIM et MLS/RBF : erreur relative*** $e_{\partial_y u}$.

***(Gauss avec*** $b = 2$ ***et Multiquadrics avec*** $\beta = 1, c = 0.5$ ***).***

***Arrangement nodal irrégulier :***

Trois dimensions $h$ du domaine d'influence rectangulaire ont été successivement utilisées : 0.6, 1 et 1.5 ($h = h_x = h_y$). L'intégration spatiale comprenaient 5x5 points de Gauss pour une structure d'intégration composée de 5x5 cellules disposées sous forme de grille régulière ou sous forme de « maillage » passant par les nœuds (*cf*. Fig. 2.13).

Trois configurations ont été envisagées pour comparer les techniques d'interpolation :

- une interpolation globale sur tout le domaine, en utilisant la méthode RKPM avec une fonction de pondération polynomiale modifiée par la relation (1.44), et la méthode RPIM avec une fonction CSRBF (Wendland $C^2$) ou GSRBF (Multiquadrics avec $\beta = 1, c = 0.5$) ;

- une interpolation uniquement localisée au niveau des frontières, en utilisant la méthode RKPM avec une fonction de pondération polynomiale (non modifiée) pour l'intérieur du domaine et 20 éléments finis linéaires de type 1D sur la frontière (*cf*. Fig. 2.20) ;

- enfin, une interpolation localisée sur un domaine comprenant la frontière, en utilisant la méthode RKPM avec une fonction de pondération polynomiale pour l'intérieur du domaine et 16 éléments finis linéaires de type 2D (*cf*. Fig. 2.20).

Quelles que soient les dimensions du domaine d'influence, on constate que le couplage de la méthode RKPM avec des éléments finis 1D produit de meilleurs résultats que celui utilisant des éléments finis 2D (*cf*. Fig. 2.21). Par ailleurs, on remarque que dans le cas de l'interpolation avec des fonctions de pondération modifiées, l'erreur obtenue en intégrant avec une structure de type maillage est plus faible que celle obtenue avec une grille régulière non nécessairement connectée aux nœuds de la discrétisation.

Par ailleurs, on voit que la méthode RPIM avec fonction GSRBF est la plus performante, à condition d'utiliser de considérer une taille de support importante (*cf*. Fig. 2.22).

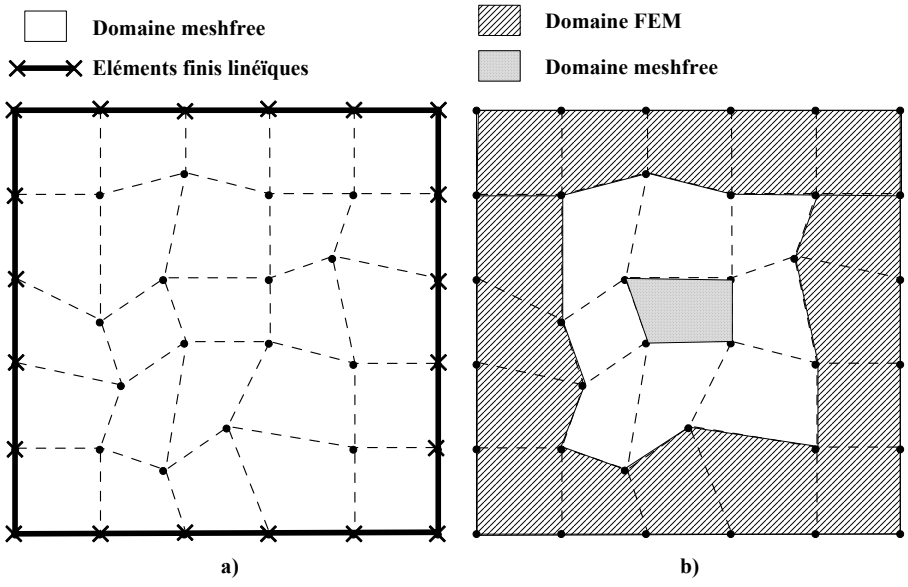
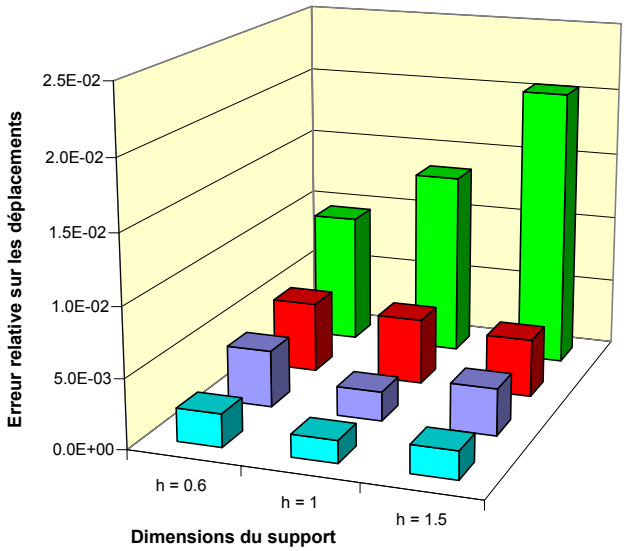
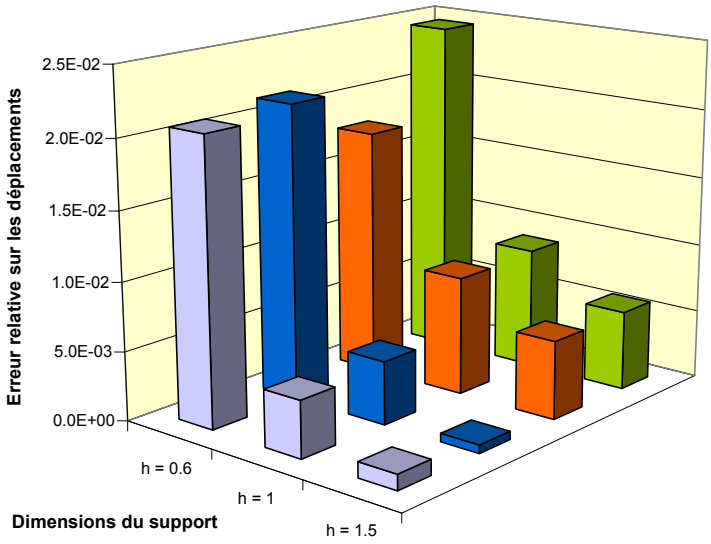



***Fig. 2.20: Principe du couplage meshfree – FEM : a) 1D ; b) 2D.***




| | h = 0.6 | h = 1 | h = 1.5 |
|---|---|---|---|
| RKPM - FEM1D | 0.00239 | 0.00162 | 0.00202 |
| RKPM - Interpol (maillage) | 0.00413 | 0.00213 | 0.00339 |
| RKPM - Interpol (grille) | 0.00514 | 0.00482 | 0.00423 |
| RKPM - FEM2D | 0.00957 | 0.0135 | 0.0205 |

*Fig. 2.21: Méthode RKPM : erreur relative sur les déplacements pour différentes techniques d'interpolation.*




| | h = 0.6 | h = 1 | h = 1.5 |
|---|---|---|---|
| RPIM MQ (grille) | 0.0207 | 0.00411 | 0.00113 |
| RPIM MQ (maillage) | 0.0214 | 0.00458 | 0.000605 |
| RPIM WC2 (maillage) | 0.0179 | 0.0086 | 0.00568 |
| RPIM WC2 (grille) | 0.0245 | 0.00883 | 0.00577 |

*Fig. 2.22: Méthode RPIM : erreur relative sur les déplacements.*

### 2.3.3 Plaque infinie avec un trou circulaire

On considère cette fois, une plaque homogène infinie percée par un trou de rayon *a* centré à l'origine, et soumise à une traction unidirectionnelle constante. Ce test est par ailleurs réalisé en élasticité linéaire isotrope avec l'hypothèse des transformations infinitésimales et dans le cadre des déformations planes.

Du fait des symétries, seul un quart de la plaque sera modélisé (*cf.* Fig. 2.23) et des conditions de Neumann sont imposées aux limites du domaine ($x=5, y=5$), les tractions $\underline{\bar{t}}$ étant calculées à partir de la solution exacte ci-après :

$$\begin{aligned}
\sigma_{xx} &= T_x\left\{1-\frac{a^2}{r^2}\left[\frac{3}{2}\cos(2\theta)+\cos(4\theta)\right]+\frac{3a^4}{2r^4}\cos(4\theta)\right\}\\
\sigma_{yy} &= -T_x\left\{\frac{a^2}{r^2}\left[\frac{1}{2}\cos(2\theta)-\cos(4\theta)\right]+\frac{3a^4}{2r^4}\cos(4\theta)\right\}\\
\sigma_{xy} &= -T_x\left\{\frac{a^2}{r^2}\left[\frac{1}{2}\sin(2\theta)+\sin(4\theta)\right]-\frac{3a^4}{2r^4}\sin(4\theta)\right\}
\end{aligned} \tag{2.30}$$

Les déplacements exacts sont également donnés par les relations :

$$\begin{aligned}
u_x &= \frac{T_x}{4\mu}\left\{r\left[\frac{(\kappa-1)}{2}+\cos(2\theta)\right]+\frac{a^2}{r}\left[1+(1+\kappa)\cos(2\theta)\right]-\frac{a^4}{r^3}\cos(2\theta)\right\}\\
u_y &= \frac{T_x}{4\mu}\left[(1-\kappa)\frac{a^2}{r}-r-\frac{a^4}{r^3}\right]\sin(2\theta)
\end{aligned} \tag{2.31}$$

avec : $\kappa = 3-4\nu$ en déformations planes et $\nu$, le coefficient de Poisson.

Remarque :

D'un point de vue pratique, la mise en œuvre des conditions aux limites de type Neumann, nécessite d'utiliser des cellules d'intégration linéiques réparties le long de la frontière concernée.

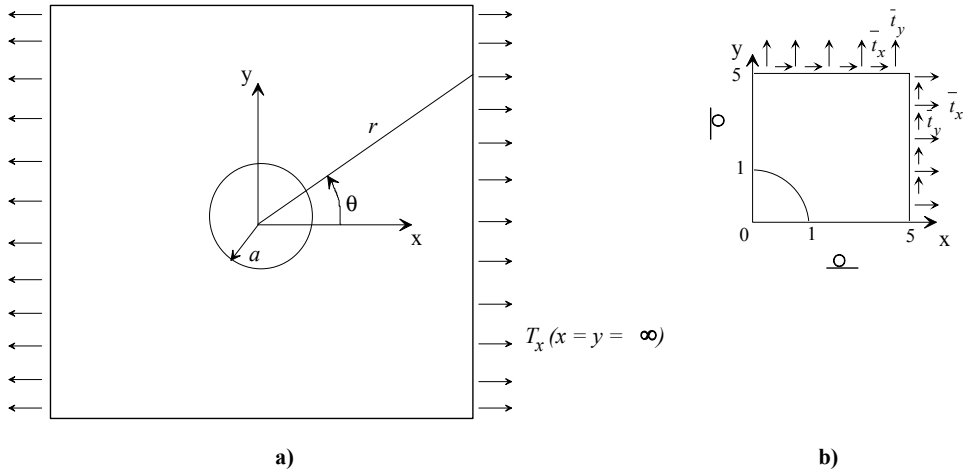



***Fig. 2.23: Plaque avec un trou circulaire : a) Plaque infinie ; b) Modèle***

Deux discrétisations ont été envisagées ici : l'une comprenait 432 nœuds et l'autre, 995 (*cf*. Fig. 2.24 et Fig. 2.25). Les domaines d'influence étaient de type rectangulaire avec de dimension variable (au minimum $h_x = h_y = 0.3$ dans les deux cas). De plus, l'intégration réalisée est de type Gauss avec une structure de type « maillage » comprenant 2x2 points d'intégration.

La comparaison entre la solution analytique et les résultats obtenus pour trois méthodes meshfree différentes, à savoir EFGM, RKPM et RPIM, ainsi que pour la méthode des éléments finis (FEM), montrent que comme on pouvait s'y attendre, la précision augmente avec le nombre de nœuds de la discrétisation. On voit en effet que toutes les méthodes fournissent des résultats très semblables dans le cas de la plaque à 995 nœuds. En revanche, les résultats sont plus dispersés pour la plaque à 432 nœuds, voire même imprécis dans le cas de la méthode RKPM (*cf*. Fig. 2.26 à Fig. 2.30).

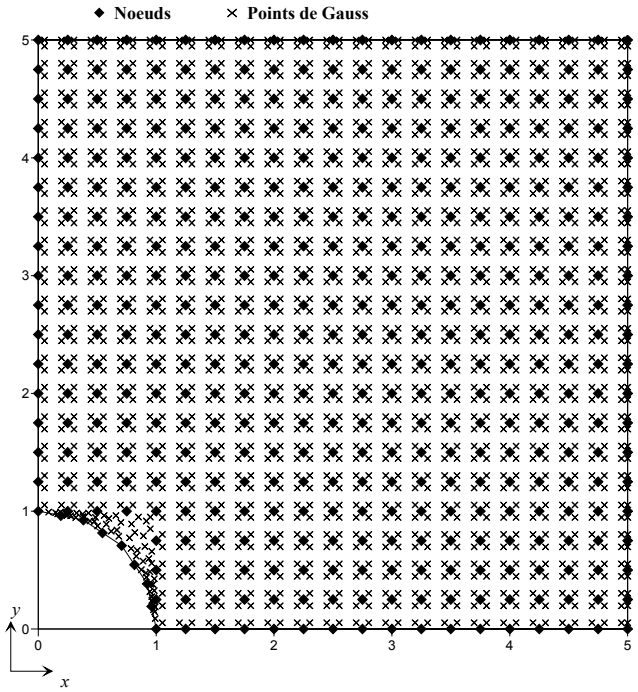



*Fig. 2.24: Plaque à 432 nœuds : arrangement nodal et points d'intégration.*

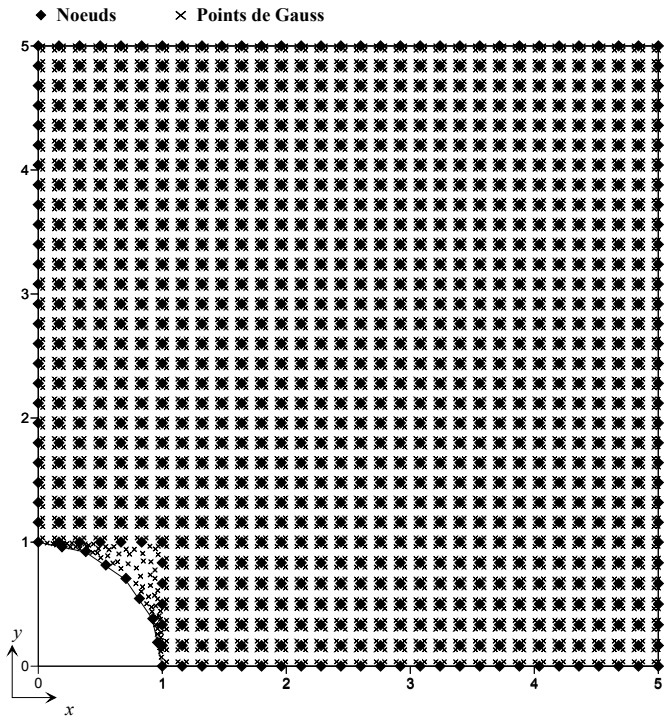



*Fig. 2.25: Plaque à 995 nœuds : arrangement nodal et points d'intégration.*

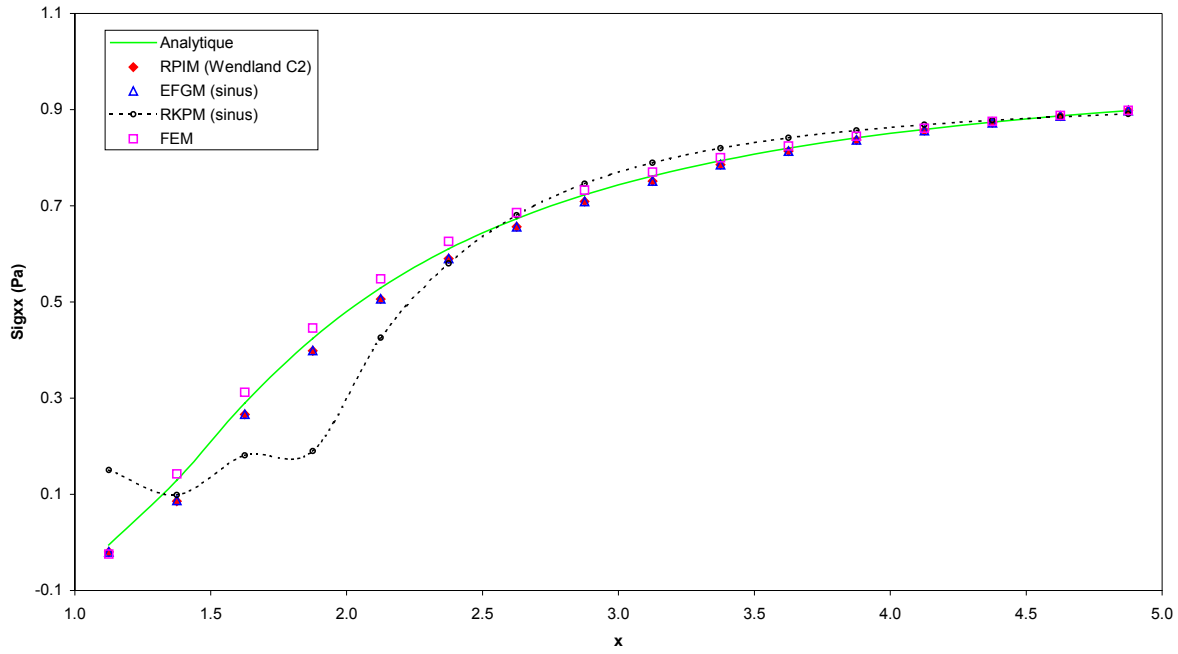



***Fig. 2.26: Contraintes $\sigma_{xx}$ pour le bord y = 0.125 (plaque à 432 nœuds).***

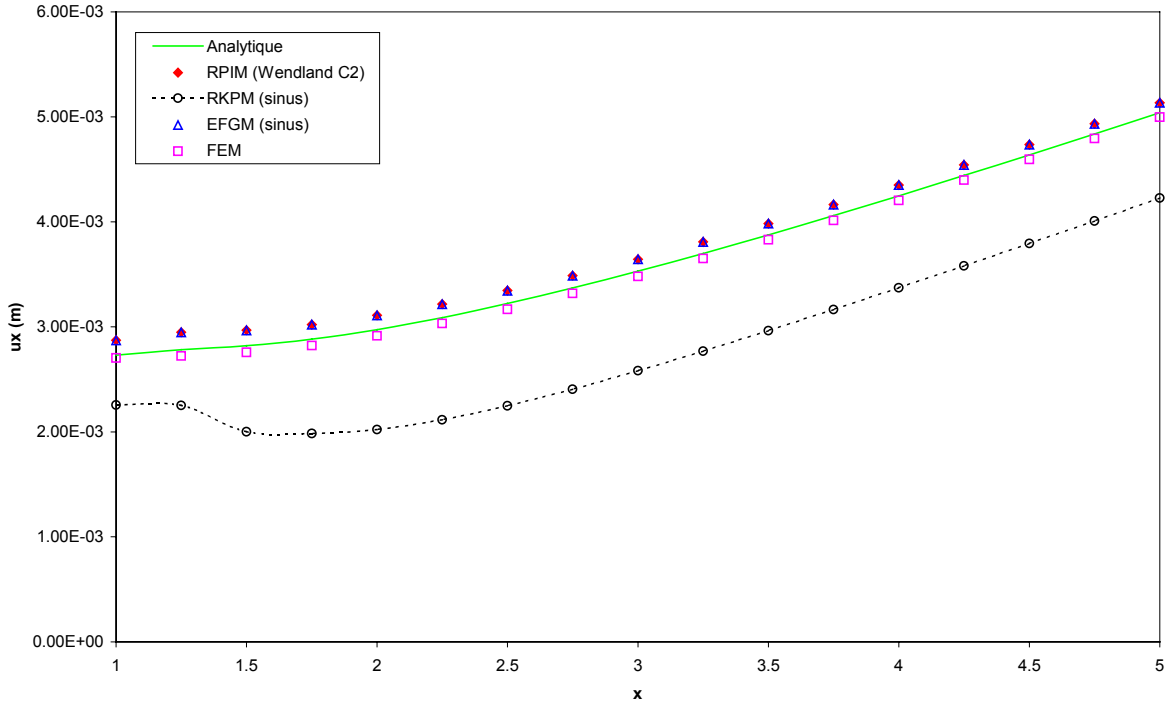



***Fig. 2.27: Déplacement $u_x$ pour le bord inférieur (y = 0, plaque à 432 nœuds).***

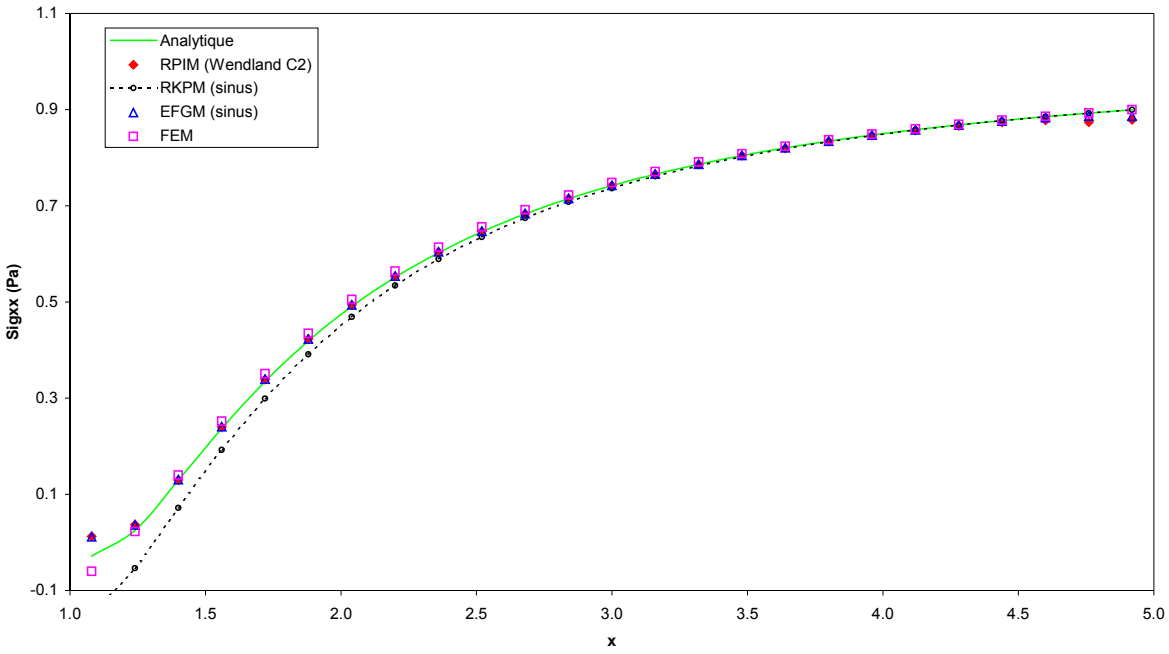



*Fig. 2.28: Contraintes $\sigma_{xx}$ pour le bord y = 0.08 (plaque à 995 nœuds).*

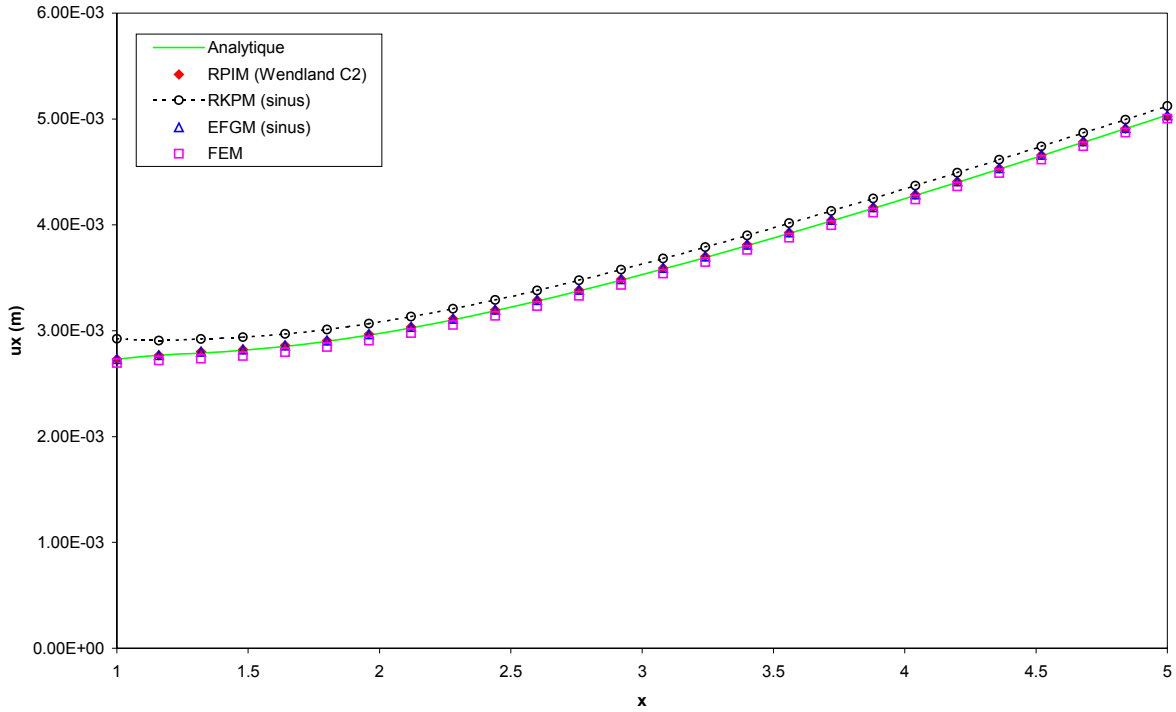



*Fig. 2.29: Déplacement $u_x$ pour le bord inférieur (y = 0, plaque à 995 nœuds).*

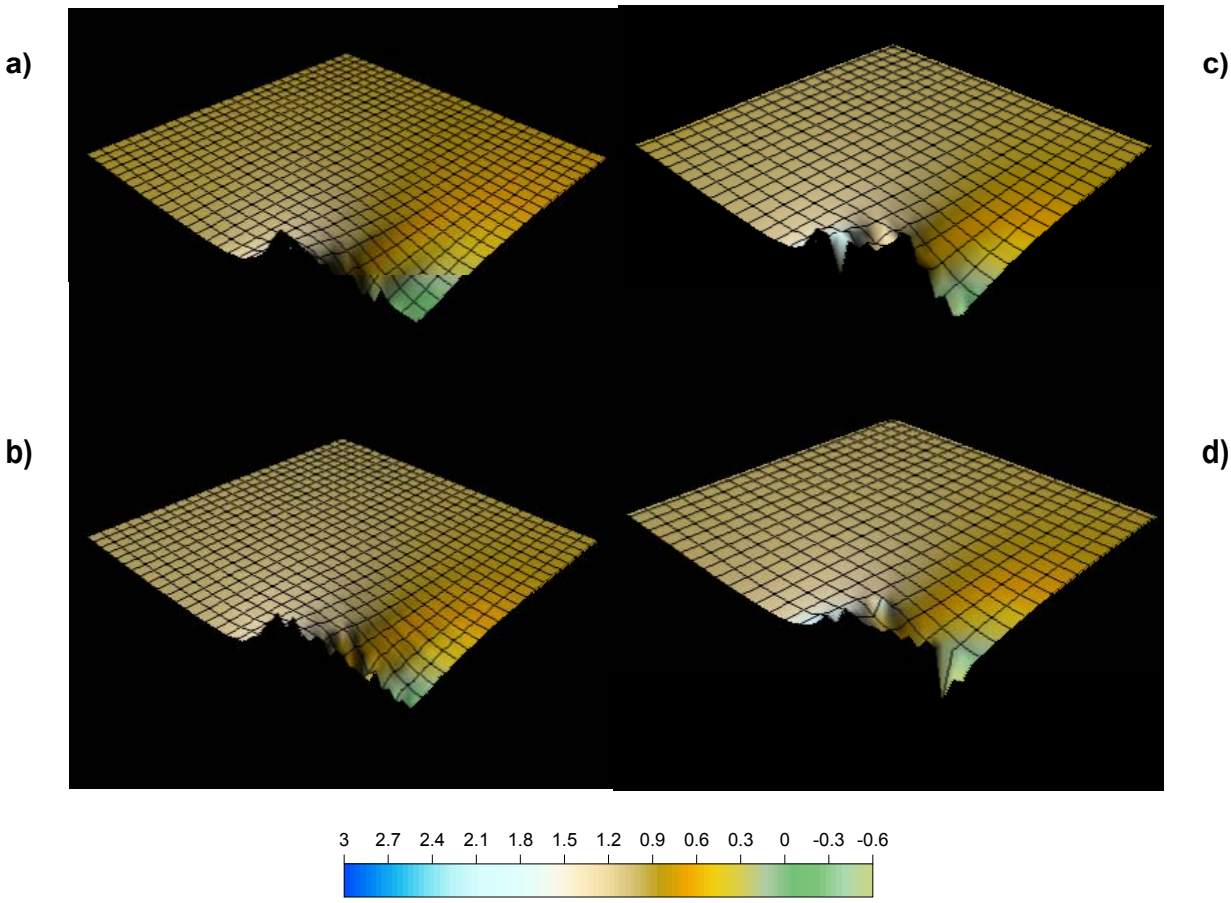



*Fig. 2.30: Contrainte $\sigma_{xx}$ : a) Analytique ; b) Méthode FEM ; c) Méthode RKPM ; d) Méthode EFGM (plaque à 995 nœuds).*

## 2.4 CONCLUSIONS

Ce chapitre nous a permis d'une part de rappeler les difficultés rencontrées lors de la mise en œuvre numérique des méthodes meshfree, aussi bien au niveau algorithmique, que dans le choix des techniques les mieux appropriées pour simuler un problème donné.

D'autre part, les résultats obtenus pour des simulations réalisées avec le logiciel *MoveFree*, dans le cas de problèmes mécaniques simples nous permettent de tirer un certain nombre de conclusions.

Tout d'abord, les différents patch tests réalisés ont permis de montrer les points suivants :

- Dans le cas d'un arrangement nodal irrégulier, la précision augmente avec le nombre de nœuds.

- L'augmentation du nombre de points d'intégration améliore également la précision, bien qu'il existe un seuil au-delà duquel les résultats se dégradent.

- L'utilisation d'un support rectangulaire permet suivant le type de problème, d'améliorer les performances numériques des différentes méthodes. Néanmoins, la technique d'interpolation sur tout le domaine (modification des fonctions de pondération) conduit à des erreurs importantes sur les dérivées lorsque l'arrangement nodal est très irrégulier.

- Les méthodes meshfree sont d'une manière générale, très sensibles à la taille du domaine d'influence.

Dans l'ensemble, les méthodes qui nous ont semblé être les plus performantes sont les méthodes EFGM et RKPM pour les méthodes *MLS* et *RKM*, et la méthode RPIM pour les méthodes mixtes polynomiales - *RBF*.

Enfin, ce chapitre nous a permis de montrer dans le cas bidimensionnel, que la technique innovante consistant à coupler les méthodes EFGM ou RKPM avec des éléments finis 1D au niveau des frontières avec conditions aux limites de Dirichlet, produit généralement de meilleurs résultats que la technique classique utilisant une rangée d'éléments finis 2D.

Nous allons à présent développer au niveau du chapitre suivant, le formalisme mathématique et la formulation variationnelle que nous utilisons dans le logiciel *MoveFree*, pour simuler des problèmes géomécaniques statiques ou dynamiques dans les milieux poreux saturés en transformations finies.

# 3. Modèle Dynamique pour les Milieux Poreux Saturés en Transformations Finies

## 3.1 INTRODUCTION

Ayant consacré les deux premiers chapitres de ce travail à présenter un certain nombre de méthodes meshfree, tant au niveau du formalisme théorique que de la mise en œuvre pratique, nous allons dans ce chapitre, présenter les formulations mathématique et variationnelle applicables aux sols saturés en transformations finies, soumis à des sollicitations statiques ou dynamiques. Quelques applications numériques seront présentées au niveau du chapitre 4.

La suite de ce chapitre est organisée en trois sections. La section 3.2 rappelle la formulation mathématique du modèle dynamique applicable aux milieux poreux saturés en transformations finies. Pour ce faire, les équations de conservation sont tout d'abord établies en description eulérienne. Le système obtenu sur la configuration actuelle est ensuite transformé pour revenir à une formulation sur la configuration initiale. Nous avons en effet fait le choix de travailler en description lagrangienne totale, pour des raisons de simplification au niveau de la mise en œuvre numérique pour les méthodes meshfree (*cf*. le chapitre 4).

La section 3.3 rappelle les principales approches théoriques utilisées pour les modèles de comportement élastoplastiques en transformations finies, à savoir l'approche basée sur l'hypoélasticité et celle reposant sur l'hyperélasticité. La première approche est sans doute la plus utilisée en pratique en Géomécanique, du fait qu'elle s'apparente à la théorie de l'élastoplasticité en transformations infinitésimales.

Les sections 3.4 à 3.6 présentent les différents aspects de la modélisation numérique, à savoir la formulation variationnelle en description lagrangienne totale, les discrétisations en espace et en temps du système obtenu, et le schéma de résolution non linéaire adopté (schéma de Newton-Raphson).

## 3.2 FORMULATION MATHEMATIQUE DU MODELE DYNAMIQUE EN TRANSFORMATIONS FINIES

### 3.2.1 Hypothèses générales

Suivant la démarche adoptée par Biot (1941, 1955, 1977), le milieu poreux est modélisé à l'échelle macroscopique par un milieu continu solide et déformable, à travers lequel peut circuler un fluide interstitiel. A cette échelle, le milieu peut être représenté comme la superposition dans l'espace et le temps d'une phase solide et d'une phase fluide (dans notre cas, de l'eau).

Tout au long de nos travaux, nous ferons les hypothèses classiques en mécanique des sols, à savoir :

- La phase solide est homogène, isotrope et incompressible.
- Le fluide interstitiel est de type visqueux newtonien (en général de l'eau).
- La cinématique de la phase solide est privilégiée pour décrire le mouvement du milieu poreux global.
- L'espace poreux connecté est totalement saturé par le fluide.
- La porosité incluse du milieu est négligée.
- Les constituants des phases solide et fluide sont chimiquement inertes et ne peuvent subir de changement de phase.
- Les transformations sont isothermes.

L'étude est réalisée sur l'intervalle de temps $[0,T]$. Par ailleurs, le milieu poreux occupe le domaine $\Omega = \Omega_{t'}$ de frontière $\Gamma$ dans la configuration actuelle (instant $t'$ quelconque) et le domaine $\Omega_t$ de frontière $\Gamma_t$ dans la configuration de référence ($t \le t'$).

Dans les sections suivantes, nous allons tout d'abord établir les équations générales du problème en description eulérienne, que nous transformerons pour obtenir la formulation lagrangienne totale correspondante, cette dernière nous servant ensuite de base pour établir la formulation variationnelle du problème.

### 3.2.2 Equations de conservation en description eulérienne

#### *3.2.2.1 Équation de conservation de la quantité de mouvement*

Dans la configuration actuelle, les efforts extérieurs exercés sur le milieu poreux sont :

- les forces de volume à distance, qui dans notre cas seront uniquement les forces liées à l'accélération de la pesanteur et représentées par le vecteur constant $\underline{g}$.

- les forces de contact exercées à la frontière $\Gamma$ de $\Omega$, de densité surfacique $\underline{t}\left(\underline{x},t',\underline{n}\right)$, où $\underline{n}$ représente la normale extérieure à $\Gamma$ au point $\underline{x}$ considéré. Ces efforts s'exercent sur toutes les particules (solide et fluides) contenues dans $\Omega$. L'existence d'un tenseur symétrique $\underline{\underline{\sigma}}$ d'ordre 2 (tenseur des contraintes totales de Cauchy) vérifiant : $\underline{t}\left(\underline{x},t',\underline{n}\right)=\underline{\underline{\sigma}}\left(\underline{x},t'\right)\cdot\underline{n}$.

Par ailleurs, l'hypothèse de continuité macroscopique matérielle et le principe des actions mutuelles impliquent que les forces d'interactions (efforts de contact) entre les différentes phases s'annulent. En effet, ce sont des efforts extérieurs pour chaque phase, mais intérieurs au volume poreux.

La conservation de la quantité de mouvement pour le milieu poreux global se traduit par l'égalité entre la résultante des efforts extérieurs et des forces d'inertie d'une part, et l'égalité entre les moments correspondants d'autre part. Sur la configuration actuelle, elle s'écrit donc :

$$\int_{\Omega}\left(\rho^{sat}\dot{\underline{v}}^{s}+\rho^{w}\theta^{w}\dot{\underline{v}}^{rw}\right)d\Omega=\int_{\Omega}\rho^{sat}\underline{g}\,d\Omega+\int_{\Gamma}\underline{\underline{\sigma}}\cdot\underline{n}\,d\Gamma \tag{3.1}$$

Nous adopterons ici la formulation simplifiée du modèle macroscopique de Biot [ZIE 80], dans laquelle l'accélération relative de la phase fluide est négligée par rapport à l'accélération du squelette lorsque l'on est à basses fréquences (ce qui est généralement vérifié pour les sollicitations sismiques). Ceci s'exprime également par : $\dot{\underline{v}}^{w}\approx\dot{\underline{v}}^{s}$.

On peut finalement réécrire (3.1) sur $\Omega$ comme suit :

$$\int_{\Omega}\left(-\underline{div\underline{\underline{\sigma}}}+\rho^{sat}\dot{\underline{v}}^{s}-\rho^{sat}\underline{g}\right)d\Omega=\underline{0} \tag{3.2}$$

### *3.2.2.2 Équation de conservation de la masse*

En l'absence de source volumique de masse (système fermé), la masse de la phase $\alpha$ demeure constante au cours du temps. La différence entre la masse de la phase $\alpha$ contenue à l'instant $t'$ dans le volume élémentaire $\delta\Omega$ quelconque et celle contenue dans le volume élémentaire $\delta\Omega_t$, doit donc être nulle, ce qui se traduit par :

$$\left({}_{t}J\,{}_{t}J^{\alpha}\rho^{\alpha}-{}^{t}\rho^{\alpha}\right)\delta\Omega_{t}=0 \tag{3.3}$$

où ${}_{t}J^{\alpha}=\theta^{\alpha}/{}^{t}\theta^{\alpha}$ et ${}_{t}J$ représente le jacobien de la transformation du milieu poreux entre les configurations de référence $\Omega_t$ et actuelle $\Omega=\Omega_{t'}$. La relation (3.3) peut également s'écrire sous la forme :

$$\int_{\Omega}d_{t}^{\alpha}\left({}_{t}J\theta^{\alpha}\rho^{\alpha}\right)d\Omega=0 \tag{3.4}$$

Après développement de la dérivée particulaire, on obtient :

$$\int_{\Omega}\left(\frac{\partial_t\theta^{\alpha}+\underline{v}^{\alpha}\cdot\underline{\underline{grad}\,\theta^{\alpha}}}{\theta^{\alpha}}+\frac{\partial_t\rho^{\alpha}+\underline{v}^{\alpha}\cdot\underline{\underline{grad}\,\rho^{\alpha}}}{\rho^{\alpha}}+div\,\underline{v}^{\alpha}\right)d\Omega=0 \tag{3.5}$$

La phase solide étant supposée homogène et incompressible et en supposant la porosité uniforme dans le volume considéré, la conservation de la masse solide obtenue à partir de (3.5), s'écrit :

$$\partial_t\theta^{s}=-\partial_t n=(1-n)div\underline{v}^{s} \tag{3.6}$$

Si la masse volumique du fluide est supposée uniforme dans le volume considéré, la conservation de la masse fluide s'écrit d'après (3.5) :

$$\int_{\Omega}\left(\frac{\partial_t n}{n}+\frac{\partial_t\rho^{w}}{\rho^{w}}+div\underline{v}^{w}\right)d\Omega=0 \tag{3.7}$$

En utilisant (3.6) pour éliminer les termes $\partial_t n$, dans (3.7), on obtient :

$$\int_{\Omega}\left(\frac{\partial_t\rho^{w}}{\rho^{w}}+div\underline{v}^{rw}+\frac{1}{n}div\underline{v}^{s}\right)d\Omega=0 \tag{3.8}$$

Par ailleurs, on montre que :

$$\frac{\partial_t\rho^{w}}{\rho^{w}}=\beta^{w}\partial_t p \tag{3.9}$$

où $p=p\left(\underline{x},t'\right)$ représente la pression du fluide interstitielle à l'échelle macroscopique (valeurs positives).

Finalement, on obtient l'équation de continuité du milieu poreux sur la configuration actuelle :

$$\int_{\Omega}\left(n\beta^{w}\partial_t p+n\,div\underline{v}^{rw}+div\underline{v}^{s}\right)d\Omega=0 \tag{3.10}$$

Si l'on adopte la loi de Darcy généralisée pour décrire l'écoulement de la phase fluide à travers le milieu poreux, la vitesse relative du fluide par rapport à la phase solide s'écrira dans la configuration actuelle :

$$n\underline{v}^{rw}=\frac{k^{w}}{\mu^{w}}\left[-\underline{\underline{grad}p}+\rho^{w}\left(\underline{g}-\underline{\dot{v}}^{w}\right)\right] \tag{3.11}$$

Plusieurs auteurs ont montré que le terme inertiel lié à la phase fluide apparaissant dans (3.11) pouvait être négligé [ZIE 84 ; MOD 87], même si dans ce cas, la loi de Darcy n'est plus généralisée.

*3.2.2.3 Système d'équations final en description eulérienne*

En utilisant le principe des contraintes effectives de Terzaghi, que nous écrirons sous la forme : $\underline{\underline{\sigma}}' = \underline{\underline{\sigma}} + p\,\underline{\underline{1}}$, alors le système à résoudre dans la configuration actuelle, peut s'écrire :

$$\int_{\Omega} \left( \rho^{sat} \dot{\underline{v}}^s - \underline{div\underline{\underline{\sigma}}'} + \underline{gradp} - \rho^{sat}\,\underline{g} \right) d\Omega = \underline{0} \tag{3.12}$$

$$\int_{\Omega} \left[ div\underline{v}^s + n\beta^w \partial_t p - div\left( \frac{k^w}{\mu^w} \left[ \underline{gradp} - \rho^w \underline{g} \right] \right) \right] d\Omega = 0 \tag{3.13}$$

où l'on a considéré $\underline{v}^s = \underline{v}^s\left(\underline{x}, t'\right)$ et $p$ comme inconnues principales. Ces équations doivent être complétées par celle de la loi de comportement.

Remarque :

Par abus de langage en modélisation des milieux poreux, il est usuel de nommer la première équation, « équation mécanique » et la seconde, « équation hydraulique ».

### 3.2.3 Equations de conservation en description lagrangienne

En partant des équations établies sur la configuration actuelle, il est possible d'établir une formulation du problème sur une autre configuration prise comme référence [KIO 85 ; VOY 97]. La configuration de référence choisie pour établir la formulation est dans ce cas la configuration initiale pour une description lagrangienne totale (description *TL*), ou la dernière configuration connue pour une description lagrangienne réactualisée (description *UL*). Nous verrons ultérieurement ce qu'implique l'une ou l'autre de ces descriptions sur la mise en œuvre numérique.

Afin d'écrire les équations (3.12) et (3.13) sur la configuration de référence, nous utiliserons la quantité ${}_t p = {}^{t'}p - {}^{t}p$, différence des valeurs de pression interstitielle mesurées pour des points homologues entre les configurations $\Omega_t$ prise comme référence et $\Omega = \Omega_{t'}$ (actuelle).

En utilisant le théorème de Gauss et la définition du premier tenseur des contraintes de Piola-Kirchhoff pour la phase solide, noté ${}_t\underline{\underline{\Pi}}' = {}^{t'}_{t}\underline{\underline{\Pi}}'$, on montre que le terme $\int_{\Omega} \underline{div\underline{\underline{\sigma}}'}\, d\Omega$ dans l'équation (3.12) peut s'écrire sur la configuration de référence sous la forme suivante :

$$\int_{\Omega} \underline{div\underline{\underline{\sigma}}'}\, d\Omega = \int_{\Omega_t} {}_t\underline{div_t\underline{\underline{\Pi}}'}\, d\Omega_t \tag{3.14}$$

Afin d'écrire l'équation de continuité (3.13) sur la configuration $\Omega_t$, nous utilisons le fait que physiquement, le terme $div\underline{v}^s$ exprime la variation de volume de la phase solide :

$$div\underline{v}^s = tr\left(\underline{\underline{d}}\right) \tag{3.15}$$

D'après la définition du tenseur taux de déformations $\underline{\underline{d}}$ en fonction du tenseur gradient de vitesse $\underline{\underline{l}} = {}_t\dot{\underline{\underline{F}}} \cdot {}_t\underline{\underline{F}}^{-1}$, on peut également écrire la relation (3.15) comme suit :

$$tr\left(\underline{\underline{d}}\right) = tr\left({}_t\dot{\underline{\underline{F}}} \cdot {}_t\underline{\underline{F}}^{-1}\right) = {}_t\dot{J} \big/ {}_tJ \tag{3.16}$$

où ${}_t\dot{J}$ représente la dérivée particulaire de ${}_tJ$ en suivant la phase solide.

Finalement, si ${}_t\underline{u}^s$ représente le vecteur de déplacements de la phase solide entre les instants $t'$ et $t$, alors en prenant ${}_t\underline{u}^s$ et ${}_tp$ comme inconnues principales, le système complet à résoudre dans la configuration de référence peut s'écrire :

$$\int_{\Omega_t} \left( - {}_t\underline{div}\, {}_t\underline{\underline{\Pi}}' + {}_tJ\, {}_t\underline{\underline{F}}^{-T} \left( {}_t\underline{u}^s \right) \cdot {}_t\underline{grad}\left( {}_tp \right) + {}^t\rho^{sat}\, {}_t\ddot{\underline{u}}^s - {}^t\rho^{sat}\, \underline{g} \right) d\Omega_t = \underline{0} \tag{3.17}$$

$$\begin{aligned} \int_{\Omega_t} \Big[ & {}_t\dot{J} \big/ {}_tJ + {}_t\hat{n}\, \beta^w\, \partial_t \left( {}_tp \right) + \hat{K}\, div\left( \rho^w \underline{g} \right) \\ & - \hat{K}\, {}_t div\left( {}_t\underline{\underline{C}}^{-1} \cdot {}_t\underline{grad}\left( {}_tp \right) \right) \Big] d\Omega_t = 0 \end{aligned} \tag{3.18}$$

en posant : $\hat{K} = \dfrac{k^w}{\mu^w}$ (constante) et ${}_t\hat{n} = 1 - \left(1 - {}^tn\right) \big/ {}_tJ$.

Dorénavant, nous adopterons la description lagrangienne pour la résolution du problème aux limites.

### 3.2.4 Conditions initiales et aux limites

Dans le cas d'une approche lagrangienne, les conditions initiales seront définies sur la configuration initiale $\overline{\Omega}_0$ et les conditions aux limites, sur la configuration de référence.

#### *3.2.4.1 Conditions initiales*

Au niveau de l'équation mécanique, on définira des conditions initiales en déplacements et vitesses (ou accélérations) du solide et en contraintes totales sur $\overline{\Omega}_0$ par :

$$\underline{u}\left(\underline{X}, 0\right) = \underline{u}_0\left(\underline{X}\right) \tag{3.19}$$

$$\underline{v}\left(\underline{X}, 0\right) = \underline{v}_0\left(\underline{X}\right) \tag{3.20}$$

$$\underline{\underline{\sigma}}\left(\underline{x}, 0\right) = \underline{\underline{\Pi}}\left(\underline{X}, 0\right) = \underline{\underline{\sigma}}_0\left(\underline{X}\right) \tag{3.21}$$

Au niveau de l'équation hydraulique, on donnera des conditions initiales en pressions et en flux :

$$p(\underline{X},0) = p_0(\underline{X}) \tag{3.22}$$

$$\Phi(\underline{X},0) = \Phi_0(\underline{X}) \tag{3.23}$$

*3.2.4.2 Conditions aux limites*

D'une manière générale, les conditions aux limites mécaniques et hydrauliques, fournies sur la configuration de référence $\bar{\Omega}_t$, se divisent en deux classes :

- les conditions de Dirichlet, également appelées conditions aux limites essentielles, sur les déplacements ou les pressions de fluide :

$$_t\underline{u} = \underline{\bar{u}} \qquad \text{sur } \Gamma_{tu} \times ]0,T] \tag{3.24}$$

$$^t p = \bar{p} \qquad \text{sur } \Gamma_{tp} \times ]0,T] \tag{3.25}$$

- les conditions de Neumann, également appelées conditions aux limites naturelles, sur les contraintes totales ou les flux de fluide :

$$_t\underline{\underline{\Pi}} \cdot \underline{N} = \underline{\bar{T}} \qquad \text{sur } \Gamma_{t\sigma} \times ]0,T] \tag{3.26}$$

$$_t\Phi = \bar{\Phi} \qquad \text{sur } \Gamma_{t\varphi} \times ]0,T] \tag{3.27}$$

Par ailleurs, les partitions de la frontière de Dirichlet doivent vérifier les relations suivantes :

$$\Gamma_t = \Gamma_{tu} \cup \Gamma_{t\sigma} = \Gamma_{tp} \cup \Gamma_{t\varphi} \text{ et } \varnothing = \Gamma_{tu} \cap \Gamma_{t\sigma} = \Gamma_{tp} \cap \Gamma_{t\varphi} \tag{3.28}$$

Dans notre cas, on prendra : $\bar{\Omega}_t = \bar{\Omega}_0$.

## 3.3 LOI DE COMPORTEMENT SOLIDE EN TRANSFORMATIONS FINIES

### 3.3.1 Introduction

Quel que soit le type de transformations envisagées, infinitésimales ou finies, l'écriture d'un modèle de comportement nécessite de respecter le principe d'objectivité ou d'indifférence matérielle, c'est-à-dire que la loi de comportement est invariante dans tout changement de référentiel. Il est également nécessaire de choisir de façon appropriée, les mesures de contraintes et de déformations à utiliser.

Par ailleurs, le comportement rhéologique d'un milieu poreux saturé d'eau étant en fait régi par le modèle constitutif de la phase solide, nous allons présenter dans les sections suivantes, les deux types d'approches classiquement utilisées en transformations finies, pour obtenir une écriture matériellement objective de la loi de comportement pour un matériau solide.

### 3.3.2 Décomposition multiplicative du gradient de la transformation

La décomposition multiplicative du gradient de la transformation fut suggérée par Lee [LEE 69] et Mandel [MAN 72], puis reprise par la plupart des auteurs (notamment [NEM 79 ; NEM 82]). Elle consiste à écrire le gradient $\underline{\underline{F}}$ sous la forme :

$$\underline{\underline{F}} = \underline{\underline{F}}^e \cdot \underline{\underline{F}}^p \tag{3.29}$$

où $\underline{\underline{F}}^e$ et $\underline{\underline{F}}^p$ représentent les gradients de transformations respectivement élastique et plastique.

Remarque : Pour simplifier les notations, nous avons omis l'indice gauche « $t$ » pour la configuration de référence dans (3.29) et nous l'omettrons également dans la suite, lorsque cela est possible.

La décomposition (3.29) repose sur l'existence d'une *configuration intermédiaire relâchée* $\Omega_p$. Cette configuration représente un état libre de toutes contraintes, obtenu en déchargeant le matériau à partir de la configuration actuelle ($\Omega$) et dans lequel aucun processus inélastique ne peut se produire au cours de la déformation. Le tenseur $\underline{\underline{F}}^p$ correspond alors au gradient de la transformation entre les configurations $\Omega_t$ (référence) et $\Omega_p$ (*cf.* Fig. 3.1). Nous noterons dorénavant $\overline{A}$, toute quantité $A$ exprimée sur la configuration relâchée.

Comme conséquence, le gradient de vitesse $\underline{\underline{l}}$ peut être décomposé en deux parties :

$$\underline{\underline{l}} = \dot{\underline{\underline{F}}} \cdot \underline{\underline{F}}^{-1} = \underline{\underline{l}}^e + \underline{\underline{F}}^e \cdot \overline{\underline{\underline{l}}}^p \cdot \underline{\underline{F}}^{e-1} \tag{3.30}$$

en posant :

$$\underline{\underline{l}}^e = \dot{\underline{\underline{F}}}^e \cdot \underline{\underline{F}}^{e-1} \tag{3.31}$$

$$\underline{\underline{\bar{l}}}^{p} = \underline{\underline{\dot{F}}}^{p} \cdot \underline{\underline{F}}^{p-1} \tag{3.32}$$

Il est alors possible d'écrire le taux eulérien des déformations sous la forme :

$$\underline{\underline{d}} = \underline{\underline{d}}^{e} + \underline{\underline{d}}^{p} \tag{3.33}$$

en posant :

$$\underline{\underline{d}}^{e} = \left(\underline{\underline{l}}^{e}\right)^{S} \tag{3.34}$$

$$\underline{\underline{d}}^{p} = \left(\underline{\underline{F}}^{e} \cdot \underline{\underline{\bar{l}}}^{p} \cdot \underline{\underline{F}}^{e-1}\right)^{S} \tag{3.35}$$

Le choix d'une configuration relâchée, ainsi que celui des mesures de déformations et de contraintes appropriées, est primordial pour obtenir une écriture objective de la loi de comportement. Si l'on choisit des mesures invariantes dans tout mouvement de corps rigide, alors on obtient une écriture vérifiant naturellement le principe d'indifférence matérielle [SID 84].

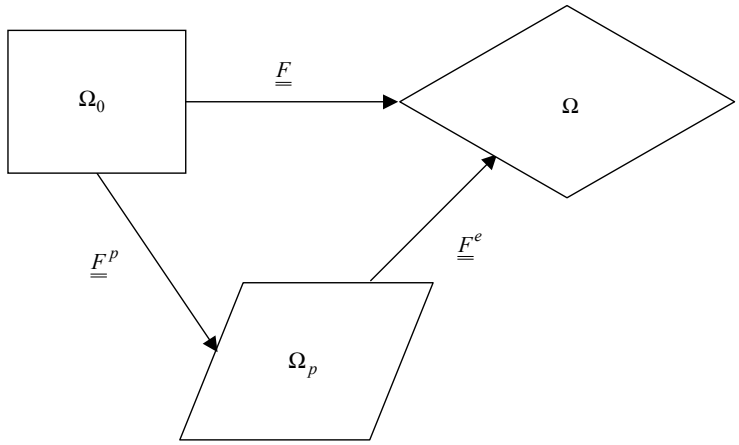



***Fig. 3.1 : Cinématique des déformations élastoplastiques en transformations finies.***

## 3.3.3 Matériaux hypoélastiques

### *3.3.3.1 Principe*

Pour les matériaux hypoélastiques, la loi de comportement est écrite de manière incrémentale, c'est-à-dire qu'elle relie les incréments de contraintes aux incréments de déformations. Cette approche est classique pour la plupart des outils numériques mettant en œuvre des modèles élastoplastiques en transformations infinitésimales. En effet, ces outils adoptent généralement comme inconnues principales, les incréments de déplacements entre deux configurations. La loi de comportement non linéaire est de ce fait intégrée de manière incrémentale, suivant un processus itératif.

Cette approche fut naturellement transposée aux transformations finies, du fait qu'elle nécessite peu de modifications au niveau des outils numériques existants. Elle est utilisée dans les domaines de la Géomécanique [CHO 92 ; MER 95], et plus généralement en Mécanique des Milieux Continus [GRE 71 ; MCK 75 ; DIE 79 ; HUG 80a ; HUG 80b ; DAF 83 ; KIO 85 ; VOY 89 ; STO 92 ; RAS 96 ; ROD 97 ; ROD 98].

La mise en œuvre de la loi incrémentale est néanmoins différente, suivant que l'on adopte une formulation en description *UL* ou *TL*.

En description *TL*, la loi relie directement l'incrément des contraintes de Piola-Kirchhoff à celui des déformations de Green-Lagrange, tandis qu'en description *UL*, on préfère généralement relier le taux de contraintes de Cauchy au taux eulérien des déformations.

En description *UL*, il est néanmoins possible d'adopter une écriture de la loi avec ou sans dérivée objective [BAT 75], c'est-à-dire avec une dérivée calculée par rapport à un référentiel lié à la matière. Cependant, l'utilisation d'une dérivée objective n'est possible que pour des matériaux isotropes, du fait que le tenseur du $4^{ème}$ ordre reliant le taux de contraintes au taux de déformations est alors nécessairement isotrope [SID 84 ; MAU 99].

Quelles que soient la description et l'option choisies, l'écriture incrémentale de la loi doit nécessairement vérifier le principe d'*objectivité incrémentale*, c'est-à-dire qu'elle doit assurer un état de contraintes cohérent avec tout mouvement de corps rigide. Par exemple, une rotation pure ne peut induire de déformation pure et *vice versa*. Cette notion a été introduite au départ par Hughes et Winget [HUG 80b], puis largement mise en œuvre par un certain nombre d'auteurs (voir notamment [PIN 83 ; ROD 97 ; ROD 98]).

Ultérieurement, Rashid [RAS 96] a complété le principe d'objectivité incrémentale de Hughes et Winget, en introduisant une condition supplémentaire : pour deux incréments de déformations identiques à une rotation de corps rigide près, le résultat doit également différer d'une rotation. Il appelle ce principe *objectivité incrémentale forte*, par opposition à celle de Hughes et Winget qu'il caractérise de *faible*.

### *3.3.3.2 Dérivées objectives usuelles*

Le choix de la dérivée objective est important pour la qualité des résultats obtenus. En effet, certains auteurs [DAF 83 ; SID 84 ; VOY 89] ont montré par exemple, que pour un problème de cisaillement simple, la dérivée de Jaumann conduisait à des oscillations au niveau des contraintes calculées et qu'il était préférable d'utiliser la dérivée de Green-Naghdi pour les éliminer.

En transformations finies, la dérivée matérielle $\dot{\underline{\underline{\sigma}}}'$ des contraintes effectives de Cauchy dépend de la réponse matérielle $\dot{\underline{\underline{\sigma}}}'^{D}$ du milieu (dérivée objective) et de celle en rotation, notée $\underline{\underline{\sigma}}'^{R}$ :

$$\dot{\underline{\underline{\sigma}}}' = \dot{\underline{\underline{\sigma}}}'^{D} + \underline{\underline{\sigma}}'^{R} \tag{3.36}$$

La réponse matérielle est déterminée à partir de la loi de comportement incrémentale (*cf.* §3.3.3.3), tandis que la réponse en rotation varie suivant le type de transport choisi pour exprimer la dérivée objective.

Nous noterons $\underline{\underline{\sigma}}'^{R}$ sous la forme condensée suivante :

$$\underline{\underline{\sigma}}'^{R} = \underline{\underline{a}} \cdot \underline{\underline{\sigma}}' + \underline{\underline{\sigma}}' \cdot \underline{\underline{a}}'^{T} \tag{3.37}$$

Nous fournissons ci-après les expressions pour les tenseurs $\underline{\underline{a}}$ et $\underline{\underline{a}}'$, en fonction de la dérivée objective considérée (se référer au chapitre des notations pour tout symbole non précisé ici).

Transport par rotation :

***Dérivée corotationnelle (Jaumann) :***

$$\underline{\underline{a}} = \underline{\underline{a}}' = \underline{\underline{\omega}} \tag{3.38}$$

***Dérivée de Green-Naghdi :***

$$\underline{\underline{a}} = \underline{\underline{a}}' = \underline{\underline{\Omega}} \tag{3.39}$$

Transport convectif contravariant :

***Dérivée de Lie ou Oldroyd :***

$$\underline{\underline{a}} = \underline{\underline{a}}' = \underline{\underline{l}} \tag{3.40}$$

Transport convectif covariant :

***Dérivée intrinsèque (Truesdell) :***

$$\underline{\underline{a}} = \underline{\underline{l}} - tr\left(\underline{\underline{d}}\right)\underline{\underline{1}} \tag{3.41}$$

$$\underline{\underline{a}}' = \underline{\underline{l}} \tag{3.42}$$

Remarques :

- Dans le cas d'une rotation de corps rigide, les dérivées de Green-Naghdi et Truesdell sont équivalentes à la dérivée de Jaumann.

- Une critique vis-à-vis d'un taux de type Truesdell est qu'il ne s'annule pas nécessairement pour une contrainte de Cauchy constante [STO 92].

*3.3.3.3 Ecritures incrémentales en description UL*

Pour être valide, l'écriture incrémentale en description *UL* suppose que les déformations élastiques demeurent infinitésimales au cours de la transformation, bien que les rotations puissent être importantes. Cette approche suppose également que l'histoire du matériau est exprimée en fonction des contraintes effectives de Cauchy et du taux eulérien des déformations.

On choisit donc une configuration relâchée pour laquelle $\underline{\underline{F}}^e$ correspond aux déformations pures : $\underline{\underline{F}}^e = \underline{\underline{V}}^e$, c'est-à-dire que $\underline{\underline{F}}^e$ est symétrique et que la totalité de la rotation est associée au tenseur des déformations plastiques $\underline{\underline{F}}^p$. Dans ce cas, l'hypothèse des déformations élastiques infinitésimales adoptée par la majorité des auteurs, c'est-à-dire $\underline{\underline{V}}^e \approx \underline{\underline{1}}$, permet d'exprimer le taux de déformation plastique sur la configuration relâchée à partir de la relation (3.35) :

$$\underline{\underline{d}}^p \approx \overline{\underline{\underline{d}}}^p = \left(\overline{\underline{\underline{l}}}^p\right)^S \tag{3.43}$$

Si l'on fait l'hypothèse que les déformations restent faibles au cours de chaque pas de temps $\Delta t$ et en notant $\Omega = \Omega_{t+\Delta t}$, la configuration actuelle, alors on peut simplifier la dérivée du tenseur des déformations de Green-Lagrange en écrivant :

$${}_t\dot{\underline{\underline{E}}} = {}_t\underline{\underline{F}}^T \cdot \underline{\underline{d}} \cdot {}_t\underline{\underline{F}} \approx \underline{\underline{d}} \tag{3.44}$$

Ce qui sous forme incrémentale devient :

$${}_t\Delta\underline{\underline{E}} \approx \Delta t\, \underline{\underline{d}} \tag{3.45}$$

où ${}_t\Delta\underline{\underline{E}}$ représente l'incrément de déformations de Green-Lagrange entre les instants $t$ et $t+\Delta t$ (mesure par rapport à la configuration $\Omega_t$).

Par ailleurs, on décompose ${}_t\Delta\underline{\underline{E}}$ en une partie « linéaire » ${}_t\Delta\underline{\underline{\varepsilon}}$, et une partie « non linéaire » ${}_t\Delta\underline{\underline{\eta}}$ :

$${}_t\Delta\underline{\underline{E}} = {}_t\Delta\underline{\underline{\varepsilon}} + {}_t\Delta\underline{\underline{\eta}} \tag{3.46}$$

en posant :

$${}_t\Delta\varepsilon_{ij} = \frac{1}{2}\left({}_t\Delta u_{i,j} + {}_t\Delta u_{j,i}\right) \tag{3.47}$$

$$ {}_t\Delta\eta_{ij} = \frac{1}{2}\,{}_t\Delta u_{k,i} \cdot {}_t\Delta u_{k,j} \tag{3.48} $$

Alors, d'après (3.45) et en utilisant les définitions de $\underline{\underline{d}}$ et ${}_t\Delta\underline{\underline{\varepsilon}}$ (relation (3.47)), ${}_t\Delta\underline{\underline{E}}$ devient simplement :

$$ {}_t\Delta\underline{\underline{E}} \approx {}_t\Delta\underline{\underline{\varepsilon}} \tag{3.49} $$

c'est-à-dire également :

$$ {}_t\Delta\underline{\underline{E}} \approx {}_t\Delta\underline{\underline{\varepsilon}}^e + {}_t\Delta\underline{\underline{\varepsilon}}^p \tag{3.50} $$

Finalement, on voit que l'écriture (3.50) en description *UL* est identique à celle utilisée pour les modèles de comportement élastoplastiques en transformations infinitésimales. La différence provient ensuite de l'utilisation ou non d'une dérivée objective pour exprimer les taux de contraintes.

***Ecriture avec dérivée objective :***

Dans le cas où l'on choisit d'utiliser une dérivée objective, le taux objectif ${}^{t+\Delta t}\underline{\underline{\dot{\sigma}}}'^{D}$, calculé au temps $t+\Delta t$, est soit déterminé à partir du taux ${}^{t+\Delta t}\underline{\underline{d}}^e$ et du tenseur élastique $\underline{\underline{\underline{\underline{D}}}}^e$ (constant) :

$$ {}^{t+\Delta t}\underline{\underline{\dot{\sigma}}}'^{D} = \underline{\underline{\underline{\underline{D}}}}^e : {}^{t+\Delta t}\underline{\underline{d}}^e \tag{3.51} $$

soit à partir du taux ${}^{t+\Delta t}\underline{\underline{d}}$ et du tenseur élastoplastique ${}_t\underline{\underline{\underline{\underline{D}}}}$ :

$$ {}^{t+\Delta t}\underline{\underline{\dot{\sigma}}}'^{D} = {}_t\underline{\underline{\underline{\underline{D}}}} : {}^{t+\Delta t}\underline{\underline{d}} \tag{3.52} $$

Le tenseur ${}_t\underline{\underline{\underline{\underline{D}}}}$ dépend de l'histoire des contraintes (${}^t\underline{\underline{\sigma}}'$) et des déformations (${}^t\underline{\underline{\varepsilon}}$) accumulées jusqu'à la configuration $\Omega_t$.

L'incrément de contraintes de Cauchy entre les instants $t$ et $t+\Delta t$ s'exprime alors en utilisant les relations (3.36), (3.45), (3.49) et (3.52) :

$$ {}_t\Delta\underline{\underline{\sigma}}' = {}_t\underline{\underline{\underline{\underline{D}}}} : {}_t\Delta\underline{\underline{\varepsilon}} + {}_t\Delta\underline{\underline{\sigma}}'^{R} \tag{3.53} $$

avec :

$$ {}_t\Delta\underline{\underline{\sigma}}'^{R} = {}_t\Delta\underline{\underline{a}} \cdot {}^t\underline{\underline{\sigma}}' + {}^t\underline{\underline{\sigma}}' \cdot {}_t\Delta\underline{\underline{a}}'^{T} \tag{3.54} $$

Par exemple, les composantes du tenseur $\underline{\underline{a}}$ pour la dérivée objective de Jaumann (3.38), seront de la forme suivante :

$$ {}_t\Delta a_{ij} = {}_t\Delta\omega_{ij} = \frac{1}{2}\left( {}_t\Delta u_{i,j} - {}_t\Delta u_{j,i} \right) \tag{3.55} $$

***Ecriture sans dérivée objective :***

Une écriture incrémentale sans dérivée objective peut être obtenue à partir de la relation (3.49), en déterminant l'incrément de contraintes de Piola-Kirchhoff ${}_t\Delta\underline{\underline{S}}'$, en fonction de ${}_t\Delta\underline{\underline{\varepsilon}}$ [HIB 70 ; BAT 75 ; KIO 88] :

$$ {}_t\Delta\underline{\underline{S}}' = \underset{t}{\underline{\underline{\underline{\underline{D}}}}} : {}_t\Delta\underline{\underline{\varepsilon}} \tag{3.56} $$

où $\underset{t}{\underline{\underline{\underline{\underline{D}}}}}$ dépend également de ${}^t\underline{\underline{\sigma}}'$ et ${}^t\underline{\underline{\varepsilon}}$.

Les contraintes effectives calculées au temps $t+\Delta t$ par rapport à la configuration $\Omega_t$, s'écrivent alors :

$$ {}^{t+\Delta t}_{t}\underline{\underline{S}}' = {}^t\underline{\underline{\sigma}}' + {}_t\Delta\underline{\underline{S}}' \tag{3.57} $$

On calcule la contrainte effective de Cauchy sur la configuration $\Omega_{t+\Delta t}$, en utilisant la formule de transport suivante (« push-forward ») :

$$ {}^{t+\Delta t}\underline{\underline{\sigma}}' = \frac{1}{{}^{t+\Delta t}_{t}J} {}^{t+\Delta t}_{t}\underline{\underline{F}} \cdot {}^{t+\Delta t}_{t}\underline{\underline{S}}' \cdot {}^{t+\Delta t}_{t}\underline{\underline{F}}^T \tag{3.58} $$

Le principal inconvénient des approches que nous venons de présenter, est que l'intégration en temps des différents taux calculés au niveau du modèle de comportement conduit à des erreurs numériques parfois importantes, bien que le principe d'objectivité soit respecté [HUG 80b ; NAG 81 ; RAS 96].

### *3.3.3.4 Ecriture incrémentale en description TL*

Une autre approche consiste à formuler la loi de comportement en description *TL*, et donc d'exprimer l'incrément de contraintes de Piola-Kirchhoff ${}_0\Delta\underline{\underline{S}}'$ entre les instants $t$ et $t+\Delta t$, en fonction de l'incrément de déformations de Green-Lagrange ${}_0\Delta\underline{\underline{E}}$ (mesures par rapport à la configuration initiale) :

$$ {}_0\Delta\underline{\underline{S}}' = \underset{0}{\underline{\underline{\underline{\underline{D}}}}} : {}_0\Delta\underline{\underline{E}} \tag{3.59} $$

Dans ce cas, on aura :

$$ {}_0\Delta\underline{\underline{E}} = {}_0\Delta\underline{\underline{\varepsilon}} + {}_0\Delta\underline{\underline{\gamma}} + {}_0\Delta\underline{\underline{\eta}} \tag{3.60} $$

en posant :

$$ {}_0\Delta\gamma_{ij} = \frac{1}{2}\left( {}^{t}_{0}u_{k,i}\ {}_0\Delta u_{k,j} + {}_0\Delta u_{k,i}\ {}^{t}_{0}u_{k,j} \right) \tag{3.61} $$

Par ailleurs, le tenseur $\underset{0}{\underline{\underline{\underline{\underline{D}}}}}$, mesuré dans la configuration initiale, est calculé de manière identique à celui des transformations infinitésimales, tout en dépendant à présent des contraintes de Piola-Kirchhoff ($\underset{0}{\overset{t}{\underline{\underline{S}}}}'$) et des déformations de Green-Lagrange ($\underset{0}{\overset{t}{\underline{\underline{E}}}}$) accumulées à l'instant *t* par rapport à la configuration initiale [BAT 75].

La contrainte effective de Piola-Kirchhoff exprimée sur la configuration $\Omega_{t+\Delta t}$ s'écrit alors simplement :

$$ \underset{0}{\overset{t+\Delta t}{\underline{\underline{S}}}}' = \underset{0}{\overset{t}{\underline{\underline{S}}}}' + {}_0\Delta\underline{\underline{S}}' \tag{3.62} $$

Sidoroff [SID 84] montre néanmoins que l'extension directe ($\dot{\underline{\underline{S}}}' = \mathcal{F}\left(\dot{\underline{\underline{E}}}\right)$) de la loi de comportement en transformations infinitésimales ($\dot{\underline{\underline{\sigma}}}' = \mathcal{F}\left(\dot{\underline{\underline{\varepsilon}}}\right)$), conduit dans certains cas à des résultats physiquement incohérents. C'est pourquoi l'approche présentée dans cette section ne devrait être utilisée que si la loi de comportement est réellement formulée en grandes transformations (notamment par des essais appropriés). Malheureusement, les modèles utilisés en Géomécanique, ont pour la plupart, été formulés en transformations infinitésimales.

## 3.3.4 Matériaux hyperélastiques

### *3.3.4.1 Introduction*

Contrairement aux matériaux hypoélastiques, la loi de comportement hyperélastique est écrite de manière directe, c'est-à-dire qu'elle relie les contraintes aux déformations sans utiliser les incréments de ces quantités. Cette approche est classique en Mécanique des Milieux Continus [NEM 79 ; NEM 82 ; DOG 84 ; SIM 85 ; SIM 88a ; SIM 88b ; SIM 92 ; ETE 90 ; HOF 90 ; CUI 92 ; EVE 94 ; PER 99a ; PER 99b ; MAU 99 ; MAS 00 ; ZAB 00], mais peu utilisée en Géomécanique [BOU 97 ; BOR 98].

Cette approche repose également sur la décomposition multiplicative du gradient de la transformation (3.29) et suppose qu'il existe une densité d'énergie libre régie soit par les déformations élastoplastiques, soit uniquement par les déformations élastiques.

Nous rappelons ci-après, l'approche basée sur la théorie classique de la Thermodynamique des Processus Irréversibles, utilisée pour établir la formulation générale d'un modèle de comportement représentatif des divers phénomènes réversibles et irréversibles se produisant au niveau de la phase solide. Pour ce faire, nous avons choisi d'utiliser la démarche présentée par Bourgeois [BOU 97], qui consiste à écrire les équations générales du comportement du milieu poreux en description *TL*. La seconde démarche possible consiste à se placer en description eulérienne [BOR 98].

### *3.3.4.2 Hypothèses générales*

Nous rappelons les hypothèses principales sur lesquelles reposent notre approche :

- Les transformations sont isothermes.
- Quelle que soit la configuration considérée, la métrique choisie est celle de l'espace euclidien $\mathbb{R}^3$.
- Le comportement élastique est isotrope.
- La partie élastique des déformations est linéaire et infinitésimale.
- Les propriétés élastiques et plastiques associées à la phase solide sont découplées : la densité volumique d'énergie libre $\psi$ (lagrangienne ou eulérienne suivant l'écriture choisie) peut donc s'écrire comme la somme d'un terme $\psi^e$, partie récupérable par décharge de l'élément de volume, et d'un terme $U$, appelé énergie bloquée, qui n'est pas restituée à la décharge et qui dépend éventuellement de variables internes $\alpha$, quantité (scalaire, vectorielle ou tensorielle) caractérisant l'écrouissage du matériau. L'expression de l'énergie libre, à savoir les variables utilisées, est différente suivant le type de formulation adoptée (*TL* ou *UL*). Nous l'expliciterons donc pour chaque écriture.

Du fait de l'hypothèse d'isotropie, on montre que le comportement élastique est uniquement fonction des déformations pures [SID 84] et que l'expression de l'énergie libre est indépendante du choix de la configuration relâchée [SIM 92 ; BOU 97].

Remarque :

Une hypothèse importante faite classiquement pour les matériaux hyperélastiques, est l'incompressibilité plastique du matériau, c'est-à-dire que la réponse volumique totale est élastique. Cette hypothèse est valide dans le cadre des critères de plasticité dépendant uniquement du second invariant des contraintes et ne faisant pas intervenir la contrainte moyenne (par exemple le critère de Von Mises). Ce type de plasticité, souvent appelée « *J2*-plasticité » [SIM 92], est notamment caractéristique des métaux. En revanche, le comportement des sols est généralement très dépendant de la contrainte moyenne, ce qui rend l'hypothèse d'incompressibilité plastique incorrecte. En effet, Hofstetter [HOF 90] montre que pour un critère de Drücker-Prager écrit avec une telle hypothèse, on peut aboutir dans certaines situations à une norme du déviateur des contraintes de Kirchhoff négative, ce qui n'a pas de signification physique. Dans nos travaux, nous ne ferons pas cette hypothèse.

### *3.3.4.3 Ecriture en description TL*

Nous allons décrire l'évolution du comportement en plaçant l'analyse sur la configuration relâchée, choisie de manière à ce que $\underline{\underline{F}}^e$ corresponde aux déformations pures (*cf.* §3.3.3) et soit de ce fait symétrique.

Nous définissons également les tenseurs suivants :

- Tenseurs « élastique » et « plastique » de Cauchy-Green droits :

$$\underline{\underline{C}}^e = \underline{\underline{F}}^{eT} \cdot \underline{\underline{F}}^e \tag{3.63}$$

$$\underline{\underline{C}}^p = \underline{\underline{F}}^{pT} \cdot \underline{\underline{F}}^p \tag{3.64}$$

- Tenseur « élastique » de Cauchy-Green gauche :

$$\underline{\underline{b}}^e = \underline{\underline{F}}^e \cdot \underline{\underline{F}}^{eT} = \underline{\underline{F}} \cdot \underline{\underline{C}}^{p-1} \cdot \underline{\underline{F}}^T \tag{3.65}$$

- Taux de déformations « élastiques » :

$$\underline{\underline{d}}^e = \underline{\underline{F}}^{eT} \cdot \underline{\underline{d}}^e \cdot \underline{\underline{F}}^e = \frac{1}{2}\dot{\underline{\underline{C}}}^e \tag{3.66}$$

- Tenseur des déformations « élastiques » principales logarithmiques (tenseur de Henky) :

$$\underline{\underline{e}} = \ln \underline{\underline{U}}^e = 1/2 \ln \underline{\underline{C}}^e \tag{3.67}$$

Par ailleurs, la décomposition multiplicative du gradient (3.29) nous permet d'écrire le jacobien *J* de la transformation sous la forme suivante :

$$J = J^e J^p = \det\left(\underline{\underline{F}}^e\right) \det\left(\underline{\underline{F}}^p\right) \tag{3.68}$$

L'hypothèse des déformations élastiques infinitésimales faite en §3.3.4.2, se traduit alors par les relations suivantes :

$$\left\| \underline{\underline{e}} \right\| \ll 1 \tag{3.69}$$

$$J^e \approx 1 \tag{3.70}$$

On admet enfin que la densité volumique lagrangienne d'énergie libre $\psi$ dépend uniquement des tenseurs $\underline{\underline{C}}^e$ et $\underline{\underline{C}}^p$ (*cf.* §3.3.4.2), ce qui permet d'écrire :

$$\psi\left(\underline{\underline{C}}^e, \underline{\underline{C}}^p\right) = \psi^e\left(\underline{\underline{C}}^e\right) + U\left(\underline{\underline{C}}^p\right) \tag{3.71}$$

***Evolution de la partie réversible de la transformation :***

On exprime tout d'abord le second tenseur de Piola-Kirchhoff $\overline{\underline{\underline{S}}}$ sur la configuration relâchée, en transportant le tenseur des contraintes effectives de Kirchhoff ( $\underline{\underline{\tau}}' = J\,\underline{\underline{\sigma}}'$ ), à partir de la configuration actuelle :

$$\overline{\underline{\underline{S}}} = \underline{\underline{F}}^{e-1} \cdot \underline{\underline{\tau}}' \cdot \underline{\underline{F}}^{e-T} \tag{3.72}$$

En utilisant les propriétés de dualité des contraintes, on exprime la réversibilité du comportement sur la configuration relâchée par l'annulation de la dissipation intrinsèque :

$$\overline{\underline{\underline{S}}} : \overline{\underline{\underline{d}}}^e - \rho_0 \dot{\psi}^e = 0 \tag{3.73}$$

avec : $\rho_0 = \rho_0^s \theta_0^s$ .

Par ailleurs, on exprime la dérivée de $\psi^e\left(\underline{\underline{C}}^e\right)$ par :

$$\dot{\psi}^e\left(\underline{\underline{C}}^e\right) = \partial_{\underline{\underline{C}}^e} \psi^e : \dot{\underline{\underline{C}}}^e = 2\,\partial_{\underline{\underline{C}}^e} \psi^e : \overline{\underline{\underline{d}}}^e \tag{3.74}$$

Finalement, en remplaçant (3.74) dans (3.73) et en identifiant termes à termes, on obtient l'équation d'état de la phase solide :

$$\overline{\underline{\underline{S}}} = 2\rho_0\, \partial_{\underline{\underline{C}}^e} \psi^e \tag{3.75}$$

D'après les relations (3.67) et (3.69), $\underline{\underline{C}}^e$ peut s'écrire :

$$\underline{\underline{C}}^e \approx \underline{\underline{1}} + 2\underline{\underline{e}} \tag{3.76}$$

et donc : $\psi^e = \psi^e\left(\underline{\underline{e}}\right)$.

En utilisant (3.66), (3.76) et l'hypothèse de découplage entre élasticité et plasticité, on obtient finalement :

$$\underline{\underline{\bar{d}}}^e \approx \underline{\underline{\dot{e}}} \tag{3.77}$$

$$\underline{\underline{\bar{S}}} = 2\rho_0 \, \partial_{\underline{\underline{e}}} \psi \tag{3.78}$$

Afin d'obtenir une relation linéaire entre les contraintes et les déformations, on représente généralement la densité d'énergie libre $\psi$ par un développement quadratique en fonction de ses arguments, à savoir dans notre cas :

$$\psi^e\left(\underline{\underline{e}}\right) = \underline{\underline{\bar{S}}}^0 : \underline{\underline{e}} + \frac{1}{2}\underline{\underline{e}} : \partial^2_{\underline{\underline{e}}\,\underline{\underline{e}}} \psi^e : \underline{\underline{e}} \tag{3.79}$$

$\underline{\underline{\bar{S}}}^0$ représente un état de contraintes constant qui dans le cas présent, est nul du fait de la configuration relâchée choisie (configuration libre de toutes contraintes). Le développement (3.79) devient alors :

$$\psi^e\left(\underline{\underline{e}}\right) = \frac{1}{2}\underline{\underline{e}} : \partial^2_{\underline{\underline{e}}\,\underline{\underline{e}}} \psi^e : \underline{\underline{e}} \tag{3.80}$$

On obtient finalement la relation d'hyperélasticité cherchée à partir de (3.78) et (3.80) :

$$\underline{\underline{\bar{S}}} = \underline{\underline{\underline{\underline{D}}}}^e : \underline{\underline{e}} \tag{3.81}$$

avec : $\underline{\underline{\underline{\underline{D}}}}^e = \rho_0 \, \partial^2_{\underline{\underline{e}}\,\underline{\underline{e}}} \psi^e$, tenseur isotrope constant du 4ème ordre.

Remarque :

Si l'on fait l'hypothèse que les déformations plastiques sont infinitésimales, c'est-à-dire : $J^p \approx 1$ dans (3.81), alors on retombe sur l'équation d'état classique pour les milieux poreux en transformations infinitésimales [COU 91].

Soient $I_1\left(\underline{\underline{e}}\right)$ et $I_2\left(\underline{\underline{e}}\right)$, les deux premiers invariants de $\underline{\underline{e}}$, définis par :

$$\begin{cases} I_1\left(\underline{\underline{e}}\right) = tr\left(\underline{\underline{e}}\right) = \ln J^e \\ I_2\left(\underline{\underline{e}}\right) = \dfrac{1}{2}\underline{\underline{e}} : \underline{\underline{e}} \end{cases} \tag{3.82}$$

D'après les théorèmes de représentation pour les fonctions tensorielles isotropes [SID 84], il est possible d'exprimer $\psi^e$ uniquement en fonction des invariants de $\underline{\underline{e}}$. Nous allons en fait adopter une démarche très utilisée en Géomécanique, qui consiste à décomposer $\psi^e$ en deux parties, l'une $\psi^e_v\left(e_v\right)$, dépendant uniquement des déformations volumiques $e_v = I_1\left(\underline{\underline{e}}\right)$, et

l'autre $\psi_s^e\left(\underline{\underline{e}}\right)$, dépendant des déformations sphériques ou déviatoriques $e_s$ et dans certain modèle, également de $e_v$. Dans ce cas, $\psi^e$ peut alors s'écrire :

$$\rho_0\psi^e\left(\underline{\underline{e}}\right)=\psi_v^e\left(e_v\right)+\psi_s^e\left(\underline{\underline{e}}\right) \tag{3.83}$$

Par ailleurs, le déviateur des déformations élastiques $\underline{\underline{e}}_d$ étant défini par la relation :

$$\underline{\underline{e}}_d=\underline{\underline{e}}-\frac{1}{3}e_v\underline{\underline{1}} \tag{3.84}$$

alors $e_s$ s'écrit :

$$e_s{}^2=2/3\,\underline{\underline{e}}_d:\underline{\underline{e}}_d=4/3\,I_2\left(\underline{\underline{e}}\right)-2/9\,I_1{}^2\left(\underline{\underline{e}}\right) \tag{3.85}$$

Par exemple, dans le cas d'un modèle purement élastique linéaire isotrope, nous pourrons choisir les formes suivantes pour $\psi_v^e\left(e_v\right)$ et $\psi_s^e\left(\underline{\underline{e}}\right)$ :

$$\begin{cases}\psi_v^e\left(e_v\right)=\dfrac{1}{2}\left(\lambda+\dfrac{2}{3}\mu\right)e_v{}^2=\dfrac{1}{2}K\,e_v{}^2\\ \psi_s^e\left(\underline{\underline{e}}\right)=\dfrac{3}{2}\mu\,e_s{}^2=\dfrac{3}{2}G\,e_s{}^2\end{cases} \tag{3.86}$$

où $\lambda$ et $\mu$ sont les coefficients drainés de Lamé.

La relation (3.85) peut également être écrite en fonction de $e_v$ et $e_s$ :

$$\underline{\underline{\bar{S}}}=\rho_0\left(\partial_{e_v}\psi^e\,\underline{\underline{1}}+\partial_{e_s}\psi^e\,\partial_{\underline{\underline{e}}}e_s\right) \tag{3.87}$$

avec :

$$\partial_{\underline{\underline{e}}}e_s=\frac{2}{3e_s}\,\underline{\underline{e}}_d \tag{3.88}$$

Par exemple, on retrouve l'expression de la loi de Hooke en transformations finies :

$$\underline{\underline{\bar{S}}}=K\,e_v\underline{\underline{1}}+2G\,\underline{\underline{e}}_d \tag{3.89}$$

Enfin, on peut également écrire la relation (3.87) sous la forme équivalente suivante :

$$\underline{\underline{\bar{S}}}=\bar{P}\underline{\underline{1}}+\frac{2}{3e_s}\bar{Q}\,\underline{\underline{e}}_d \tag{3.90}$$

en posant :

$$\bar{P} = \rho_0 \, \partial_{e_v} \psi^e \left( \underline{\underline{e}} \right) \tag{3.91}$$

$$\bar{Q} = \rho_0 \, \partial_{e_s} \psi^e \left( \underline{\underline{e}} \right) \tag{3.92}$$

Dans la cas élastique, on aura alors par exemple :

$$\begin{cases} \bar{P} = K \, e_v \\ \bar{Q} = 3G \, e_s \end{cases} \tag{3.93}$$

Remarque :

On voit que le choix de $\underline{\underline{e}}$ permet bien de retrouver la condition : $e_v = 0$, lorsque le matériau est élastiquement incompressible.

***Règle d'écoulement plastique :***

Nous allons à présent nous intéresser aux évolutions irréversibles de la phase solide, caractérisées par la dissipation intrinsèque $W^p$, exprimée d'après l'inégalité de Clausius-Duheim :

$$W^p = \underline{\underline{\tau}}' : \underline{\underline{d}} - \rho_0 \dot{\psi} \geq 0 \tag{3.94}$$

D'après la formule de transport (3.72), la décomposition (3.33) et les relations (3.35), (3.73), on peut finalement exprimer (3.94) de la manière suivante :

$$W^p = \underline{\underline{\bar{S}}}^* : \underline{\underline{\bar{d}}}^p + A \dot{\alpha} \geq 0 \tag{3.95}$$

en posant $\underline{\underline{\bar{S}}}^* = \underline{\underline{\bar{S}}} \cdot \underline{\underline{C}}^e$ et $A = -\rho_0 \, \partial_\alpha U = -\mathcal{H} \alpha$, les forces d'écrouissage associées aux variables internes $\alpha$ avec $\mathcal{H}$, le module d'écrouissage.

La fonction seuil élastoplastique $\bar{f} \left( \underline{\underline{\bar{S}}}, \underline{\underline{C}}^e, A \right)$ vérifie par ailleurs les conditions de Kuhn-Tucker :

$$\begin{cases} \bar{f} \left( \underline{\underline{\bar{S}}}, \underline{\underline{C}}^e, A \right) \leq 0 \\ \left\langle \dot{\lambda}^p \right\rangle \bar{f} \left( \underline{\underline{\bar{S}}}, \underline{\underline{C}}^e, A \right) = 0 \end{cases} \tag{3.96}$$

Le multiplicateur plastique $\dot{\lambda}^p$ est tel que :

$$\langle \dot{\lambda}^p \rangle = \begin{cases} \dot{\lambda}^p & \text{si } \dot{\lambda}^p > 0 \\ 0 & \text{si } \dot{\lambda}^p \leq 0 \end{cases} \tag{3.97}$$

Le principe du travail plastique maximal [HIL 50] nous permet de déterminer l'évolution des flux thermodynamiques $\underline{\underline{\bar{d}}}^p$ et $\dot{\alpha}$, en fonction de $\dot{\lambda}^p$ et d'une fonction $\bar{g}\left(\underline{\underline{\bar{S}}}, \underline{\underline{C}}^e, A\right)$ appelée potentiel plastique. Dans ce cas, la règle d'écoulement plastique s'écrit sous la forme :

$$\begin{cases} \underline{\underline{\bar{d}}}^p = 0 & \text{si } \bar{f} < 0 \\ \underline{\underline{\bar{d}}}^p = \langle \dot{\lambda}^p \rangle \partial_{\underline{\underline{\bar{S}}}} \bar{g} & \text{si } \bar{f} = 0 \end{cases} \tag{3.98}$$

L'évolution des variables internes s'obtient par la relation :

$$\dot{\alpha} = \langle \dot{\lambda}^p \rangle \partial_A \bar{g} \qquad \text{si } \bar{f} = 0 \tag{3.99}$$

On doit également vérifier la condition de compatibilité suivante :

$$\langle \dot{\lambda}^p \rangle \dot{\bar{f}} = 0 \tag{3.100}$$

Sidoroff [SID 84] et Bourgeois [BOU 97] montrent que l'hypothèse d'isotropie de l'énergie libre, ainsi que celle de déformations réversibles infinitésimales, conduisent nécessairement à utiliser la dérivée objective de Jaumann pour exprimer les dérivées matérielles. Bourgeois montre en revanche que le caractère associé ou non de la règle d'écoulement n'a pas d'influence sur la formulation obtenue.

La plupart des auteurs préfèrent adopter une approche plus physique pour décrire les évolutions irréversibles de la phase solide [SIM 85 ; SIM 88a ; SIM 88b ; SIM 92 ; ETE 90 ; CUI 92 ; PER 99a ; PER 99b ; BOR 98]. Ainsi, ils préfèrent utiliser les contraintes $\underline{\underline{\tau}}'$ (Kirchhoff-Trefftz) et le tenseur $\underline{\underline{b}}^e$ (Cauchy-Green gauche élastique) comme mesures eulériennes de contraintes et de déformations. Dans ce cas, l'énergie libre $\psi^e$ dépend de $\underline{\underline{b}}^e$ et l'équation d'état de la phase solide s'écrit alors :

$$\underline{\underline{\tau}}' = 2\rho\, \underline{\underline{b}}^e \cdot \partial_{\underline{\underline{b}}^e} \psi^e \tag{3.101}$$

Et on montre que la relation d'hyperélasticité (3.81), ainsi que la dissipation intrinsèque (3.95) deviennent dans ce cas :

$$\underline{\underline{\tau}}' = \underline{\underline{\underline{\underline{D}}}}^e : \underline{\underline{e}} \tag{3.102}$$

$$W^p = \underline{\underline{\tau}}' : \left( -\frac{1}{2} \mathcal{L}_v \underline{\underline{b}}^e \cdot \underline{\underline{b}}^{e-1} \right) + A\dot{\alpha} \geq 0 \tag{3.103}$$

$\mathcal{L}_v \underline{\underline{b}}^e$ représente la dérivée de Lie et peut s'exprimer simplement par la relation :

$$\mathcal{L}_v \underline{\underline{b}}^e = \underline{\underline{F}} \cdot \dot{\underline{\underline{C}}}^{p-1} \cdot \underline{\underline{F}}^T \tag{3.104}$$

La fonction seuil $f\left(\underline{\underline{\tau}}', B\right)$ est alors écrite dans l'espace des contraintes de Kirchhoff et l'évolution des variables plastiques et internes s'écrit sous la forme :

$$-\frac{1}{2}\mathcal{L}_v \underline{\underline{b}}^e \cdot \underline{\underline{b}}^{e-1} = \langle \dot{\lambda}^p \rangle \partial_{\underline{\underline{\tau}}'} g \qquad \text{si } f = 0 \tag{3.105}$$

$$\dot{\underline{\underline{\alpha}}}^D = \langle \dot{\lambda}^p \rangle \partial_{\underline{\underline{A}}} g \qquad \text{si } f = 0 \tag{3.106}$$

Une autre écriture possible de la règle d'écoulement s'obtient en utilisant (3.65) et (3.104) :

$$\dot{\underline{\underline{C}}}^{p-1} = -2\langle \dot{\lambda}^p \rangle \left( \underline{\underline{F}}^{-1} \cdot \partial_{\underline{\underline{\tau}}'} g \cdot \underline{\underline{F}} \right) \cdot \underline{\underline{C}}^{p-1} \quad \text{si } f = 0 \tag{3.107}$$

Le schéma d'intégration numérique le plus souvent adopté pour déterminer l'évolution de la partie irréversible du comportement est alors un schéma de type « prédicteur élastique - retour radial », identique à celui utilisé en transformations infinitésimales, à la différence près que l'intégration est réalisée ici dans l'espace des contraintes principales de Kirchhoff (voir notamment [SIM 92 ; BOR 98 ; PER 99a ; PER 99b]).

## 3.4 FORMULATION VARIATIONNELLE

### 3.4.1 Principe

Nous allons construire la formulation variationnelle en supposant que les fonctions tests associées aux champs solutions sont indépendantes du temps : elles sont fonctions uniquement des points matériels.

Soient $\mathcal{U}$ et $\mathcal{P}$, les espaces des champs solutions de type déplacement (solide) et pression (fluide) , exprimés par rapport à la configuration de référence :

$$ {}_t\underline{u}^s\left(\underline{X},t'\right)\in\mathcal{U} \quad \mathcal{U}=\left\{ {}_t\underline{u}^s\left(\cdot,t'\right)\in\left[H^1\left(\bar{\Omega}_t\right)\right]^3 \,/\, {}_t\underline{u}^s\left(\underline{X},t'\right)=\underline{\bar{u}},\underline{X}\in\Gamma_{tu}\right\} \tag{3.108} $$

$$ {}_t p\left(\underline{X},t'\right)\in\mathcal{P} \quad \mathcal{P}=\left\{ {}_t p\left(\cdot,t'\right)\in H^1\left(\bar{\Omega}_t\right) \,/\, {}_t p\left(\underline{X},t'\right)=\bar{p},\underline{X}\in\Gamma_{tp}\right\} \tag{3.109} $$

Soient $\hat{\mathcal{U}}$ et $\hat{\mathcal{P}}$, les espaces des déplacements virtuels cinématiquement admissibles et des pressions virtuelles admissibles, dans lesquels seront choisies les fonctions tests :

$$ \hat{\mathcal{U}}=\left\{\underline{\hat{u}}\left(\cdot,t'\right)\in\left[H^1\left(\bar{\Omega}_t\right)\right]^3 \,/\, \underline{\hat{u}}\left(\underline{X},t'\right)=0,\underline{X}\in\Gamma_{tu}\right\} \tag{3.110} $$

$$ \hat{\mathcal{P}}=\left\{\hat{p}\left(\cdot,t'\right)\in H^1\left(\bar{\Omega}_t\right) \,/\, \hat{p}\left(\underline{X},t'\right)=0,\underline{X}\in\Gamma_{tp}\right\} \tag{3.111} $$

Nous avons fait le choix de nous placer dans un cadre où les conditions aux limites de Dirichlet sont imposées au sens fort. Dans ce cas, et pour que les fonctions tests vérifient la condition d'annulation, nous choisirons une méthode de discrétisation spatiale (meshfree ou éléments finis), pour laquelle les fonctions de forme sont interpolantes au niveau des frontières concernées. Cette condition n'est pas nécessaire si l'on utilise des multiplicateurs de Lagrange ou une méthode de pénalisation.

La formulation variationnelle correspondant aux équations (3.17) et (3.18), s'écrit :

$$ \begin{aligned} &\int_{\Omega_t} {}^t\rho^{sat}\, {}_t\ddot{\underline{u}}^s\cdot\underline{\hat{u}}\,d\Omega_t-\int_{\Omega_t} {}_t\underline{\underline{div}}\,{}_t\underline{\underline{\Pi}}'\cdot\underline{\hat{u}}\,d\Omega_t \qquad \forall\underline{\hat{u}}\in\hat{\mathcal{U}} \\ &+\int_{\Omega_t} {}_tJ\,{}_t\underline{\underline{F}}^{-T}\left({}_t\underline{u}^s\right)\cdot\underline{grad\left({}_t p\right)}\cdot\underline{\hat{u}}\,d\Omega_t=\int_{\Omega_t} {}^t\rho^{sat}\,\underline{g}\cdot\underline{\hat{u}}\,d\Omega_t \end{aligned} \tag{3.112} $$

$$ \begin{aligned} &\int_{\Omega_t} {}_t\dot{J}/{}_tJ\,\hat{p}\,d\Omega_t+\int_{\Omega_t} {}_t\hat{n}\beta^w\,\partial_t\left({}_t p\right)\hat{p}\,d\Omega_t \qquad \forall\hat{p}\in\hat{\mathcal{P}} \\ &+\int_{\Omega_t}\hat{K}\,div\left(\rho^w\underline{g}\right)\hat{p}\,d\Omega_t-\int_{\Omega_t}\hat{K}\,{}_t div\left({}_t\underline{\underline{C}}^{-1}\cdot\underline{grad\left({}_t p\right)}\right)\hat{p}\,d\Omega_t=0 \end{aligned} \tag{3.113} $$

Après intégration par parties et en utilisant les conditions aux limites (3.26), ainsi que les relations de dualité entre les contraintes et les déformations, on montre que :

$$-\int_{\Omega_t} {}_t\underline{\underline{div\Pi}}' \cdot \underline{\hat{u}}\, d\Omega_t = \int_{\Omega_t} {}_t\underline{\underline{S}}' : {}_t\delta\underline{\underline{E}}(\underline{\hat{u}})\, d\Omega_t - \int_{\Gamma_{t\sigma}} \underline{\overline{T}} \cdot \underline{\hat{u}}\, d\Gamma_t \tag{3.114}$$

L'équation (3.112) s'écrit finalement :

$$\begin{aligned} &\int_{\Omega_t} {}^t\rho^{sat}\, {}_t\underline{\ddot{u}}^s \cdot \underline{\hat{u}}\, d\Omega_t + \int_{\Omega_t} {}_t\underline{\underline{S}}'\left({}_t\underline{u}^s\right) : {}_t\delta\underline{\underline{E}}(\underline{\hat{u}})\, d\Omega_t \qquad \forall \underline{\hat{u}} \in \hat{\mathcal{U}} \\ &-\int_{\Omega_t} ({}_t p)\, {}_t\delta J(\underline{\hat{u}})\, d\Omega_t = \int_{\Omega_t} {}^t\rho^{sat}\, \underline{g} \cdot \underline{\hat{u}}\, d\Omega_t + \int_{\Gamma_{t\sigma}} \underline{\overline{T}} \cdot \underline{\hat{u}}\, d\Gamma_t \end{aligned} \tag{3.115}$$

avec :

$${}_t\delta J(\underline{\hat{u}}) = {}_t J\, tr\left({}_t\delta\underline{\underline{F}}(\underline{\hat{u}}) \cdot {}_t\underline{\underline{F}}^{-1}\left({}_t\underline{u}^s\right)\right) \tag{3.116}$$

Après intégration par parties et en utilisant les conditions aux limites (3.27), l'équation (3.113) s'écrit également :

$$\begin{aligned} &\int_{\Omega_t} {}_t\dot{J}\left({}_t\underline{u}^s\right)\hat{p}\, d\Omega_t + \int_{\Omega_t} {}_t J\, {}_t\hat{n}\beta^w\, \partial_t({}_t p)\, \hat{p}\, d\Omega_t \qquad \forall \hat{p} \in \hat{\mathcal{P}} \\ &+\int_{\Omega_t} {}_t J\, \hat{K}\, {}_t\underline{\underline{C}}^{-1}\, {}_t\underline{grad}({}_t p) \cdot {}_t\underline{gradp}\, d\Omega_t \\ &= \int_{\Omega_t} {}_t J\, \hat{K}\, \rho^w \underline{g} \cdot {}_t\underline{\underline{F}}^{-1} \cdot {}_t\underline{gradp}\, d\Omega_t + \int_{\Gamma_{t\varphi}} \overline{\Phi}\, \hat{p}\, d\Gamma_t \end{aligned} \tag{3.117}$$

Le problème d'évolution entre les instants $t$ et $t$' revient donc à déterminer les champs inconnus ${}_t\underline{u}^s$ et ${}_t p$ vérifiant les conditions aux limites sur $\Gamma_{tu}$ et $\Gamma_{tp}$, et tels que les équations (3.115) et (3.117) sont satisfaites.

### 3.4.2 Linéarisation de l'équation de conservation de la quantité de mouvement

Le problème variationnel étant fortement non linéaire du point de vue géométrique et du fait de la loi de comportement associée à la phase solide (*cf.* §3.3), il est nécessaire de linéariser l'équation (3.115).

A cette fin, nous allons considérer l'évolution du problème entre deux instants consécutifs (par exemple $t'$ et $t'+\Delta t$), et exprimer les contraintes et déformations cherchées en fonction des valeurs sur la dernière configuration connue et des incréments correspondants (inconnus). Dans ce cas, on peut écrire les relations suivantes :

$${}^{t'+\Delta t}_{t}\underline{\underline{S}}' = {}^{t'}_{t}\underline{\underline{S}}' + {}_t\Delta\underline{\underline{S}}' \tag{3.118}$$

$$ {}^{t'+\Delta t}_{t}\underline{\underline{E}} = {}^{t'}_{t}\underline{\underline{E}} + {}_{t}\Delta\underline{\underline{E}} \tag{3.119} $$

On remarque que dans le cas de la description *UL* ($t = t'$), on aura : ${}^{t}\underline{\underline{\sigma}}' = {}^{t}_{t}\underline{\underline{S}}'$. Par ailleurs, ${}_{t}\Delta\underline{\underline{E}}$ peut également s'écrire de la façon suivante :

$$ {}_{t}\Delta\underline{\underline{E}} = {}_{t}\Delta\underline{\underline{\varepsilon}} + {}_{t}\Delta\underline{\underline{\gamma}} + {}_{t}\Delta\underline{\underline{\eta}} \tag{3.120} $$

où ${}_{t}\Delta\underline{\underline{\varepsilon}}$ et ${}_{t}\Delta\underline{\underline{\eta}}$ sont définis respectivement en (3.47), (3.48) et ${}_{t}\Delta\underline{\underline{\gamma}}$ sera donné par :

$$ {}_{t}\Delta\gamma_{ij} = \frac{1}{2}\left( {}^{t'}_{t}u_{k,i}\ {}_{t}\Delta u_{k,j} + {}_{t}\Delta u_{k,i}\ {}^{t'}_{t}u_{k,j} \right) \tag{3.121} $$

Dans le cas de la description *UL* ($t = t'$), on retrouve : ${}_{t}\Delta\gamma_{ij} = 0$.

Par ailleurs, en supposant que l'on a : ${}_{t}\Delta\underline{\underline{E}} \approx {}_{t}\Delta\underline{\underline{\varepsilon}}'$ et ${}_{t}\delta\underline{\underline{E}} \approx {}_{t}\delta\underline{\underline{\varepsilon}}'$, avec :

$$ {}_{t}\Delta\underline{\underline{\varepsilon}}' = {}_{t}\Delta\underline{\underline{\varepsilon}} + {}_{t}\Delta\underline{\underline{\gamma}} \tag{3.122} $$

On peut finalement simplifier (3.115) comme suit :

$$ \begin{aligned} &\int_{\Omega_t} {}^{t}\rho^{sat}\ {}_{t}\ddot{\underline{u}}^{s} \cdot \hat{\underline{u}}\, d\Omega_t + \int_{\Omega_t} \underline{\underline{D}}^{ep}_{t} : {}_{t}\Delta\underline{\underline{\varepsilon}}'\left( {}_{t}\underline{u}^{s} \right) : {}_{t}\delta\underline{\underline{\varepsilon}}'\left(\hat{\underline{u}}\right) d\Omega_t \qquad \forall \hat{\underline{u}} \in \hat{\mathcal{U}} \\ &+ \int_{\Omega_t} {}^{t'}_{t}\underline{\underline{S}}' : {}_{t}\delta\underline{\underline{\eta}}\left(\hat{\underline{u}}\right) d\Omega_t - \int_{\Omega_t} \left( {}_{t}p \right) {}_{t}\delta J\left(\hat{\underline{u}}\right) d\Omega_t = \int_{\Omega_t} {}^{t}\rho^{sat}\, \underline{g} \cdot \hat{\underline{u}}\, d\Omega_t \\ &+ \int_{\Gamma_{t\sigma}} \overline{\underline{T}} \cdot \hat{\underline{u}}\, d\Gamma_t - \int_{\Omega_t} {}^{t'}_{t}\underline{\underline{S}}' : {}_{t}\delta\underline{\underline{\varepsilon}}'\left(\hat{\underline{u}}\right) d\Omega_t \end{aligned} \tag{3.123} $$

## 3.5 DISCRETISATIONS

### 3.5.1 Discrétisation spatiale sur la configuration initiale

Soient $U^h \subset U$ et $P^h \subset P$, deux sous-espaces de dimension finie, dans lesquels seront cherchées les solutions approchées ${}_t\underline{u}^h$ et ${}_t p^h$ de la formulation variationnelle (3.115), c'est-à-dire engendrés par les bases de fonctions lagrangiennes $\left\{\phi_I^s\right\}_{I=1}^{N^u}$ et $\left\{\phi_I^w\right\}_{I=1}^{N^w}$ :

$$\underline{X} \in \overline{\Omega}_t, \begin{cases} {}_t\underline{u}^h\left(\underline{X}, t'\right) = \sum_{I=1}^{N^u} \phi_I^s\left(\underline{X}\right) {}_t u_{Ij}\left(t'\right) \underline{e}_j \qquad j = 1,2,3 \\ {}_t p^h\left(\underline{X}, t'\right) = \sum_{I=1}^{N^w} \phi_I^w\left(\underline{X}\right) {}_t p_I\left(t'\right) \end{cases} \tag{3.124}$$

où ${}_t u_{Ij}\left(t'\right)$, ${}_t p_I\left(t'\right)$ représentent les contributions nodales en déplacement (solide) et en pression (fluide) à l'instant $t' \geq t$, mesurées par rapport à la configuration de référence $\Omega_t$, contributions égales aux valeurs nodales en cas d'approximation interpolante. $N^u$ et $N^p$ représentent le nombre de nœuds utilisés pour construire les approximations respectivement pour les champs de déplacement et de pression. Les fonctions de forme sont supposées ne dépendre que de la variable d'espace et une approximation différente peut être envisagée pour les deux champs cherchés.

Par ailleurs, l'écriture (3.124) s'adapte aussi bien aux méthodes meshfree présentées au chapitre 1, qu'à la méthode des éléments finis.

Pour les fonctions tests, on prendra :

$$\underline{X} \in \overline{\Omega}_t, \begin{cases} \hat{\underline{u}} = \phi_J^s\left(\underline{X}\right) \underline{e}_j & J \leq N^u \\ \hat{p} = \phi_J^w\left(\underline{X}\right) & J \leq N^w \end{cases} \tag{3.125}$$

Cette discrétisation conduit à l'écriture matricielle suivante pour les équations (3.123) et (3.117) :

$$\begin{cases} {}_t\boldsymbol{M}^s \cdot {}_t\ddot{\boldsymbol{U}} + \left({}_t\boldsymbol{K}_L + {}_t\boldsymbol{K}_{NL}\right) {}_t\Delta\boldsymbol{U} - {}_t\boldsymbol{L}\left({}_t\boldsymbol{U}\right) \cdot {}_t\boldsymbol{P} = {}_t\boldsymbol{F}^s - {}_t\boldsymbol{F}^{\text{int}}\left({}^{t'}_{t}\boldsymbol{U}\right) \\ {}_t\boldsymbol{L}^T\left({}_t\boldsymbol{U}\right) \cdot {}_t\dot{\boldsymbol{U}} + {}_t\boldsymbol{M}^{\boldsymbol{w}} \cdot {}_t\dot{\boldsymbol{P}} + {}_t\boldsymbol{H}\left({}_t\boldsymbol{U}\right) \cdot {}_t\boldsymbol{P} = {}_t\boldsymbol{F}^w\left({}_t\boldsymbol{U}\right) \end{cases} \tag{3.126}$$

où ${}_t\boldsymbol{U}$ (${}_t\Delta\boldsymbol{U}$) et ${}_t\boldsymbol{P}$ désignent les vecteurs de contributions nodales en déplacements (incréments) et pressions mesurés par rapport à la configuration $\Omega_t$. Les différents termes matriciels sont donnés par les expressions suivantes :

$${}_t M^s_{IJij} = \delta_{ij} \int_{\Omega_t} {}_t\rho^{sat} \phi_I^s \, \phi_J^s \, d\Omega_t \tag{3.127}$$

$$ {}_tL_{IJj} = \int_{\Omega_t} {}_tJ\,\phi_I^w\,\partial_{X_k}\phi_J^s\,\partial_{x_k}X_j\,d\Omega_t \qquad k=1,2,3 \tag{3.128} $$

$$ {}_tM_{IJ}^w = \int_{\Omega_t} {}_tJ\,{}_t\hat{n}\beta^w\phi_I^w\,\phi_J^w\,d\Omega_t \tag{3.129} $$

$$ {}_tH_{IJ} = \int_{\Omega_t} {}_tJ\hat{K}\,\partial_{x_k}X_i\,\partial_{x_k}X_j\,\partial_{X_j}\phi_I^w\,\partial_{X_i}\phi_J^w\,d\Omega_t \qquad i,j,k=1,2,3 \tag{3.130} $$

$$ {}_tF_{Jj}^s = \int_{\Omega_t} {}_t\rho^{sat}\,\underline{g}\cdot\phi_J^s\underline{e}_j\,d\Omega_t + \int_{\Gamma_{t\sigma}} \overline{\underline{T}}\cdot\phi_J^s\underline{e}_j\,d\Gamma_t \tag{3.131} $$

$$ {}_tF_J^w = \int_{\Omega_t} {}_tJ\hat{K}\,{}_t\rho^w\,{}_t\hat{n}\,\partial_{x_j}X_i\,\underline{g}\cdot\underline{e}_j\,\partial_{X_i}\phi_J^w\,d\Omega_t + \int_{\Gamma_{t\varphi}} \overline{\Phi}\,\phi_J^w\,d\Gamma_t \quad i,j=1,2,3 \tag{3.132} $$

$$ {}_tF_{Jj}^{\text{int}} = \int_{\Omega_t} \left[{}_tB_L^J\right]_{ji}^T {}_t^{t'}\hat{S}_i\,d\Omega_t \tag{3.133} $$

avec ${}_t^{t'}\underline{\hat{S}}^T = \left\{ {}_t^{t'}S'_{11} \quad {}_t^{t'}S'_{22} \quad {}_t^{t'}S'_{33} \quad {}_t^{t'}S'_{12} \quad {}_t^{t'}S'_{13} \quad {}_t^{t'}S'_{23} \right\}$, le vecteur associé au tenseur ${}_t^{t'}\underline{\underline{S}}'$ calculé au point d'évaluation (par exemple, au point de Gauss).

On a par ailleurs posé :

$$ {}_t\boldsymbol{K}_L = \int_{\Omega_t} {}_t\boldsymbol{B}_L^T \cdot \underline{\underline{\underline{\underline{D}}}}_t \cdot {}_t\boldsymbol{B}_L\,d\Omega_t \tag{3.134} $$

$$ {}_t\boldsymbol{K}_{NL} = \int_{\Omega_t} {}_t\boldsymbol{B}_{NL}^T \cdot {}_t^{t'}\underline{\underline{S}}' \cdot {}_t\boldsymbol{B}_{NL}\,d\Omega_t \tag{3.135} $$

où : ${}_t\boldsymbol{B}_L = \left[ {}_t\boldsymbol{B}_L^1 \cdots {}_t\boldsymbol{B}_L^J \cdots {}_t\boldsymbol{B}_L^N \right]$, ${}_t\boldsymbol{B}_{NL} = \left[ {}_t\boldsymbol{B}_{NL}^1 \cdots {}_t\boldsymbol{B}_{NL}^J \cdots {}_t\boldsymbol{B}_{NL}^N \right]$.

${}_t\boldsymbol{B}_L^J$ est définie par la relation suivante :

$$ {}_t\boldsymbol{B}_L^J = {}_t\boldsymbol{B}_{L0}^J + {}_t\boldsymbol{B}_{L1}^J \tag{3.136} $$

avec :

$$ {}_t\boldsymbol{B}_{L0}^J = \begin{bmatrix} {}_t\phi_{J,1} & 0 & 0 \\ 0 & {}_t\phi_{J,2} & 0 \\ 0 & 0 & {}_t\phi_{J,3} \\ {}_t\phi_{J,2} & {}_t\phi_{J,1} & 0 \\ {}_t\phi_{J,3} & 0 & {}_t\phi_{J,1} \\ 0 & {}_t\phi_{J,3} & {}_t\phi_{J,2} \end{bmatrix} \tag{3.137} $$

$$ {}_t\boldsymbol{B}_{L1}^J = \begin{bmatrix} {}^{t'}_{t}a_{11}\,{}_t\phi_{J,1} & {}^{t'}_{t}a_{21}\,{}_t\phi_{J,1} & {}^{t'}_{t}a_{31}\,{}_t\phi_{J,1} \\ {}^{t'}_{t}a_{12}\,{}_t\phi_{J,2} & {}^{t'}_{t}a_{22}\,{}_t\phi_{J,2} & {}^{t'}_{t}a_{32}\,{}_t\phi_{J,2} \\ {}^{t'}_{t}a_{13}\,{}_t\phi_{J,3} & {}^{t'}_{t}a_{23}\,{}_t\phi_{J,3} & {}^{t'}_{t}a_{33}\,{}_t\phi_{J,3} \\ {}^{t'}_{t}a_{11}\,{}_t\phi_{J,2} + {}^{t'}_{t}a_{12}\,{}_t\phi_{J,1} & {}^{t'}_{t}a_{21}\,{}_t\phi_{J,2} + {}^{t'}_{t}a_{22}\,{}_t\phi_{J,1} & {}^{t'}_{t}a_{31}\,{}_t\phi_{J,2} + {}^{t'}_{t}a_{32}\,{}_t\phi_{J,1} \\ {}^{t'}_{t}a_{11}\,{}_t\phi_{J,3} + {}^{t'}_{t}a_{13}\,{}_t\phi_{J,1} & {}^{t'}_{t}a_{21}\,{}_t\phi_{J,3} + {}^{t'}_{t}a_{23}\,{}_t\phi_{J,1} & {}^{t'}_{t}a_{31}\,{}_t\phi_{J,3} + {}^{t'}_{t}a_{33}\,{}_t\phi_{J,1} \\ {}^{t'}_{t}a_{12}\,{}_t\phi_{J,3} + {}^{t'}_{t}a_{13}\,{}_t\phi_{J,2} & {}^{t'}_{t}a_{22}\,{}_t\phi_{J,3} + {}^{t'}_{t}a_{23}\,{}_t\phi_{J,2} & {}^{t'}_{t}a_{32}\,{}_t\phi_{J,3} + {}^{t'}_{t}a_{33}\,{}_t\phi_{J,2} \end{bmatrix} \tag{3.138} $$

$$ \left[ {}_t\boldsymbol{B}_{NL}^J \right]^T = \begin{bmatrix} {}_t\phi_{J,1} & {}_t\phi_{J,2} & {}_t\phi_{J,3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & {}_t\phi_{J,1} & {}_t\phi_{J,2} & {}_t\phi_{J,3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & {}_t\phi_{J,1} & {}_t\phi_{J,2} & {}_t\phi_{J,3} \end{bmatrix} \tag{3.139} $$

et :

$$ {}^{t'}_{t}a_{ij} = \sum_{I=1}^{N} {}_t\phi^s_{J,j}\ {}^{t'}_{t}u^s_{J,i} \quad i,j = 1,2,3 \tag{3.140} $$

Dans le cas de la description *UL*, on note que ${}^{t}_{t}a_{ij} = 0$.

### 3.5.2 Discrétisation en temps

La semi-discrétisation en temps s'effectue en découpant l'intervalle d'étude $[0,T]$ en pas de temps $\Delta t$ non nécessairement constants et en utilisant un schéma « *General Newmark* » [ZIE 91] de type prédiction – correction, c'est-à-dire utilisant la méthode de Newmark pour les variables liées au solide (GN22) et la $\theta$-méthode pour celles liées au fluide (GN11).

Nous posons : $\Delta t = t_{n+1} - t_n$, le pas de temps entre les étapes $n$ et $(n+1)$ quelconques. Nous posons par ailleurs : ${}_t\Delta\underline{\ddot{u}}_{n+1} = {}_t\underline{\ddot{u}}_{n+1} - {}_t\underline{\ddot{u}}_n$ et ${}_t\Delta\dot{p}_{n+1} = {}_t\dot{p}_{n+1} - {}_t\dot{p}_n$. Les mesures des champs solutions cherchés à l'étape $(n+1)$ sont réalisées pour des points homologues entre la configuration $\Omega_{n+1}$ et la configuration de référence $\Omega_t$ ($\Omega_t = \Omega_n$ pour la description *UL* et $\Omega_t = \Omega_0$ pour la description *TL*).

Dans ce cas, la phase de prédiction du schéma choisi peut alors s'écrire de la façon suivante :

$$\begin{cases} {}_t\tilde{\underline{u}}_{n+1} = {}_t\underline{u}_n + \Delta t\ {}_t\dot{\underline{u}}_n + \dfrac{1}{2}\Delta t^2\ {}_t\ddot{\underline{u}}_n \\ {}_t\tilde{\dot{\underline{u}}}_{n+1} = {}_t\dot{\underline{u}}_n + \Delta t\ {}_t\ddot{\underline{u}}_n \\ {}_t\tilde{p}_{n+1} = {}_t p_n + \Delta t\ {}_t\dot{p}_n \end{cases} \tag{3.141}$$

Et la phase de correction s'écrira :

$$\begin{cases} {}_t\underline{u}_{n+1} = {}_t\tilde{\underline{u}}_{n+1} + \beta\Delta t^2\ {}_t\Delta\ddot{\underline{u}}_{n+1} \\ {}_t\dot{\underline{u}}_{n+1} = {}_t\tilde{\dot{\underline{u}}}_{n+1} + \gamma\Delta t\ {}_t\Delta\ddot{\underline{u}}_{n+1} \\ {}_t p_{n+1} = {}_t\tilde{p}_{n+1} + \theta\Delta t\ {}_t\Delta\dot{p}_{n+1} \end{cases} \tag{3.142}$$

avec $(\beta,\gamma)\in[0,1]^2$ pour la méthode de Newmark et $\theta\in[0,1]$.

Remarque :

Nous rappelons que dans le cas d'une résolution linéaire et implicite, le schéma choisi est inconditionnellement stable si $\gamma\geq\frac{1}{2}$, $\theta\geq\frac{1}{2}$ et $\beta=\frac{1}{4}\left(\gamma+\frac{1}{2}\right)^2$.

Les inconnues choisies sont les incréments de déplacements ${}_t\Delta\underline{u}_{n+1} = {}_t\underline{u}_{n+1} - {}_t\underline{u}_n$ et de pression ${}_t\Delta p_{n+1} = {}_t p_{n+1} - {}_t p_n$, ce qui nous amène à éliminer les termes d'accélérations et de vitesses de l'étape $(n+1)$, en écrivant :

$$\begin{cases} {}_t\Delta\ddot{\underline{u}}_{n+1} = \dfrac{{}_t\underline{u}_{n+1} - {}_t\tilde{\underline{u}}_{n+1}}{\beta\Delta t^2} \\ {}_t\Delta\dot{\underline{u}}_{n+1} = \Delta t\ {}_t\ddot{\underline{u}}_n + \dfrac{\gamma}{\beta\Delta t}\left({}_t\underline{u}_{n+1} - {}_t\tilde{\underline{u}}_{n+1}\right) \\ {}_t\Delta\dot{p}_{n+1} = \dfrac{{}_t p_{n+1} - {}_t\tilde{p}_{n+1}}{\theta\Delta t} \end{cases} \tag{3.143}$$

Finalement, le système d'équations (3.126) devient :

$$\left(\frac{{}_t\boldsymbol{M}^s}{\beta\Delta t^2}+{}_t\boldsymbol{K}_{Ln+1}+{}_t\boldsymbol{K}_{NL}\right){}_t\Delta\boldsymbol{U}_{n+1}-{}_t\boldsymbol{L}_{n+1}\cdot{}_t\Delta\boldsymbol{P}_{n+1}=$$
$${}_t\boldsymbol{F}^s_{n+1}-{}_t\boldsymbol{M}^s\cdot\left({}_t\ddot{\boldsymbol{U}}_n+\frac{{}_t\boldsymbol{U}_n-{}_t\tilde{\boldsymbol{U}}_{n+1}}{\beta\Delta t^2}\right)-{}_t\boldsymbol{F}^{\text{int}}\left({}_t\boldsymbol{U}_{\boldsymbol{n}}\right)+{}_t\boldsymbol{L}_{n+1}\cdot{}_t\boldsymbol{P}_n \tag{3.144}$$

$$-{}_t\boldsymbol{L}^T_{n+1}\cdot{}_t\Delta\boldsymbol{U}_{n+1}-\frac{\beta}{\gamma\theta}\left[{}_t\boldsymbol{M}^w_{n+1}+\theta\Delta t\,{}_t\boldsymbol{H}_{n+1}\right]\cdot{}_t\Delta\boldsymbol{P}_{n+1}=$$
$$\frac{\beta\Delta t}{\gamma}\left[-{}_t\boldsymbol{F}^w_{n+1}+{}_t\boldsymbol{M}^w_{n+1}\cdot\left({}_t\dot{\boldsymbol{P}}_n+\frac{{}_t\boldsymbol{P}_n-{}_t\tilde{\boldsymbol{P}}_{n+1}}{\theta\Delta t}\right)+{}_t\boldsymbol{H}_{n+1}\cdot{}_t\boldsymbol{P}_n+{}_t\boldsymbol{L}^T_{n+1}\cdot{}_t\tilde{\dot{\boldsymbol{U}}}_{n+1}\right] \tag{3.145}$$
$$+{}_t\boldsymbol{L}^T_{n+1}\cdot\left({}_t\boldsymbol{U}_n-{}_t\tilde{\boldsymbol{U}}_{n+1}\right)$$

en notant : ${}_t\boldsymbol{L}_{n+1}={}_t\boldsymbol{L}\left({}_t\boldsymbol{U}_{n+1}\right)$, ${}_t\boldsymbol{H}_{n+1}={}_t\boldsymbol{H}\left({}_t\boldsymbol{U}_{n+1}\right)$ et ${}_t\boldsymbol{M}^w_{n+1}={}_t\boldsymbol{M}^w\left({}_t\boldsymbol{U}_{n+1}\right)$.

## 3.6 RESOLUTION DU PROBLEME NON LINEAIRE

Du fait de la nature fortement non linéaire du problème, il est nécessaire d'adopter un schéma itératif pour permettre la résolution des équations (3.144) et (3.145).

Dans le cas d'un schéma de type Newton-Raphson et connaissant les champs solutions de l'étape *n*, le système à résoudre à l'itération $(i+1)$ de l'étape $(n+1)$ s'écrit :

$$\underbrace{\begin{bmatrix} \dfrac{{}_t\boldsymbol{M}^s}{\beta\Delta t^2} + {}_t\boldsymbol{K}^i_{n+1} & -\boldsymbol{L}^i_{n+1} \\ -{}_t\boldsymbol{L}^{i\,T}_{n+1} & -\dfrac{\beta}{\gamma\theta}\left[{}_t\boldsymbol{M}^{w,i}_{n+1} + \theta\Delta t\ {}_t\boldsymbol{H}^i_{n+1}\right] \end{bmatrix}}_{\boldsymbol{J}^i_{n+1}} \begin{Bmatrix} \Delta\boldsymbol{U}^{i+1}_{n+1} \\ \Delta\boldsymbol{P}^{i+1}_{n+1} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{F}^{i+1}_{n+1} \\ \boldsymbol{G}^{i+1}_{n+1} \end{Bmatrix} \tag{3.146}$$

$\boldsymbol{J}^i_{n+1}$ représente la matrice jacobienne pour l'itération *i*. L'opérateur de rigidité tangente ${}_t\boldsymbol{K}^i_{n+1}$ est également déterminé par l'expression :

$${}_t\boldsymbol{K}^i_{n+1} = \int_{\Omega_t} {}_t\boldsymbol{B}^T_L \cdot \underset{t\equiv n+1}{\underline{\underline{D}}}^i \cdot {}_t\boldsymbol{B}_L\, d\Omega_t + {}_t\boldsymbol{K}_{NL}\left({}_t\boldsymbol{U}_n\right) \tag{3.147}$$

Par ailleurs, on initialise en prenant : $\boldsymbol{J}^0_{n+1} = \boldsymbol{J}_n$ et ${}_t\boldsymbol{K}^0_{n+1} = {}_t\boldsymbol{K}_n$.

Dans la méthode de Newton-Raphson classique, les termes de la matrice jacobienne du système (3.146) sont évalués à chaque itération, ce qui est très coûteux en temps *CPU*. C'est pourquoi, il est usuel d'utiliser l'algorithme modifié, dans lequel l'évaluation de la matrice est réalisée moins fréquemment (par exemple en début d'étape seulement). Cette méthode présente néanmoins l'inconvénient d'avoir une convergence plus lente.

Dans notre cas, nous avons choisi un schéma de type Newton-Raphson modifié.

Les vecteurs ${}_t\boldsymbol{U}^{i+1}_{n+1}$, ${}_t\boldsymbol{P}^{i+1}_{n+1}$, ${}_t\boldsymbol{F}^{s,i+1}_{n+1}$ et ${}_t\boldsymbol{F}^{w,i+1}_{n+1}$ sont définis à l'itération $(i+1)$ par les relations :

$$\begin{cases} {}_t\boldsymbol{U}^{i+1}_{n+1} = {}_t\boldsymbol{U}^i_{n+1} + {}_t\Delta\boldsymbol{U}^{i+1}_{n+1} \\ {}_t\boldsymbol{P}^{i+1}_{n+1} = {}_t\boldsymbol{P}^i_{n+1} + {}_t\Delta\boldsymbol{P}^{i+1}_{n+1} \end{cases} \tag{3.148}$$

$$\begin{cases} {}_t\boldsymbol{F}^{i+1}_{n+1} = {}_t\boldsymbol{F}^s_{n+1} - {}_t\boldsymbol{M}^s \cdot {}_t\ddot{\boldsymbol{U}}^i_{n+1} - {}_t\boldsymbol{F}^{\text{int}}\left({}_t\boldsymbol{U}^i_{n+1}\right) + {}_t\boldsymbol{L}^i_{n+1} \cdot {}_t\boldsymbol{P}^i_{n+1} \\ {}_t\boldsymbol{G}^{i+1}_{n+1} = \dfrac{\beta\Delta t}{\gamma}\Big[-{}_t\boldsymbol{F}^{w,i}_{n+1} + {}_t\boldsymbol{M}^{w,i}_{n+1} \cdot {}_t\dot{\boldsymbol{P}}^i_{n+1} + {}_t\boldsymbol{H}^i_{n+1} \cdot {}_t\boldsymbol{P}^i_{n+1} \\ \qquad\qquad + {}_t\boldsymbol{L}^{i\,T}_{n+1} \cdot \left({}_t\dot{\boldsymbol{U}}^i_{n+1} - \gamma\Delta t\ {}_t\ddot{\boldsymbol{U}}_n\right)\Big] \end{cases} \tag{3.149}$$

avec :

$$\begin{cases} {}_t\ddot{\boldsymbol{U}}^i_{n+1} = \dfrac{{}_t\boldsymbol{U}^i_{n+1} - {}_t\tilde{\boldsymbol{U}}_{n+1}}{\beta\Delta t^2} + {}_t\ddot{\boldsymbol{U}}_n \\ {}_t\dot{\boldsymbol{U}}^i_{n+1} = {}_t\tilde{\dot{\boldsymbol{U}}}_{n+1} + \gamma\Delta t\ {}_t\ddot{\boldsymbol{U}}^i_{n+1} \\ {}_t\dot{\boldsymbol{P}}^i_{n+1} = \dfrac{{}_t\boldsymbol{P}^i_{n+1} - {}_t\tilde{\boldsymbol{P}}_{n+1}}{\theta\Delta t} + {}_t\dot{\boldsymbol{P}}_n \end{cases} \tag{3.150}$$

Pour chaque itération, on vérifie les tests de convergence suivants :

$$\begin{cases} \left\| {}_t\Delta\boldsymbol{U}^{i+1}_{n+1} \right\| \le tol \left\| {}_t\boldsymbol{U}^{i+1}_{n+1} \right\| \\ \left\| {}_t\Delta\boldsymbol{P}^{i+1}_{n+1} \right\| \le tol \left\| {}_t\boldsymbol{P}^{i+1}_{n+1} \right\| \end{cases} \tag{3.151}$$

$$\begin{cases} \left\| {}_t\boldsymbol{F}^{i+1}_{n+1} \right\| \le Ftol \cdot \left\| {}_t\boldsymbol{F}^{0}_{n+1} \right\| \\ \left\| {}_t\boldsymbol{G}^{i+1}_{n+1} \right\| \le Ftol \cdot \left\| {}_t\boldsymbol{G}^{0}_{n+1} \right\| \end{cases} \tag{3.152}$$

où *tol* et *Ftol* sont des tolérances fixées respectivement pour les incréments de déplacements et pressions, ainsi que pour les déséquilibres résiduels.

## 3.7 CONCLUSIONS

Dans ce chapitre, nous avons présenté le modèle dynamique adapté aux milieux poreux saturés en transformations finies. Pour ce faire, nous avons rappelé les principales équations de conservation tout d'abord en description eulérienne, puis en description lagrangienne totale.

Nous avons ensuite présenté les différentes approches possibles pour écrire la loi de comportement de la phase solide en transformations finies. L'approche la plus classique consiste à considérer la phase solide comme un matériau hypoélastique, ce qui conduit à une écriture incrémentale de la loi. L'autre approche consiste à se placer dans le cadre de la Thermodynamique des Processus Irréversibles en considérant la phase solide comme un matériau hyperélastique et en exprimant la loi à partir des densités volumiques d'énergie libre et de dissipation intrinsèque.

Enfin, la dernière partie de ce chapitre était consacrée à la résolution numérique du problème non linéaire. Pour ce faire, nous avons présenté la formulation variationnelle en description lagrangienne, cette formulation servant de base pour la discrétisation en espace et en temps, et pour la résolution numérique suivant un schéma itératif de Newton.

Les applications numériques regroupées dans le chapitre 4 suivant, vont nous permettre de valider la mise en œuvre de la formulation présentée dans ce chapitre. Au niveau des tests utilisant des modèles de comportement élastoplastiques, seule l'approche hypoélastique a été mise en œuvre et testée.

# 4. Applications Géomécaniques en Transformations Finies

## 4.1 INTRODUCTION

Au cours des chapitres précédents, nous nous sommes bornés à appliquer les principales méthodes meshfree à notre disposition sur des problèmes mécaniques statiques en transformations infinitésimales et pour des matériaux élastiques, ceci afin de valider leur implantation dans le logiciel *MoveFree* d'une part et également pour évaluer et comparer leurs performances numériques. Nous visions par ailleurs à déterminer les méthodes les plus susceptibles d'être appliquées dans un cadre géomécanique plus réaliste, par exemple la simulation d'un sol naturel soumis à une sollicitation sismique, en utilisant des lois rhéologiques adaptées.

Au niveau de ce chapitre, nous allons tout d'abord présenter quelques tests de validation concernant les divers développements relatifs à la formulation dynamique en transformations finies. Puis, nous abordons l'étude d'un cas réel, à savoir la réponse sismique du barrage de « *El Infiernillio* » (Mexique), lors du tremblement de terre du 14/3/1979. Les observations réalisées sur site sont comparées aux résultats issus des simulations numériques effectuées avec différentes méthodes meshfree (RKPM, EFGM et RPIM) et avec la méthode des éléments finis (FEM), en utilisant la loi de comportement multi-mécanismes de Hujeux [AUB 82 ; HUJ 85] dans le cadre des transformations finies. Dans ce dernier cas, les méthodes EFGM et RKPM ont été rendues interpolantes sur la totalité du domaine d'étude.

## 4.2 TESTS MECANIQUES SIMPLES EN TRANSFORMATIONS FINIES

### 4.2.1 Introduction

Le principe des tests qui vont suivre, consiste à résoudre l'équation d'équilibre statique sur le domaine initial homogène $\Omega = [0,1]\times[0,1]$, en imposant des conditions aux limites en déplacements sur la frontière $\Gamma_u$, sous la forme :

$$\forall t \in [0,1], \begin{cases} \overline{u}_x(t) = x(t) - X \\ \overline{u}_y(t) = y(t) - Y \end{cases} \tag{4.1}$$

avec $t = 0$, la configuration initiale et $t = 1$, la configuration finale.

Nous avons considéré ici des chemins de déformation dont la solution est connue : extension bi-axiale, rotation complète et compression – extension à volume constant. Par ailleurs, l'état de contrainte initial est supposé nul.

Pour réaliser ces tests, 8 nœuds répartis sur la frontière et un nœud intérieur ont été utilisés (*cf.* Fig. 4.1). Nous avons par ailleurs considéré un domaine d'influence radial de rayon maximal $h_{max} = 1.2$ et une base primaire polynomiale linéaire. De plus, l'intégration réalisée était de type Gauss avec une seule cellule coïncidant avec $\Omega$ et 2x2 points d'intégration. Enfin, le matériau constitutif était de type hypoélastique avec pour module d'Young : $E = 250\ kPa$, et comme coefficient de Poisson : $\nu = 0$.

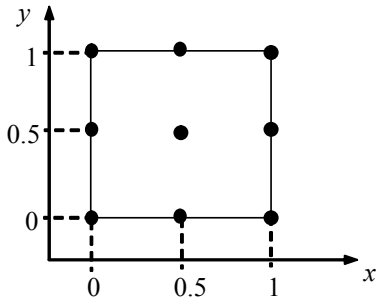



***Fig. 4.1 : Schéma des tests simples en transformations finies.***

Remarque :

Pour les trois tests considérés, les résultats obtenus avec les différentes méthodes meshfree étaient identiques et n'ont donc pas été différentiés.

### 4.2.2 Extension bi-axiale

Pour ce test, le pas de déformation imposé est de 2% (pas de temps : $\Delta t = 0.02\,\text{sec}$). Par ailleurs, le mouvement considéré est le suivant :

$$\begin{cases} x(t) = (1+t)\,X \\ y(t) = (1+t)\,Y \end{cases} \tag{4.2}$$

La solution analytique en déplacements est donc la suivante :

$$\begin{cases} u_x(t) = t\,X \\ u_y(t) = t\,Y \end{cases} \tag{4.3}$$

Les contraintes de Cauchy, obtenue en considérant un taux objectif de Jaumann, s'écrivent :

$$\begin{cases} \sigma_{xx}(t) = \sigma_{yy}(t) = \dfrac{E}{(1+\nu)(1-2\nu)} \ln(1+t) \\ \sigma_{xy}(t) = 0 \end{cases} \tag{4.4}$$

L'erreur commise sur les déplacements étaient de l'ordre de $10^{-7}$, et celle sur les contraintes était relativement faible (*cf.* Fig. 4.2 à Fig. 4.4).

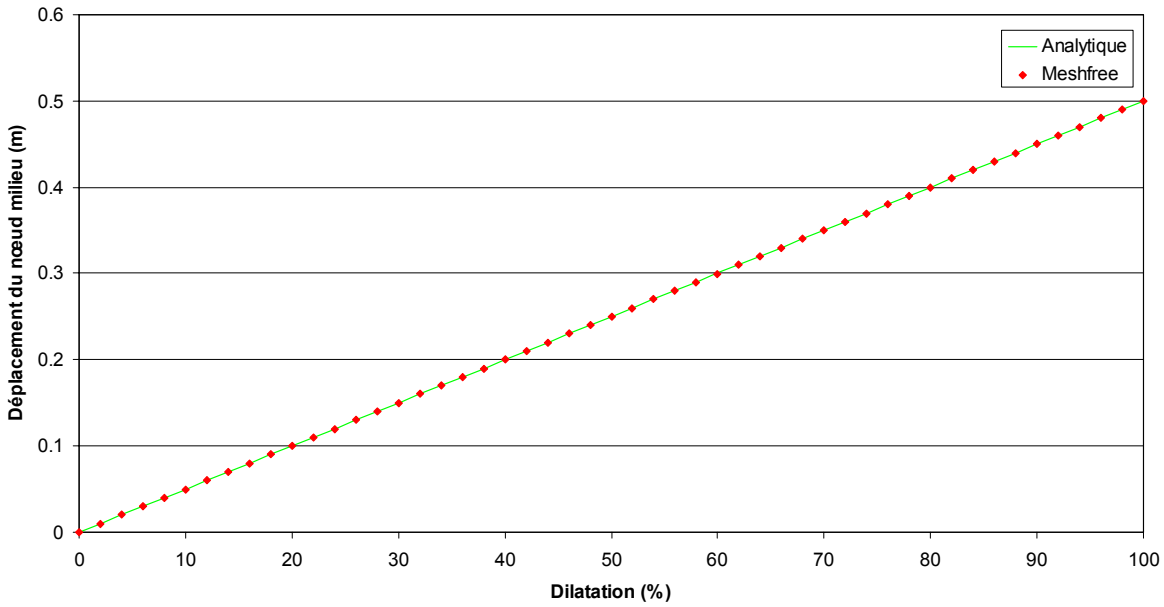



***Fig. 4.2 : Extension bi-axiale : déplacements analytique et calculé.***

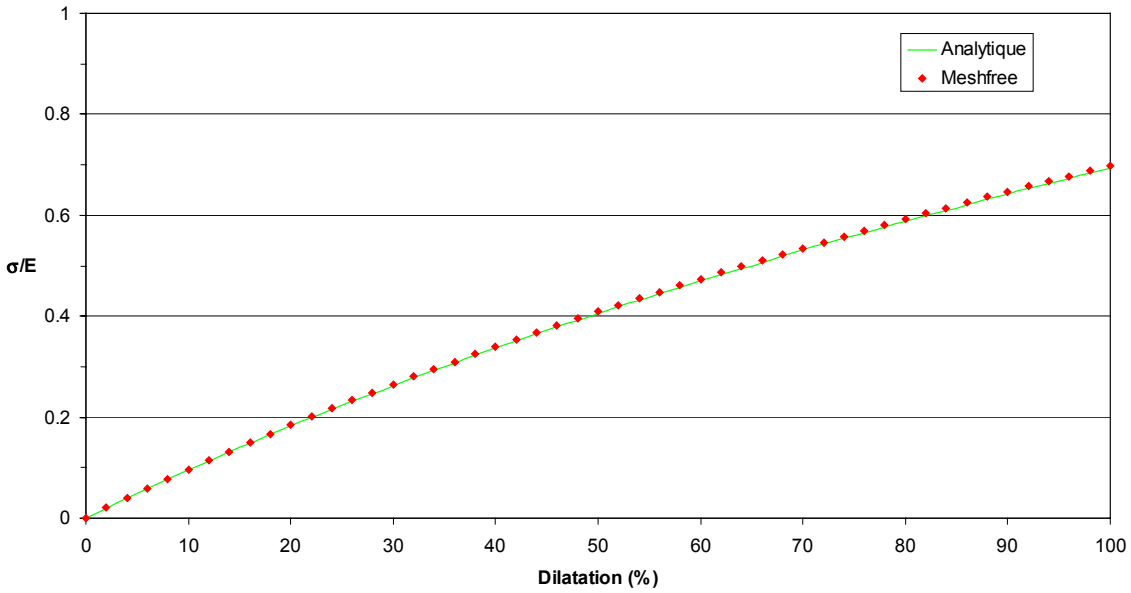



*Fig. 4.3 : Extension bi-axiale : contraintes normalisées analytique et calculée.*

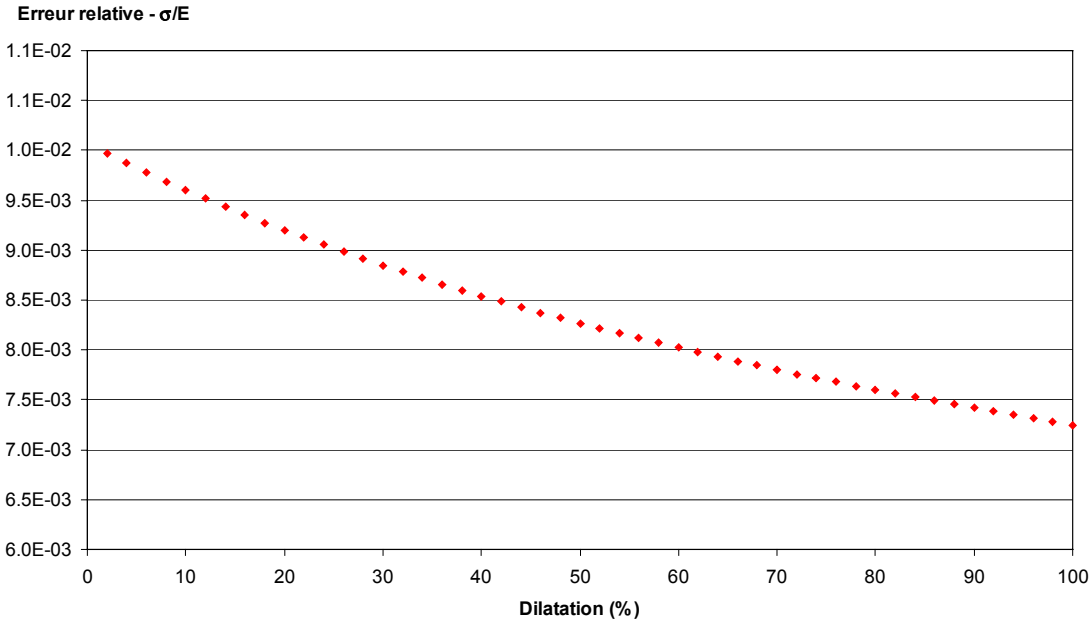



*Fig. 4.4 : Extension bi-axiale : erreur relative sur les contraintes normalisées (calcul meshfree).*

### 4.2.3 Rotation complète

Pour ce test, le pas de rotation imposé est de 7.2 degrés (pas de temps : $\Delta t = 0.02\,\text{sec}$). Par ailleurs, le mouvement considéré est le suivant (*cf.* Fig. 4.5) :

$$\begin{cases} x(t) = X\cos(2\pi t) - Y\sin(2\pi t) \\ y(t) = X\sin(2\pi t) + Y\cos(2\pi t) \end{cases} \tag{4.5}$$

La solution analytique en déplacements est donc la suivante :

$$\begin{cases} u_x(t) = X\left[(1+t)\cos(2\pi t) - 1\right] - Y\sin(2\pi t) \\ u_y(t) = (1+t)\,X\sin(2\pi t) + Y\left[\cos(2\pi t) - 1\right] \end{cases} \tag{4.6}$$

Les contraintes exactes de Cauchy sont par ailleurs nulles.

L'erreur commise sur les déplacements étaient ici de l'ordre de $10^{-7}$ (*cf.* Fig. 4.6), et celle sur les contraintes était de l'ordre de $10^{-15}$ (quelle que soit la composante considérée).

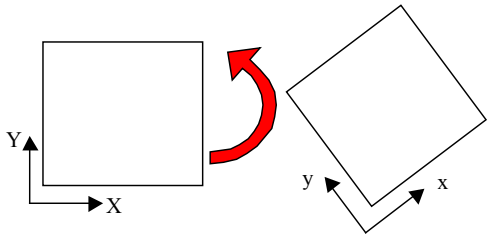



***Fig. 4.5 : Schéma du test de rotation complète.***

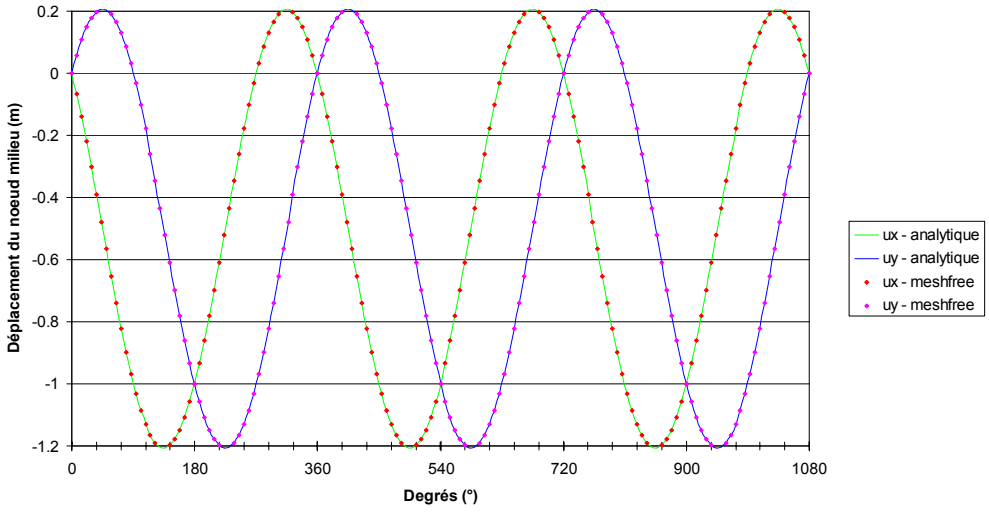



***Fig. 4.6 : Rotation complète : déplacements analytiques et calculés.***

### 4.2.4 Compression – extension à volume constant

Pour ce test, le pas de déformation imposé est de 2% (pas de temps : $\Delta t = 0.02\,\text{sec}$). Par ailleurs, le mouvement considéré est le suivant (*cf.* Fig. 4.5) :

$$\begin{cases} x(t) = (1+t)\,X \\ y(t) = Y/(1+t) \end{cases} \tag{4.7}$$

La solution analytique en déplacements est donc la suivante :

$$\begin{cases} u_x(t) = t\;X \\ u_y(t) = -t\;Y/(1+t) \end{cases} \tag{4.8}$$

Les contraintes de Cauchy, obtenue en considérant un taux objectif de Jaumann, s'écrivent :

$$\begin{cases} \sigma_{xx}(t) = \dfrac{E}{(1+\nu)(1-2\nu)} \ln(1+t) \\ \sigma_{yy}(t) = \dfrac{-E}{(1+\nu)(1-2\nu)} \ln(1+t) \\ \sigma_{xy}(t) = 0 \end{cases} \tag{4.9}$$

L'erreur commise sur les déplacements étaient également de l'ordre de $10^{-7}$, et celle sur les contraintes est très faible (*cf.* Fig. 4.9).

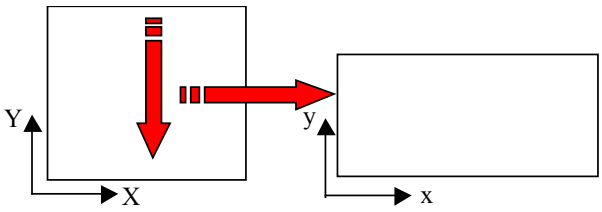



***Fig. 4.7 : Schéma du test de compression – extension à volume constant.***

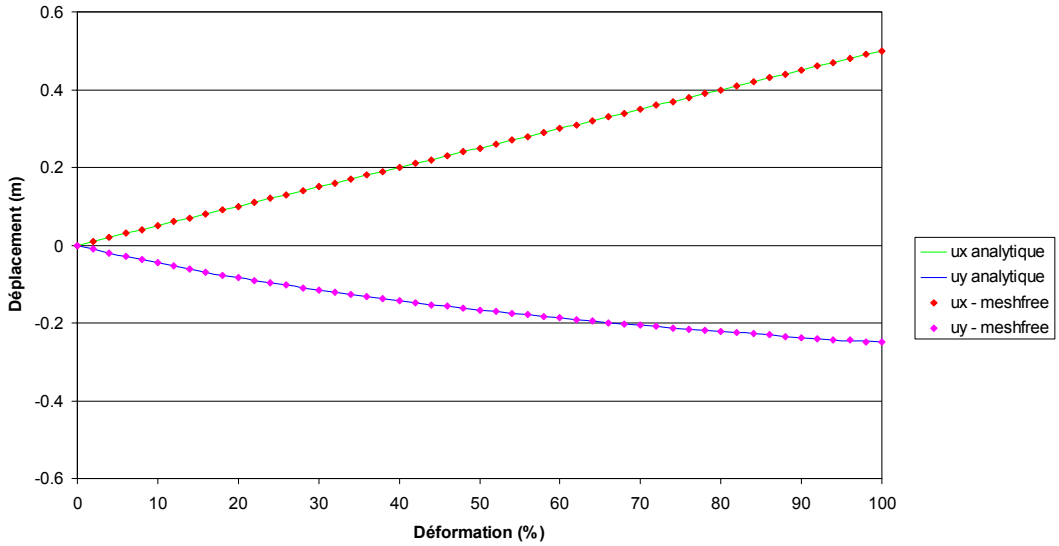



***Fig. 4.8 : Compression – extension à volume constant : déplacements analytiques et calculés.***

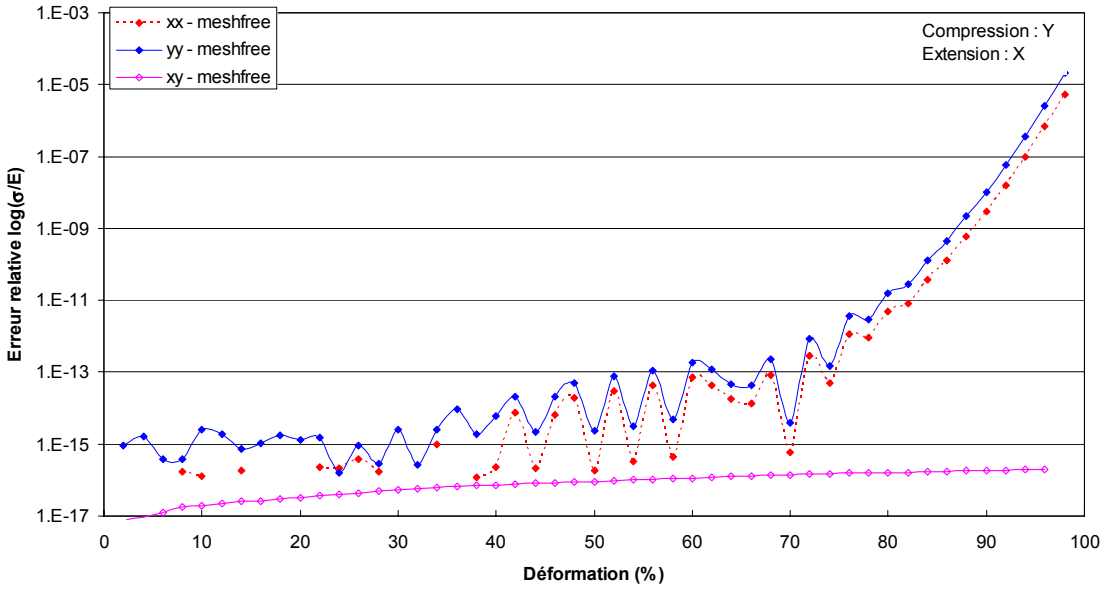



***Fig. 4.9 : Compression – extension à volume constant : erreur relative sur les contraintes normalisées (calcul meshfree).***

## 4.3 SIMULATIONS GEOMECANIQUES SIMPLES EN TRANSFORMATIONS FINIES

### 4.3.1 Colonne de sol soumise à une charge répartie en surface

#### *4.3.1.1 Cas dynamique drainé*

Dans cette application monodimensionnelle, nous avons considéré une colonne de sol monophasique en déformations planes, reposant sur une base rigide et soumise à une charge mécanique *P* répartie en surface (*cf.* Fig. 4.10). Dans un premier temps, nous supposons que la colonne est constituée d'un seul matériau élastique linéaire (transformations infinitésimales). Par ailleurs, les forces de volumes ne sont pas prises en compte.

Pour les différents calculs, les domaines d'influence ont une taille fixe, la base primaire est de type polynomiale linéaire et l'intégration est réalisée au moyen d'éléments linéiques de un mètre comprenant chacun 2 points de Gauss. Nous avons par ailleurs utilisé une fonction de pondération polynomiale modifiée pour la méthode RKPM et une fonction *Spline2* pour la méthode EFGM (interpolation globale sur le domaine). La méthode RPIM utilise ici une fonction CSRBF de Wendland ($C^2$).

Enfin, les paramètres de discrétisation en temps sont $\Delta t = 0.01s$ et $\gamma = 0.5, \beta = 0.25$ pour le schéma de Newmark.

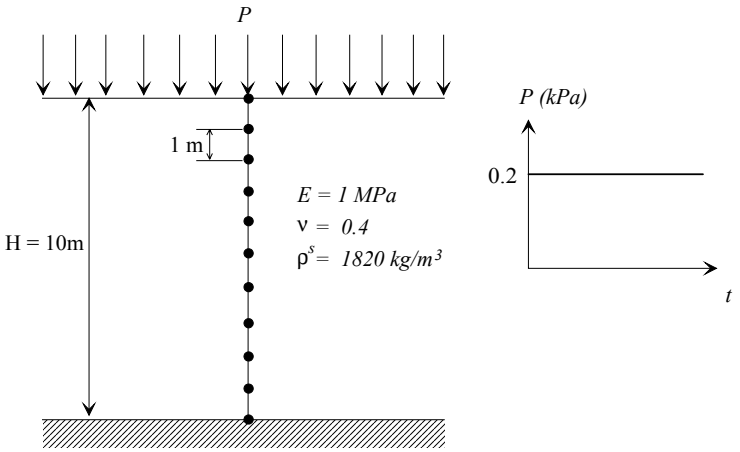



***Fig. 4.10 : Colonne 1D monophasique : discrétisation, propriétés et chargement.***

Lorsque la taille du rayon est fixée à 1.2, on constate que toutes les méthodes meshfree donnent des résultats identiques, et que la différence avec un calcul éléments finis classique est très faible (*cf.* Fig. 4.11).

Lorsque l'on augmente le rayon du domaine d'influence à 4, on s'aperçoit que la méthode RPIM est peu sensible aux variations, tandis que la dispersion devient très forte pour les méthodes EFGM et RKPM (*cf.* Fig. 4.12).

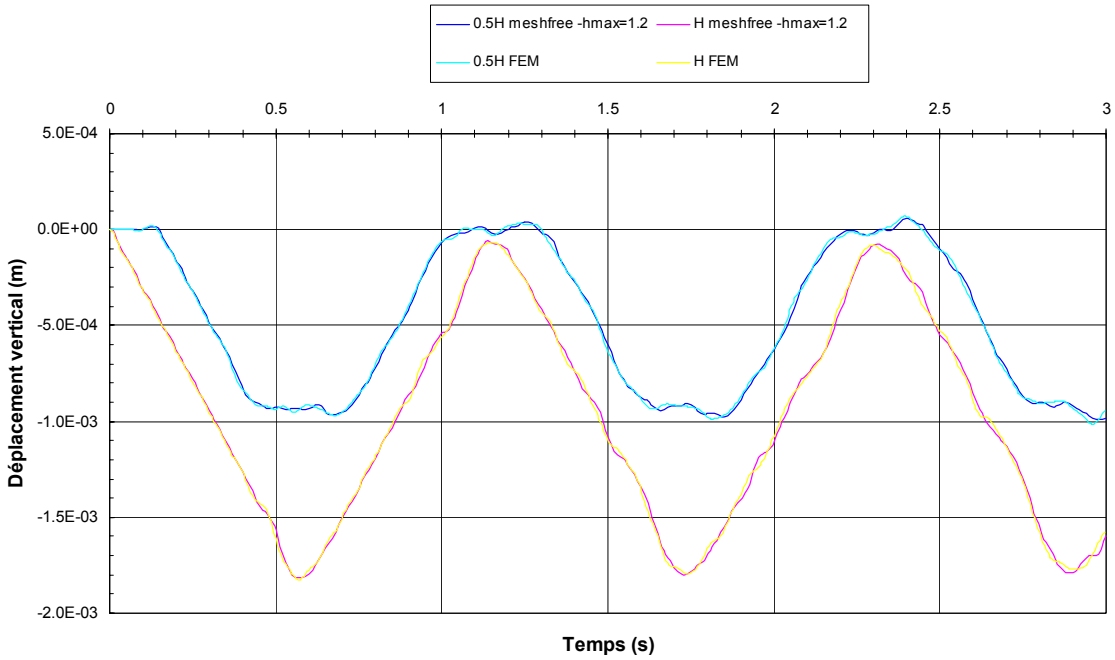



*Fig. 4.11 : Colonne 1D monophasique : comparaison meshfree- FEM pour $h_{max}$ = 1.2.*

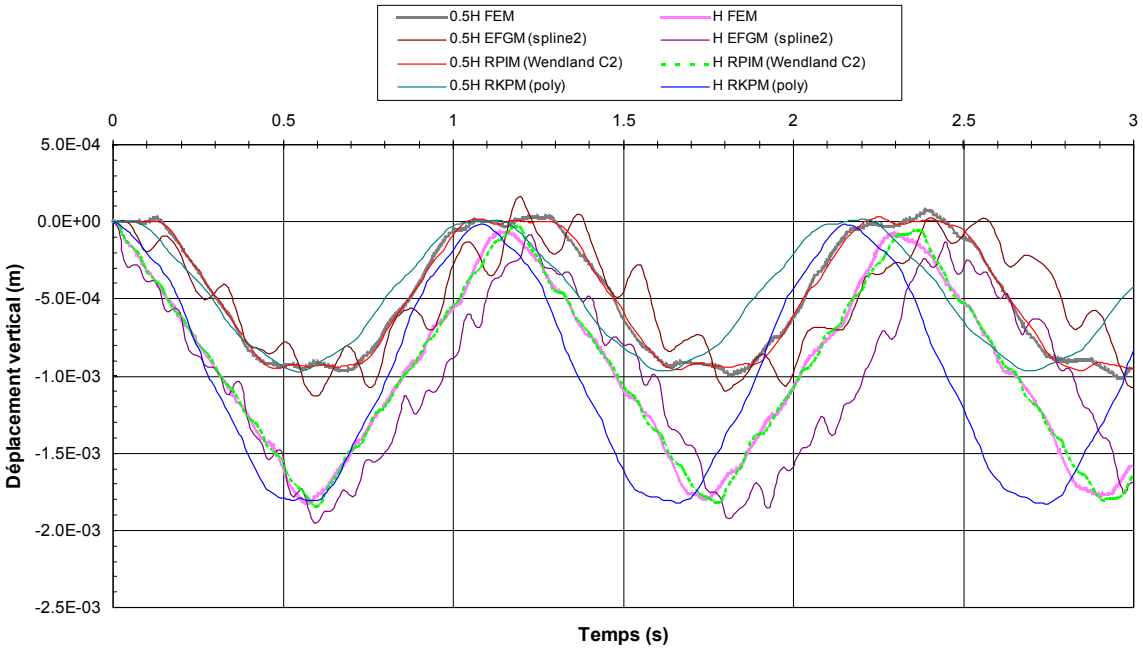



*Fig. 4.12 : Colonne 1D monophasique : comparaisons meshfree - FEM pour $h_{max}$ = 4.*

Lorsque la taille du rayon est fixée à 1.2 et pour le chargement *P* pris égal à 0.2kPa, il n'y a pas de différence entre l'approche transformations finies (*TF*) ou infinitésimales (*TI*), les déformations demeurant trop faibles (*cf*. Fig. 4.13). Les deux réponses commencent à différer lorsque *P* augmente (*cf*. Fig. 4.14).

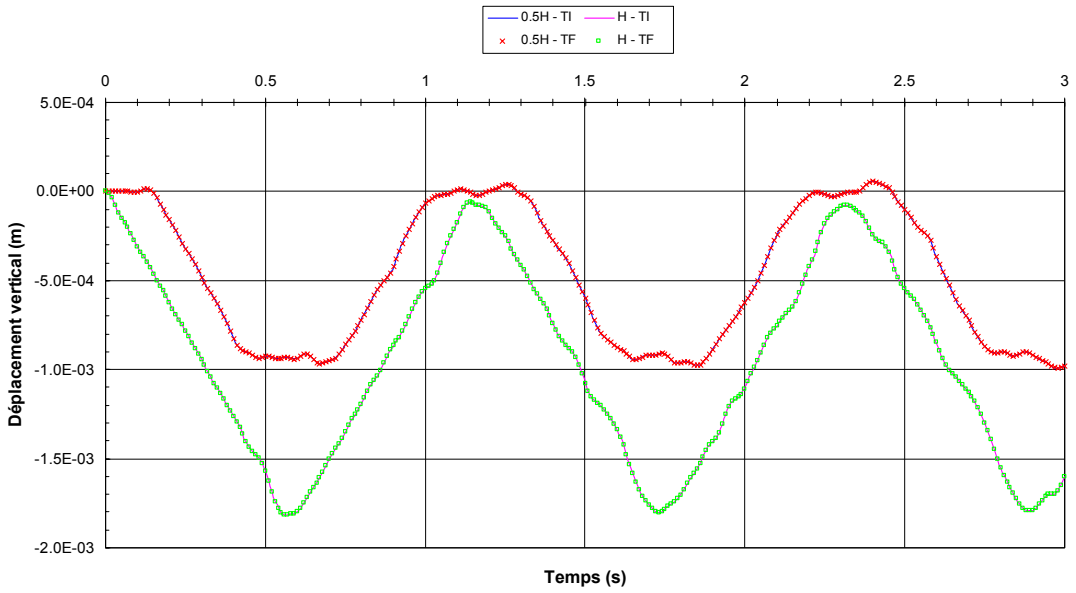



***Fig. 4.13 : Colonne 1D monophasique : comparaison entre les transformations finies et infinitésimales pour P = 0.2kPa (méthode RPIM, $h_{max}$ = 1.2).***

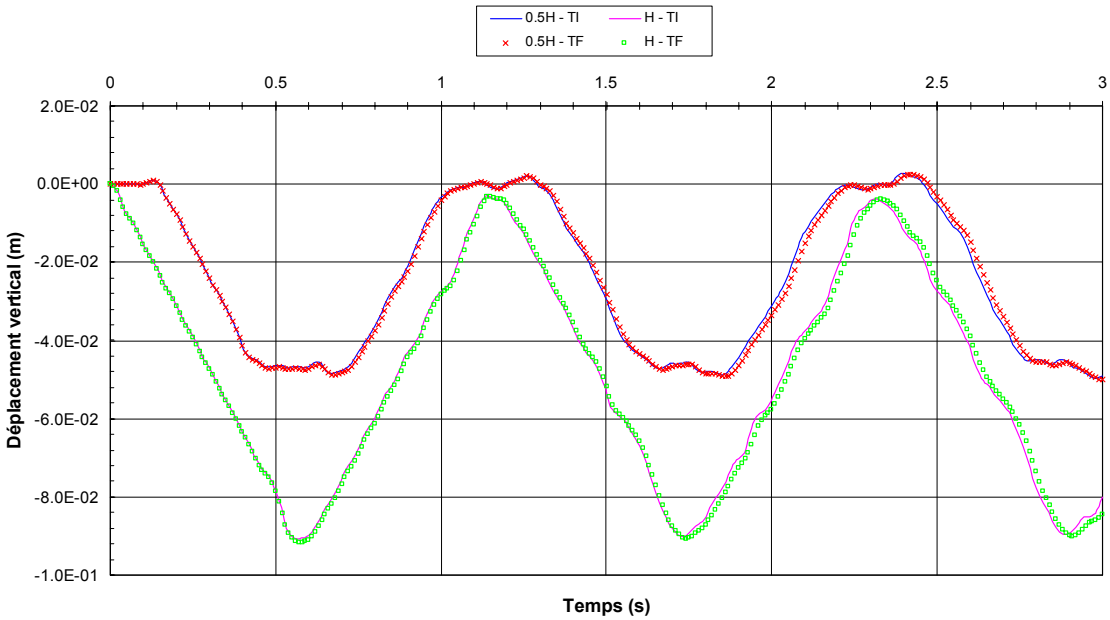



***Fig. 4.14 : Colonne 1D monophasique : comparaison entre les transformations finies et infinitésimales pour P = 10kPa (méthode RPIM, $h_{max}$ = 1.2).***

### *4.3.1.2 Cas statique saturé*

Nous étudions cette fois le cas d'une colonne de sol élastique en déformations planes, reposant sur une base rigide et soumise à une charge mécanique statique *P* répartie en surface. Le milieu envisagé est perméable, de perméabilité $\hat{K} = 10^{-6} m/s$ et de porosité initiale ${}^0n = 0.3$, saturé par un fluide compressible (avec $\rho^w = 1000\ kg/m^3$, $\beta^w = 4.5\,10^{-7} kg^{-1}.m.s^{-2}$ et $\mu^w = 1.053\,10^{-3} kg.m^{-1}.s^{-1}$). Les pressions sont imposées nulles en surface et la base est supposée non drainante (*cf.* Fig. 4.15). Par ailleurs, les forces de volumes ne sont pas prises en compte.

Pour les différents calculs, la taille des domaines d'influence est fixée à 3.2, la base primaire est de type polynomiale linéaire et l'intégration est réalisée au moyen d'éléments linéiques de trois mètre comprenant chacun 2 points de Gauss. Par ailleurs, nous avons choisi de montrer les résultats obtenus en utilisant la méthode RPIM avec une fonction CSRBF de Wendland ($C^2$), les autres méthodes envisagées (RKPM, EFGM) fournissant dans ce cas des résultats identiques.

Enfin, les paramètres de discrétisation en temps sont $\Delta t = 0.1s$ et $\theta = 1$ pour la $\theta$-méthode.

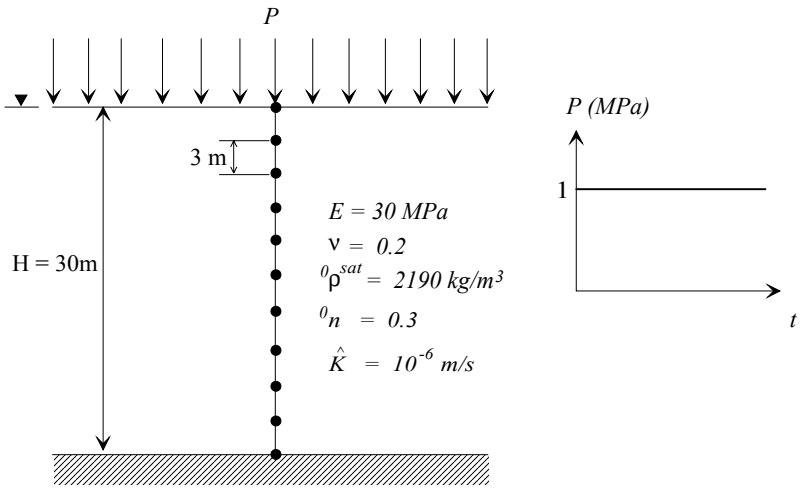



***Fig. 4.15 : Colonne 1D saturée : discrétisation, propriétés et chargement.***

Lorsque la charge est appliquée en surface, une partie des efforts est reprise par le fluide. Puis, la surpression générée par le chargement se dissipe rapidement du fait de l'écoulement à travers le milieu perméable.

Nous comparons dans un premier temps, les résultats obtenus avec les méthodes RPIM et FEM pour des pressions interstitielles calculées en trois points de la colonne, à savoir proche du sommet (0.1H), à mi-hauteur (0.5H) et à la base (H), et en faisant l'hypothèse des transformations finies ou infinitésimales. En transformations infinitésimales, les résultats sont identiques entre les deux méthodes (*cf.* Fig. 4.17).

En transformations finies, on constate généralement que pour une profondeur donnée de la colonne, la pression décroît plus rapidement que dans le cas infinitésimal. Physiquement, ceci s'explique en effet par le fait que la porosité diminuant au cours de la déformation (gradient de la transformation inférieur à 1 en compression), la vitesse du fluide augmente de manière inversement proportionnelle (*cf.* Fig. 4.17). La valeur finale pour les déplacements verticaux est quant à elle légèrement plus élevée dans le cas des transformations infinitésimales, du fait d'une pression moyenne effective également plus importante au cours du chargement (*cf.* Fig. 4.19).

Par ailleurs, on voit que les pressions calculées avec la méthode FEM sont dans un premier temps supérieures à celles calculées avec la méthode RPIM et qu'elles décroissent ensuite plus rapidement (*cf.* Fig. 4.18). Les différences constatées peuvent s'expliquer par une moins bonne convergence de la méthode FEM dans ce cas. Les déplacements verticaux sont quant à eux assez proches (*cf.* Fig. 4.20).

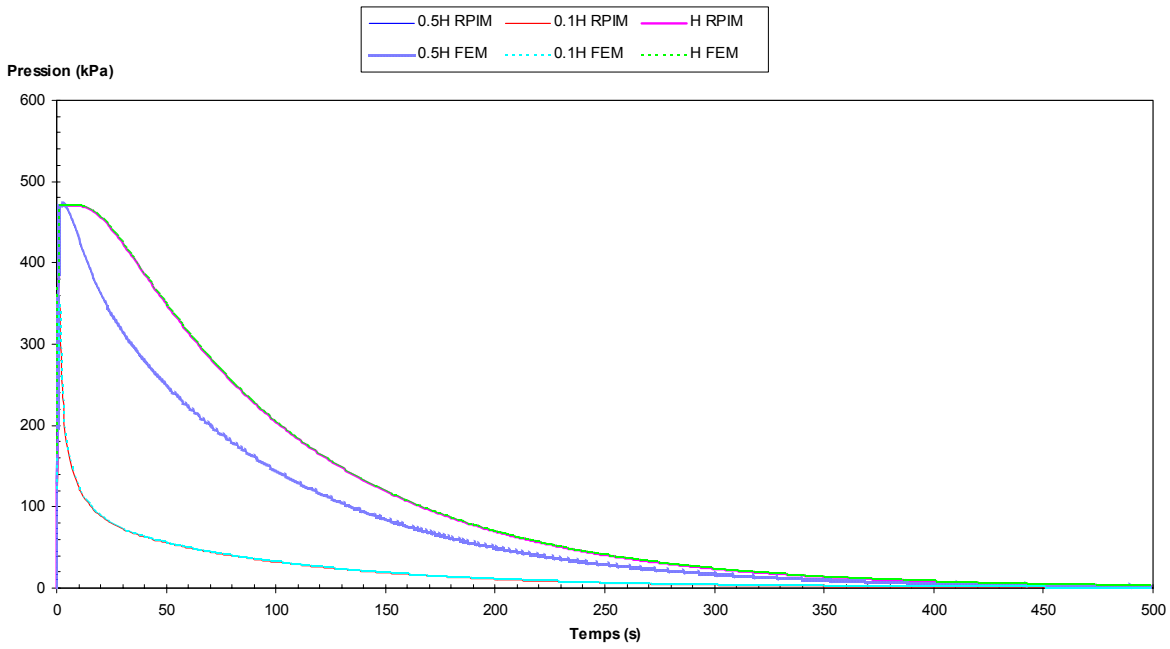



***Fig. 4.16 : Colonne 1D saturée : comparaison des pressions interstitielles calculées avec les méthodes RPIM et FEM en transformations infinitésimales.***

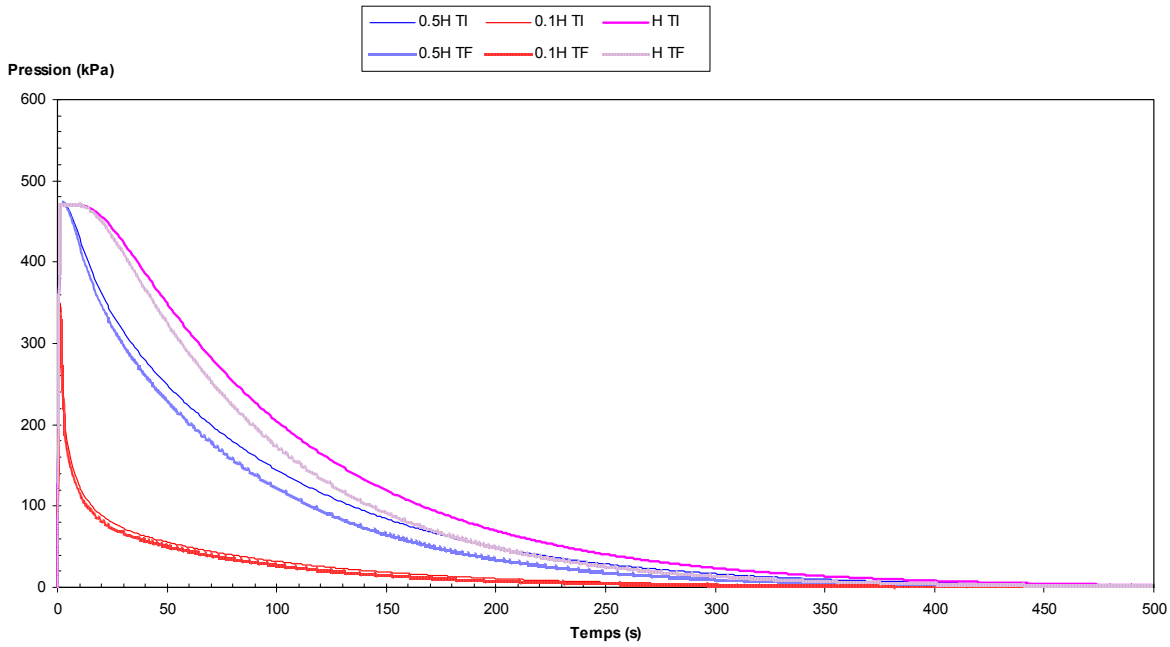



*Fig. 4.17 : Colonne 1D saturée : comparaison des pressions interstitielles calculées avec la méthode RPIM, en transformations finies et infinitésimales.*

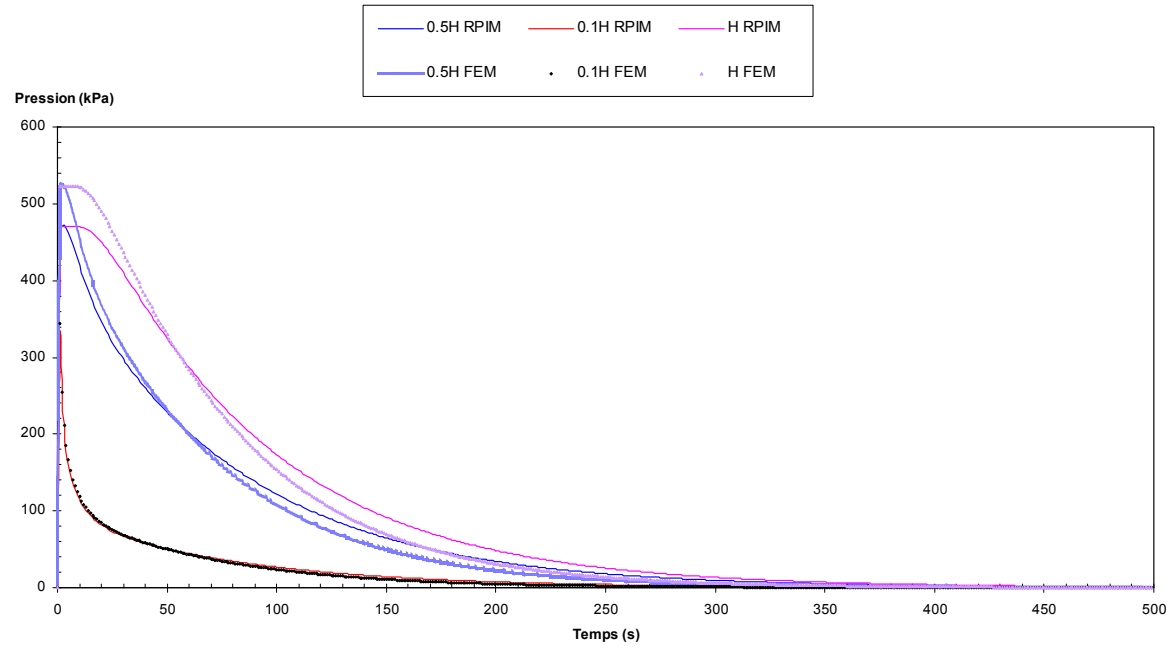



*Fig. 4.18 : Colonne 1D saturée : comparaison des pressions interstitielles calculées avec les méthodes RPIM et FEM en transformations finies.*

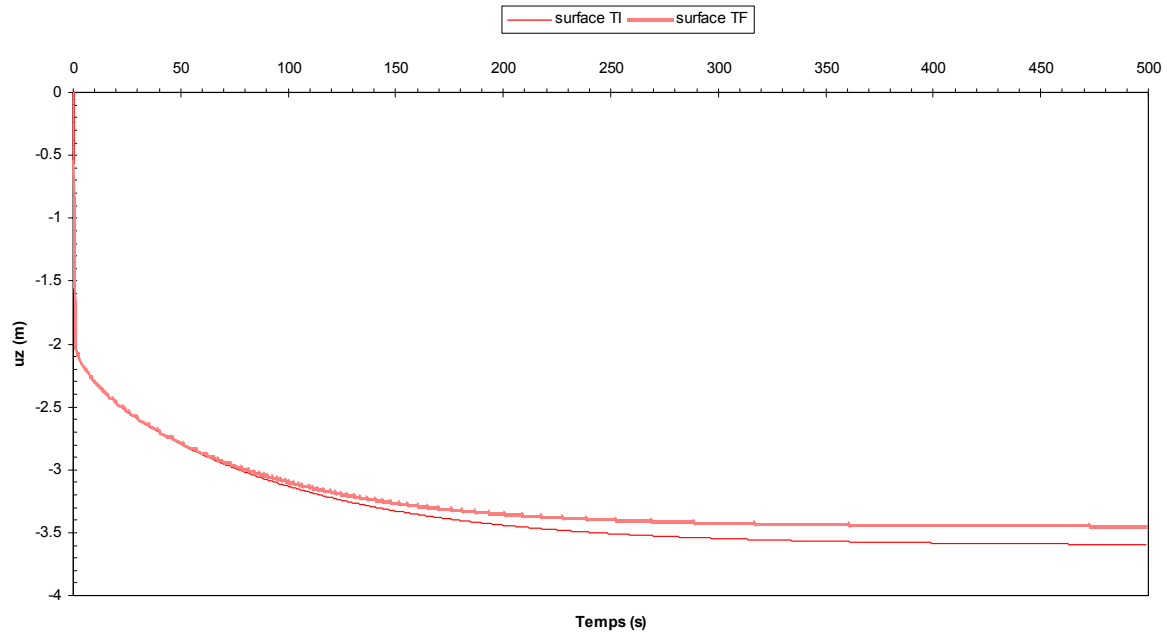



***Fig. 4.19 : Colonne 1D saturée : comparaison des déplacements verticaux calculés au sommet avec la méthode RPIM, en transformations finies et infinitésimales.***

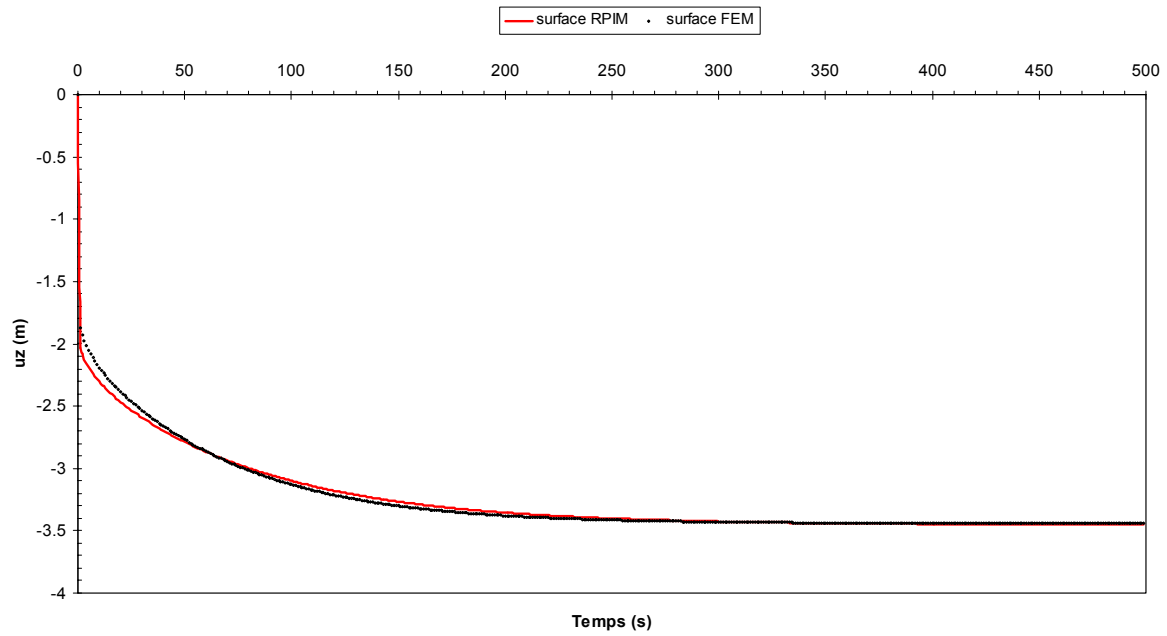



***Fig. 4.20 : Colonne 1D saturée : comparaison des déplacements verticaux calculés au sommet avec les méthodes RPIM et FEM en transformations finies.***

### 4.3.2 Massif de sol drainé soumis à une charge dynamique en surface

Dans cette application bidimensionnelle, nous étudions cette fois le cas d'un massif de sol monophasique en déformations planes, reposant sur une base rigide et soumise à une charge mécanique $P$ centrée en surface (*cf.* Fig. 4.21). Le milieu envisagé est constitué d'un seul matériau, dont le comportement élastoplastique (ou hypo-plastique) est régi par la loi de Drücker-Prager, que nous avons supposée associée avec un angle de frottement interne $\phi = 33°$ et une cohésion nulle. Par ailleurs, les forces de volumes ne sont pas prises en compte.

Pour les différents calculs, les domaines d'influence sont radiaux de rayon adaptatif au moins égal à 2, la base primaire est de type polynomiale linéaire et l'intégration est réalisée au moyen d'éléments quadrangulaires de 1.5m comprenant chacun 2x2 points de Gauss. Enfin, les paramètres de discrétisation en temps sont $\Delta t = 0.05s$ et $\gamma = 0.5, \beta = 0.25$ pour le schéma de Newmark.

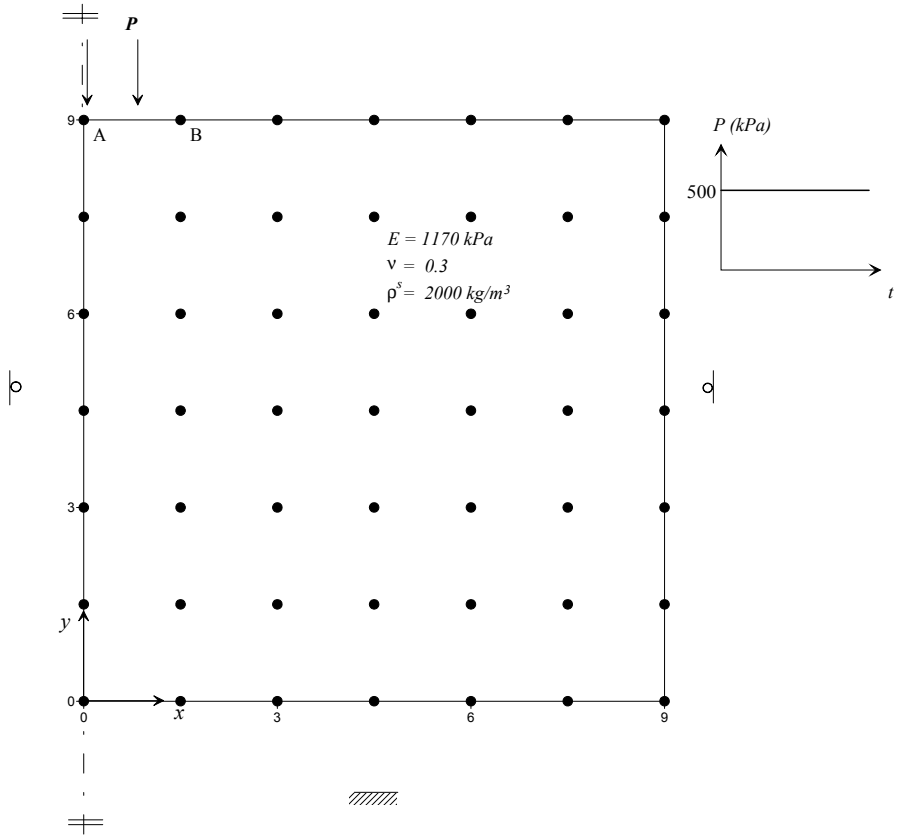



***Fig. 4.21 : Massif 2D monophasique : discrétisation, propriétés et chargement.***

Nous avons tout d'abord réalisé une analyse de sensibilité portant sur l'influence de quelques fonctions de pondération (ou de lissage) usuelles sur les résultats en transformations infinitésimales, pour les méthodes meshfree les plus couramment utilisées en pratique, à savoir les méthodes EFGM et RKPM, ainsi que la méthode RPIM. La comparaison était

chaque fois réalisée avec la méthode FEM, pour les déplacements calculés aux points particuliers A et B du massif (*cf*. Fig. 4.21).

Ici encore, on constate une sensibilité importante des résultats obtenus aux fonctions de pondérations utilisées pour les méthodes EFGM et RPIM. La méthode RKPM fournit dans tous les cas des résultats très éloignés par rapport à ceux fournis notamment par la méthode FEM (*cf*. Fig. 4.22 à Fig. 4.25).

Nous avons ensuite choisi les fonctions de pondérations les plus performantes et nous avons réalisé une analyse en transformations finies. Comme pour le cas monodimensionnel, on constate que pour toutes les méthodes, les déplacements verticaux sont généralement plus faibles en transformations finies par rapport au cas infinitésimal, excepté pour le point situé au centre sous le chargement. Par ailleurs, les déplacements horizontaux sont plus importants en transformations finies (*cf*. Fig. 4.26 à Fig. 4.29).

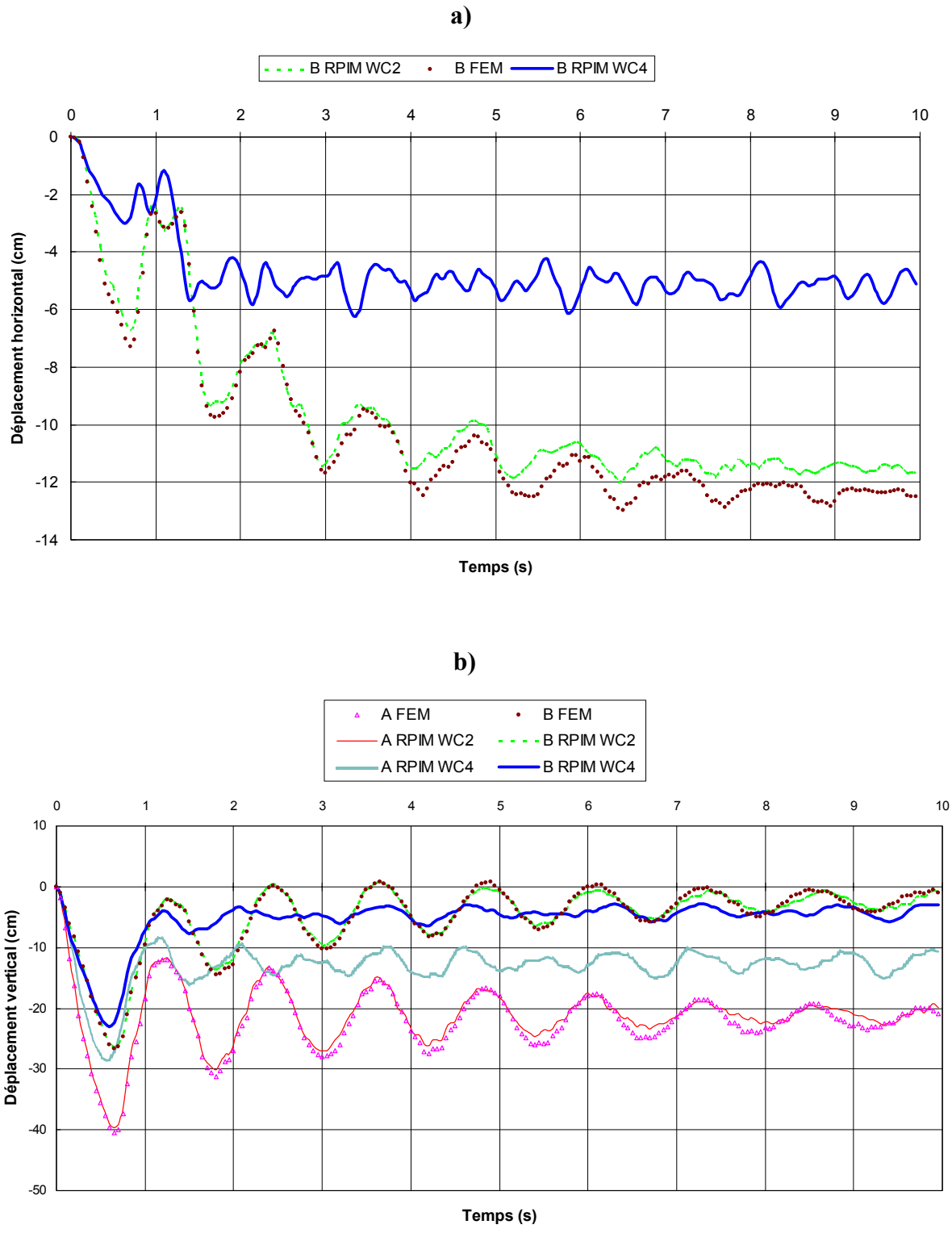
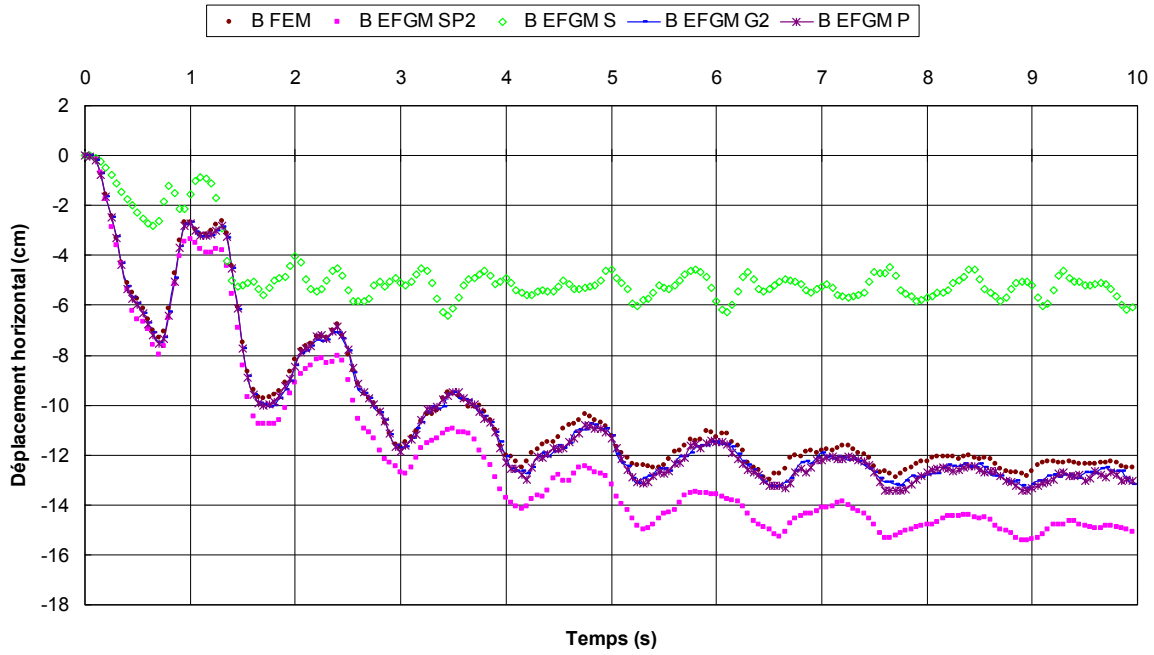



***Fig. 4.22 : Méthodes FEM et RPIM en transformations infinitésimales : comparaison des déplacements calculés aux points A et B, pour des fonctions CSRBF de Wendland $C^2$ (WC2) et $C^4$ (WC4) : a) déplacements horizontaux ; b) déplacements verticaux.***

**a)**




**b)**

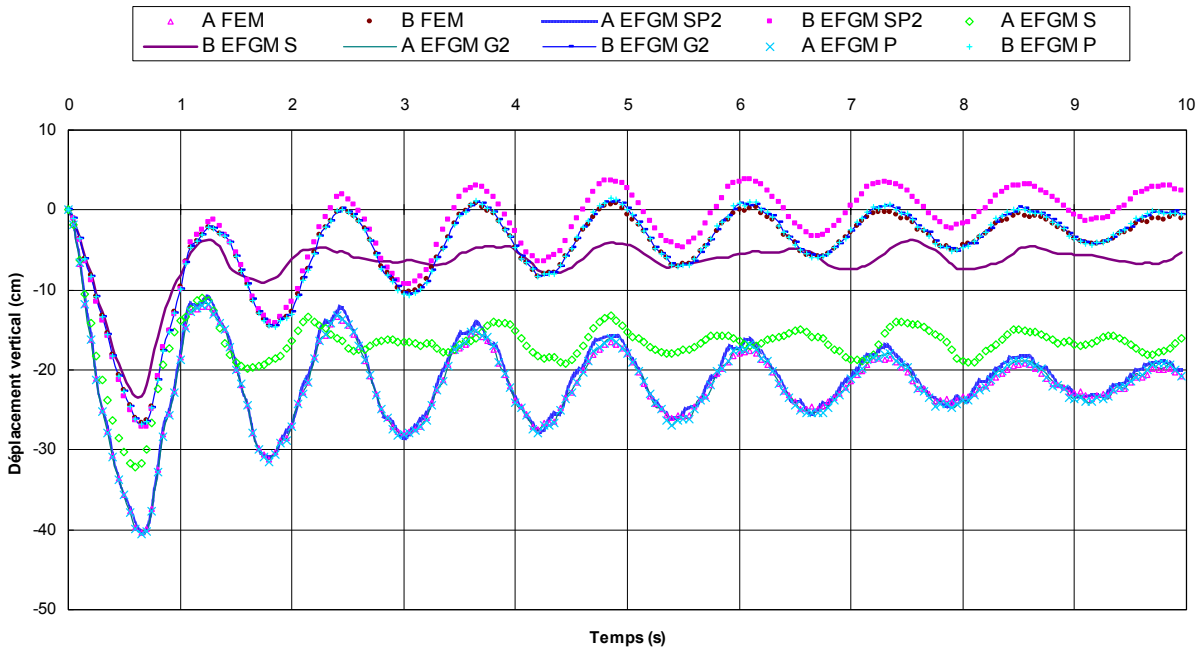
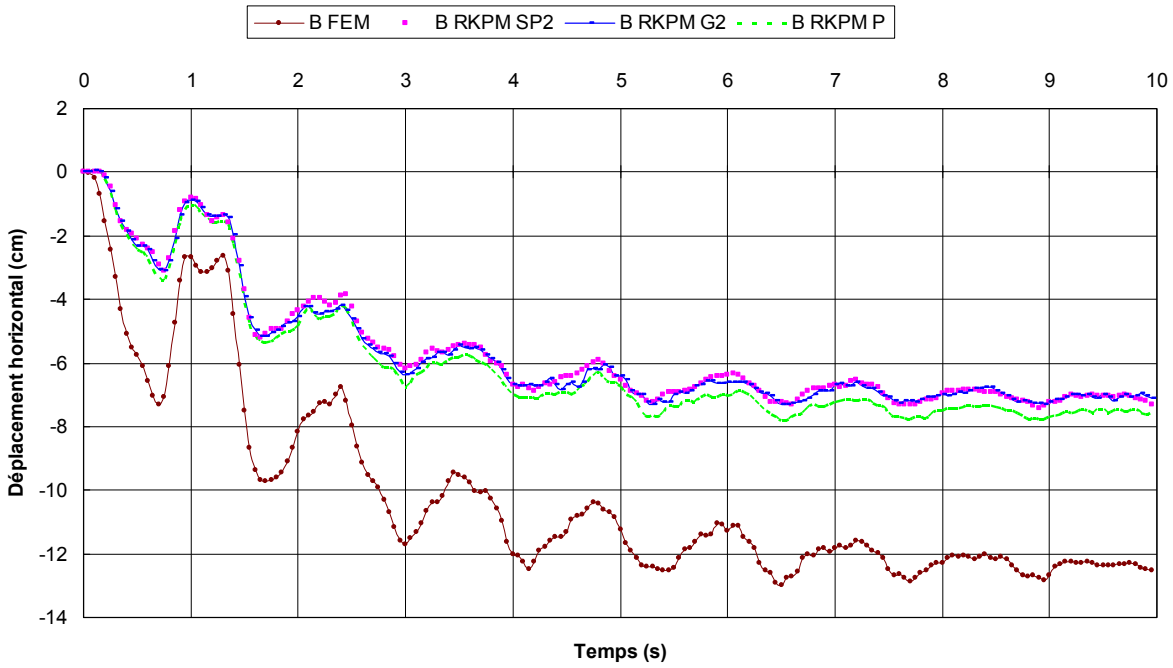



***Fig. 4.23 : Méthodes FEM et EFGM en transformations infinitésimales : comparaison des déplacements calculés aux points A et B, pour des fonctions Spline2 (SP2), Sinus (S), Polynomiale (P) et Gauss2 avec k = 1 et $c_I$ = 1 (G2) : a) déplacements horizontaux ; b) déplacements verticaux.***

**a)**




**b)**

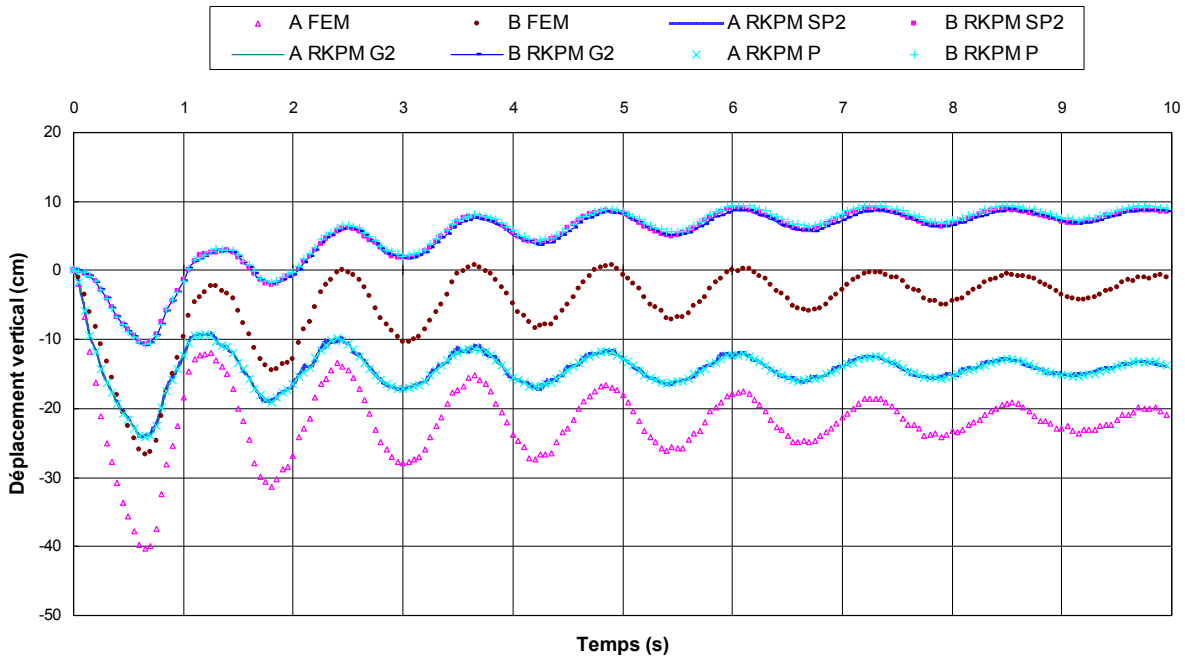
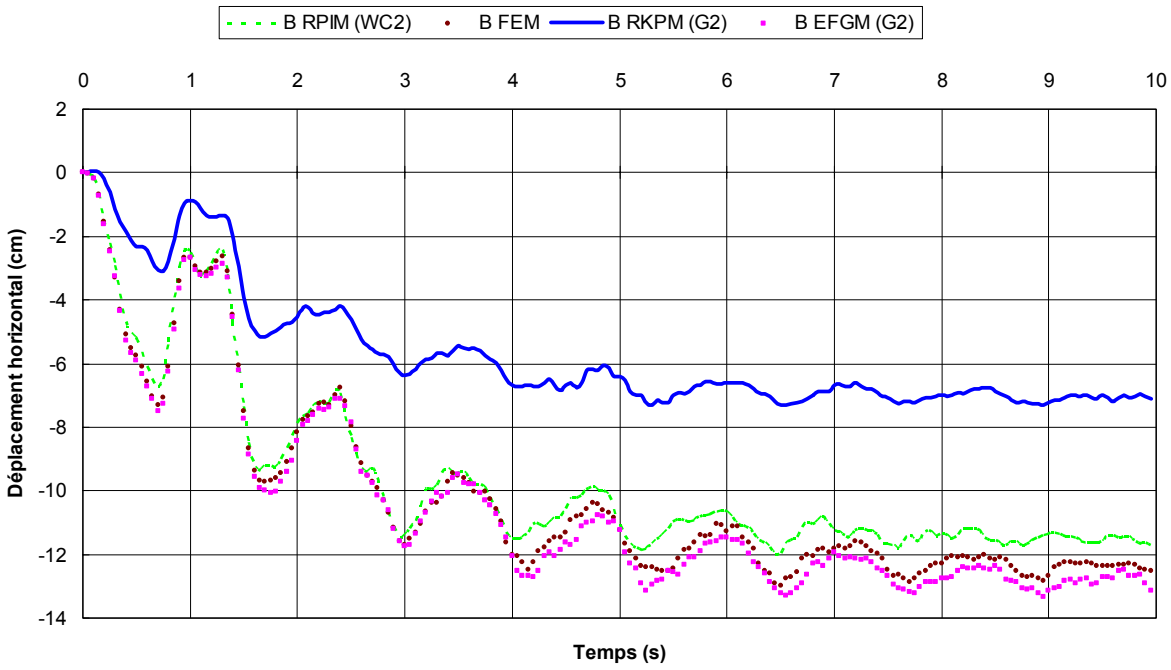



***Fig. 4.24 : Méthodes FEM et RKPM en transformations infinitésimales : comparaison des déplacements calculés aux points A et B, pour des fonctions Spline2 (SP2), Polynomiale (P) et Gauss2 avec k = 1 et $c_1$ = 1 (G2) : a) déplacements horizontaux ; b) déplacements verticaux.***

**a)**




**b)**

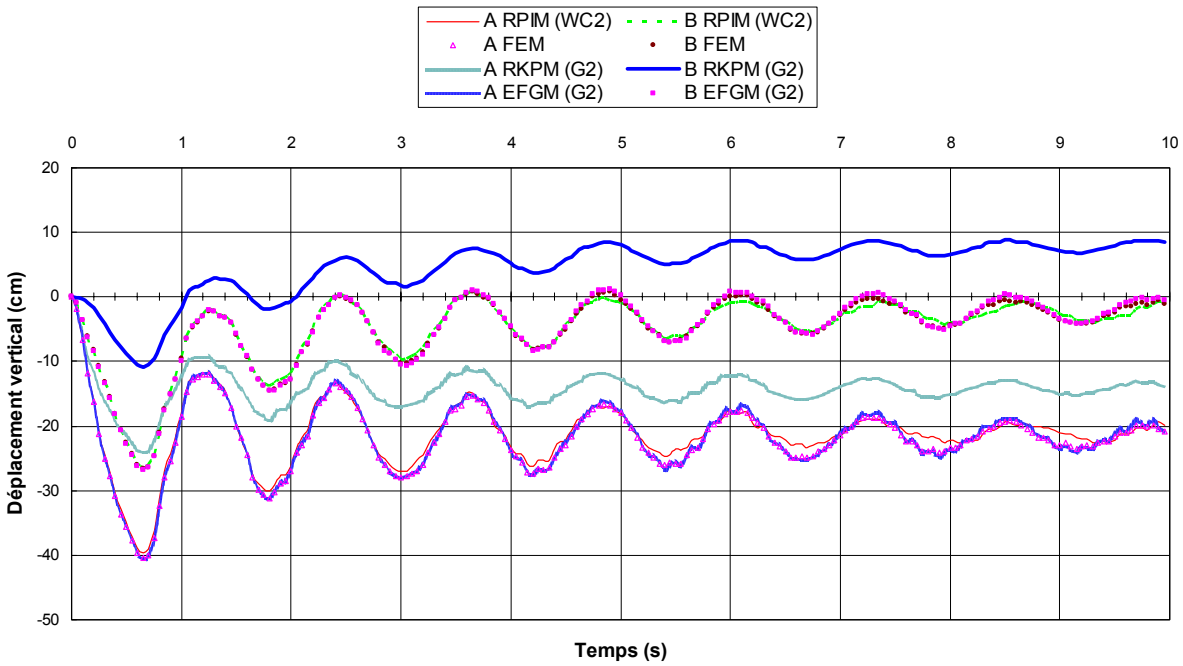
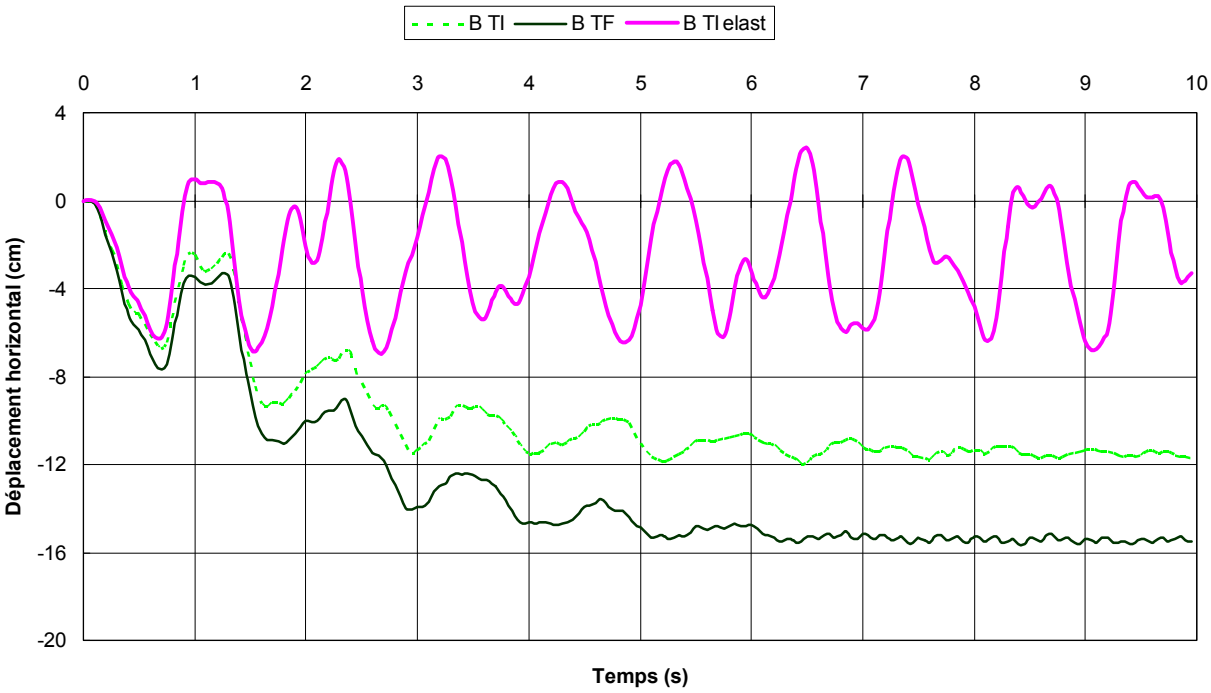



***Fig. 4.25 : Comparaison des déplacements calculés points A et B en transformations infinitésimales avec toutes les méthodes (loi de Drücker-Prager) : a) déplacements horizontaux ; b) déplacements verticaux.***

**a)**




**b)**

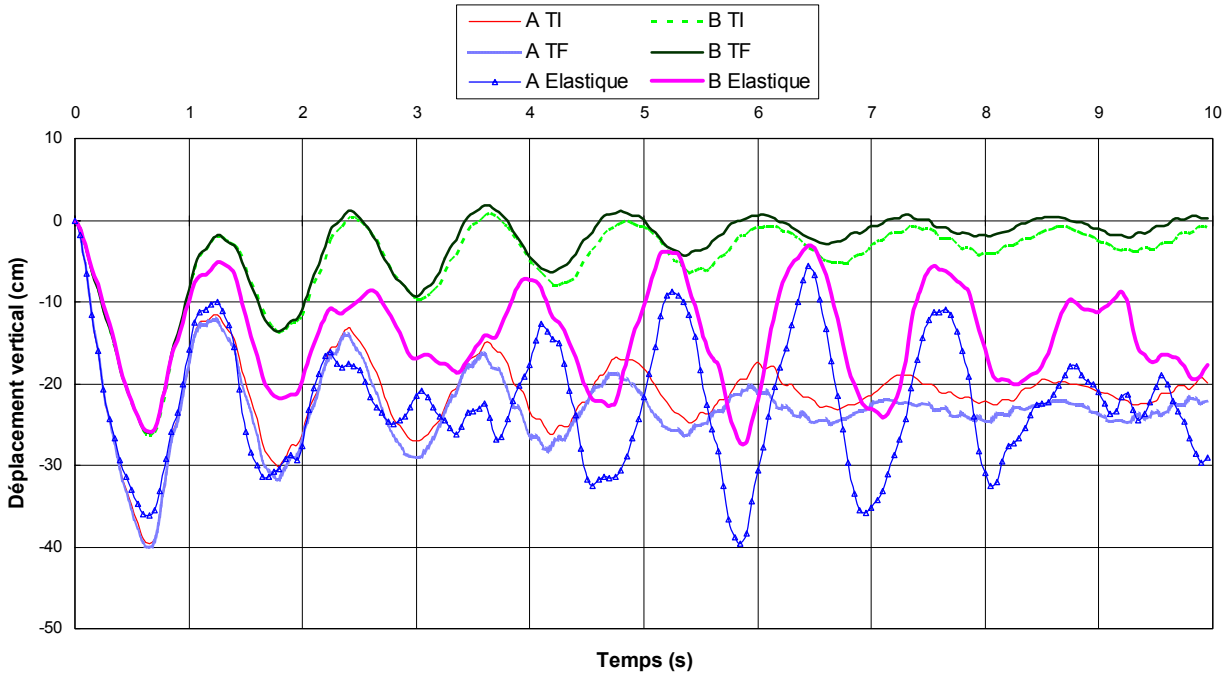
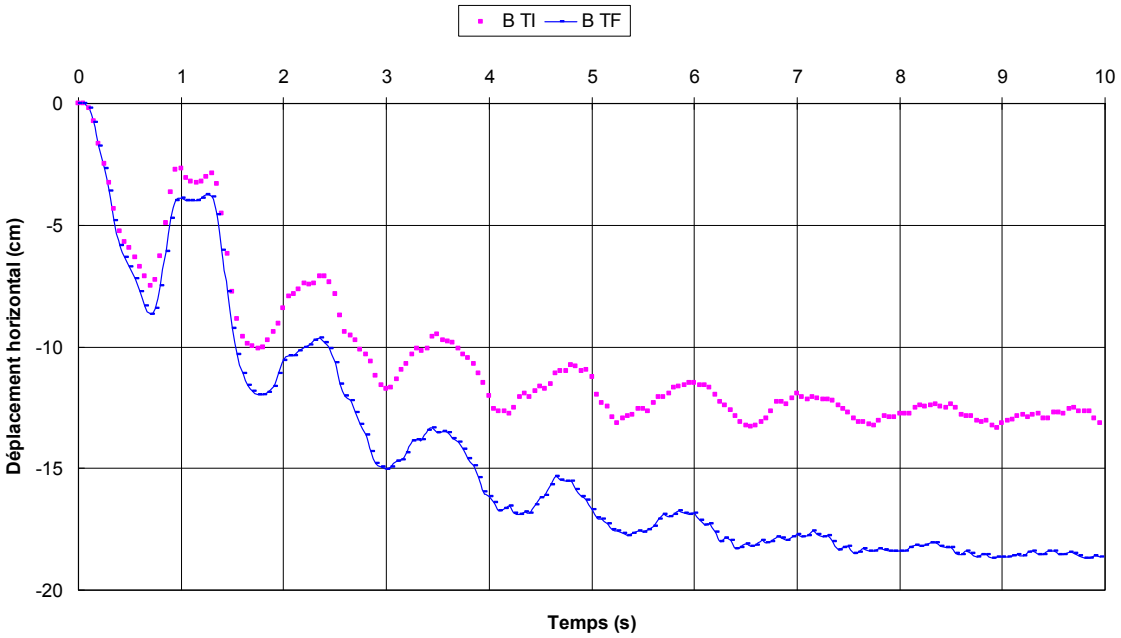



***Fig. 4.26 : Méthode RPIM (fonction WC2) en transformations infinitésimales (TI) et finies (TF) : comparaison des déplacements calculés aux points A et B en élasticité et avec la loi de Drücker-Prager : a) déplacements horizontaux ; b) déplacements verticaux.***

**a)**




**b)**

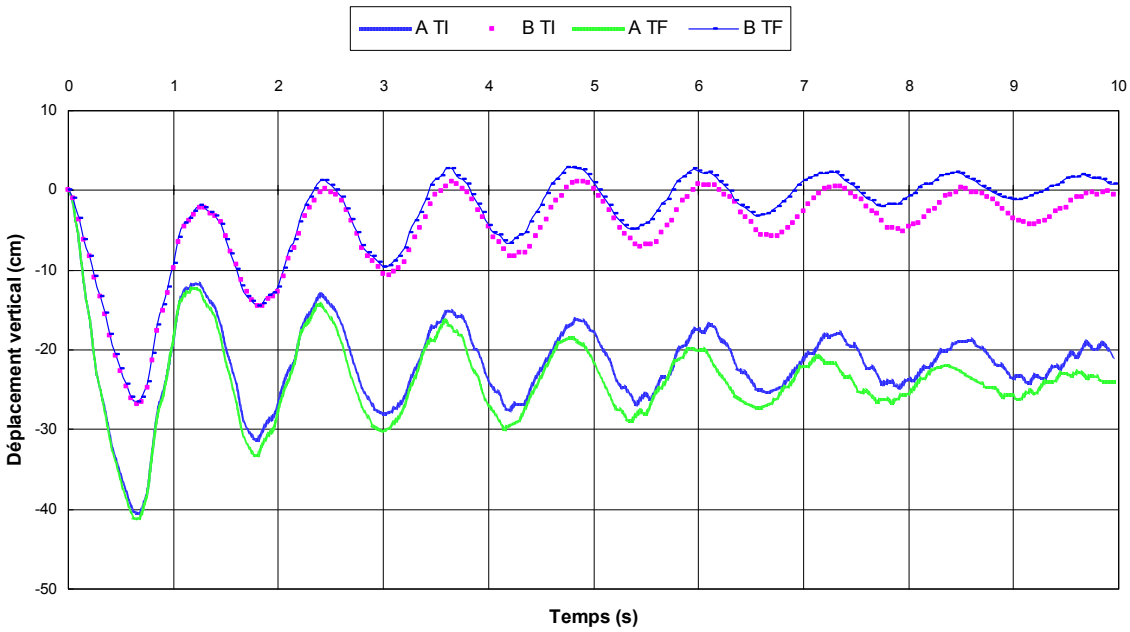
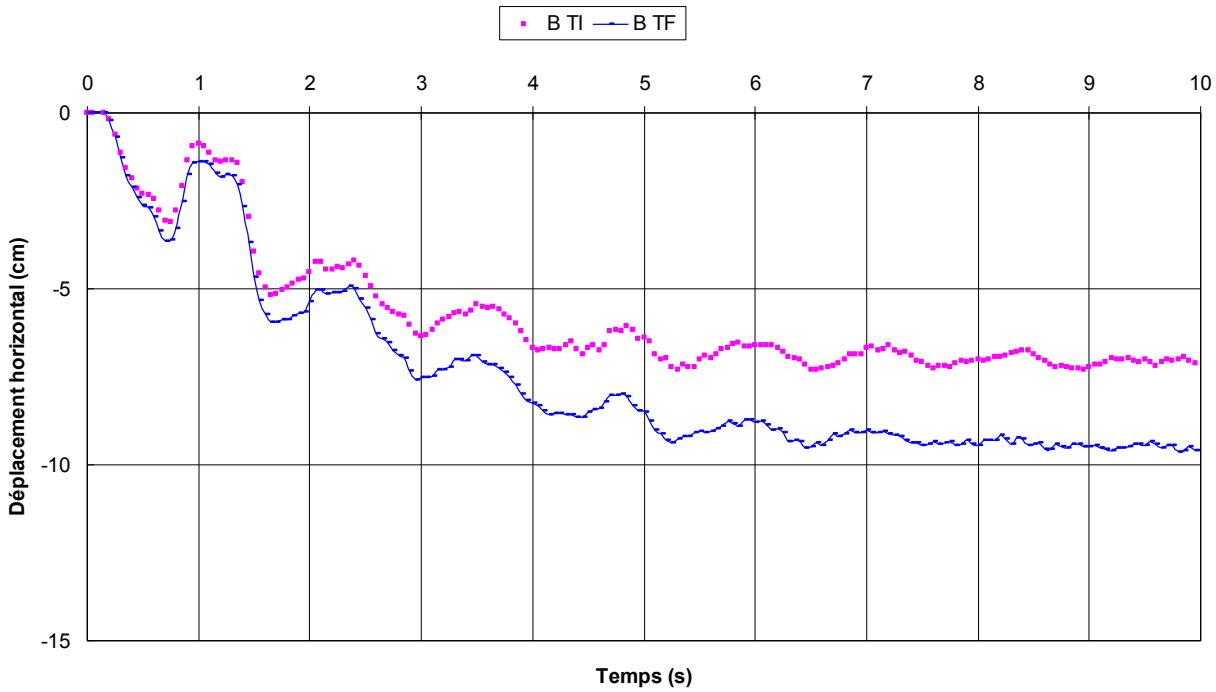



***Fig. 4.27 : Méthode EFGM (fonction G2) en transformations infinitésimales (TI) et finies (TF) : comparaison des déplacements calculés aux points A et B avec la loi de Drücker-Prager : a) déplacements horizontaux ; b) déplacements verticaux.***

**a)**




**b)**

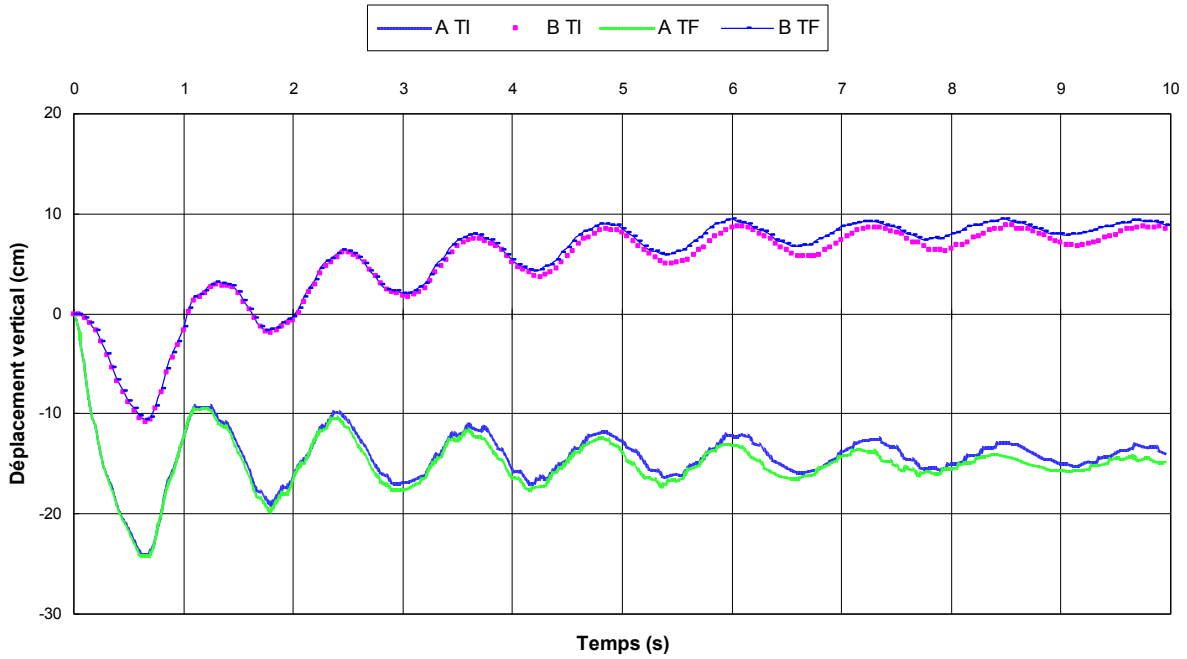
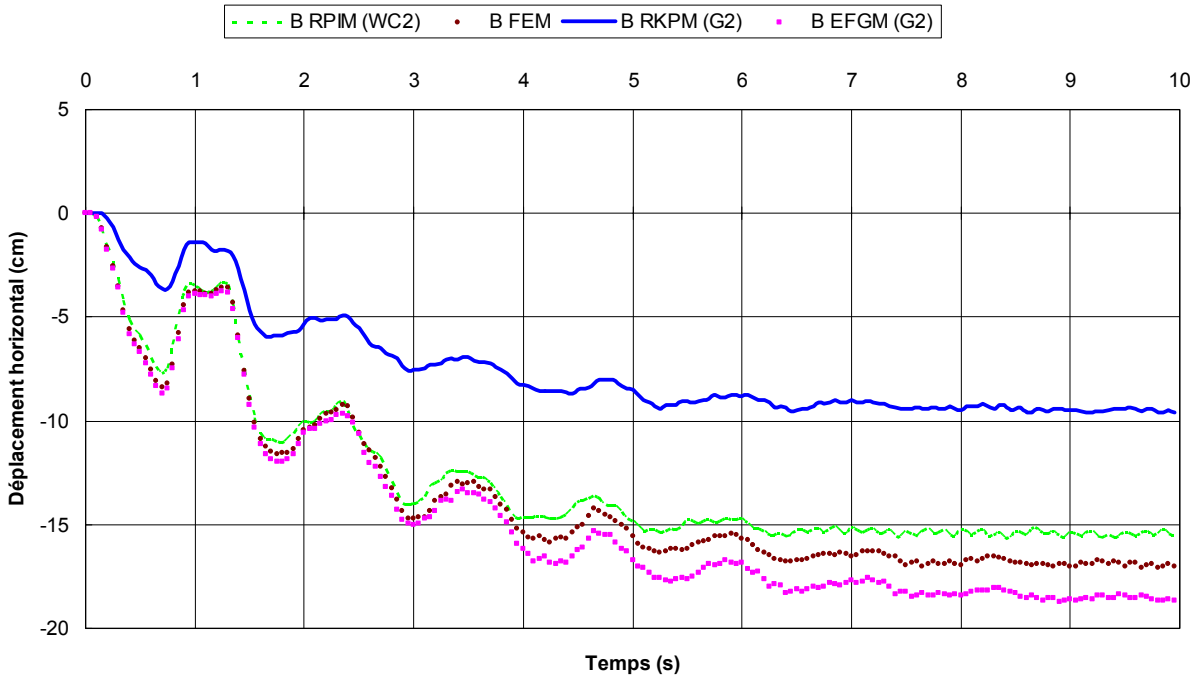



***Fig. 4.28 : Méthode RKPM (fonction G2) en transformations infinitésimales (TI) et finies (TF) : comparaison des déplacements calculés aux points A et B avec la loi de Drücker-Prager : a) déplacements horizontaux ; b) déplacements verticaux.***

**a)**




**b)**

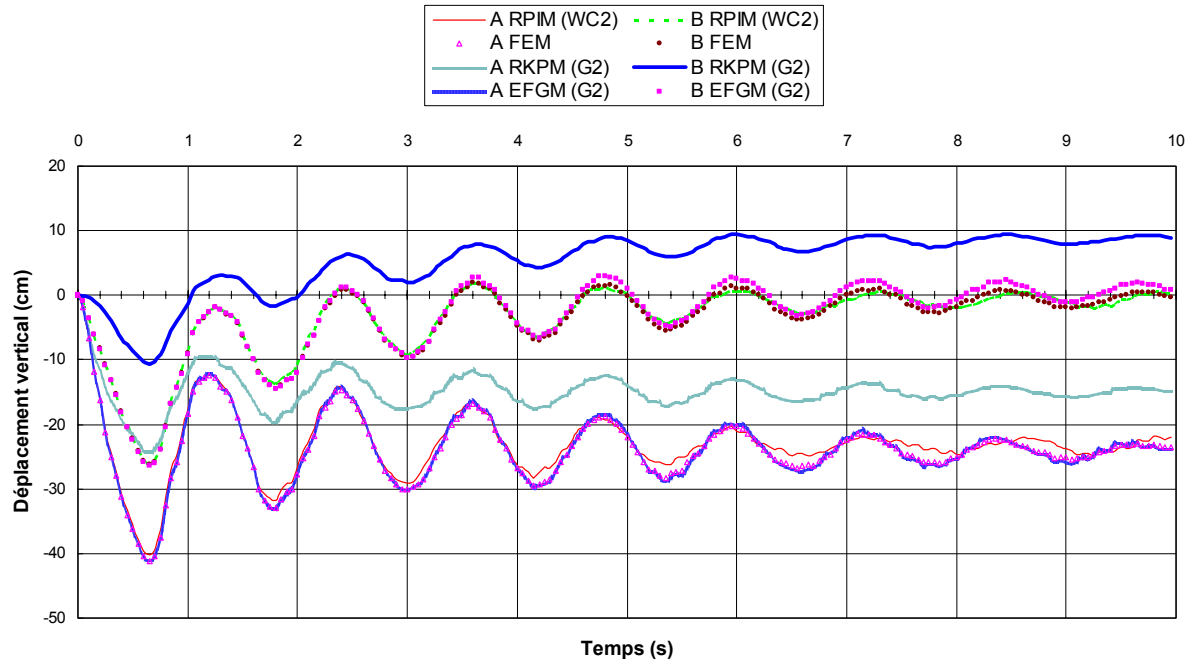



***Fig. 4.29 : Comparaison des déplacements calculés aux points A et B en transformations finies avec toutes les méthodes (loi de Drücker-Prager) : a) déplacements horizontaux ; b) déplacements verticaux.***

## 4.4 REPONSE SISMIQUE DU BARRAGE EN TERRE DE « EL INFIERNILLO »

### 4.4.1 Introduction

L'objectif général de cette section est tout d'abord de valider les développements réalisés dans notre outil de calcul en l'appliquant à l'étude d'un cas réel bien documenté, et ensuite de comparer d'une part les résultats numériques obtenus avec les différentes méthodes numériques utilisées (meshfree et éléments finis), et d'autre part de comparer les performances générales de ces méthodes (temps de calcul, etc.).

Cette section est organisée en deux parties : la première partie résume les caractéristiques et hypothèses générales adoptées pour réaliser les simulations ; la seconde présente l'analyse des résultats et des performances numériques obtenues pour les différentes méthodes utilisées.

### 4.4.2 Caractéristiques et hypothèses générales des simulations

#### *4.4.2.1 Le barrage*

Le barrage de « *El Infiernillo* » (Mexique) est un barrage en terre constitué de quatre zones différentes, comprenant un noyau (matériau 1) formé de silt sableux compact et de plasticité moyenne (indice de plasticité $IP$ = 26%, limite de liquidité $w_L$= 45%), deux zones de transition latérales (matériaux 2 et 3) et enfin une zone d'enrochements (matériau 4).

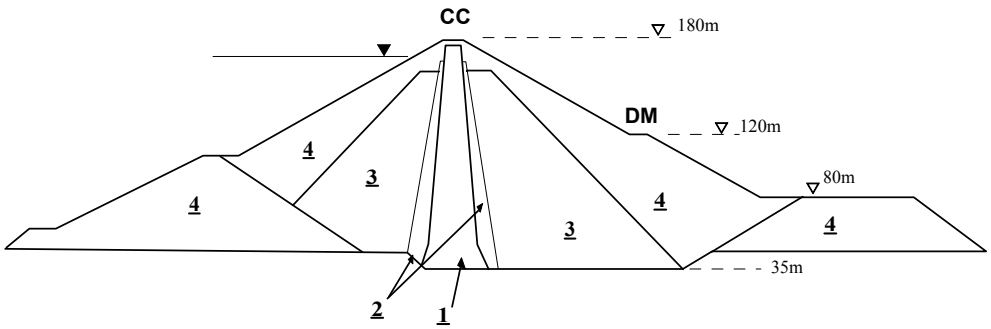



***Fig. 4.30: Section maximale du barrage de « El Infiernillo ».***

#### *4.4.2.2 Le mouvement d'entrée*

Ce barrage est situé dans une région à forte sismicité et a de ce fait, subit de nombreux séismes depuis sa construction, et notamment celui du 14/3/1979, évènement fort (magnitude 7.6 sur l'échelle de Richter) caractérisé par une distance épicentrale de 134 Km. Une accélération horizontale maximale de 0.1g (PGA) a été enregistrée dans la partie gauche du barrage (station INCM), et la fréquence dominante était d'environ 2.3Hz.

Par ailleurs, les enrochements étant de même nature que le substratum rocheux (fondation du barrage), il a été possible de les considérer comme rocher affleurant pour les simulations

réalisées, et donc d'utiliser directement l'accélération enregistrée comme mouvement d'entrée à la base du barrage.

Le mouvement d'entrée utilisé pour les simulations comprenait deux composantes identiques, l'une horizontale et l'autre verticale, situées dans le plan d'analyse du barrage et correspondant à l'accélération enregistrée à la station INCM.

#### *4.4.2.3 Caractéristiques numériques*

Afin d'être capable de reproduire le comportement non linéaire du sol sous sollicitation sismique, nous avons implanté le modèle élastoplastique multi-mécanismes de Hujeux dans le logiciel *MoveFree*. Ce modèle permet entre autre de considérer un écrouissage cinématique, et est caractérisé par trois mécanismes de déformations plastiques déviatoires pouvant être mobilisés dans trois plans orthogonaux, et un mécanisme de consolidation isotrope. Ces quatre mécanismes peuvent être activés aussi bien lors de chargements monotones (ou primaires), que cycliques.

Pour réaliser les simulations dynamiques, il a été nécessaire d'initialiser l'histoire des matériaux constituant le barrage (variables internes et contraintes initiales). Pour ce faire, un calcul statique 2D préalable en transformations infinitésimales a été effectué avec le logiciel GEFDYN[1], afin de simuler les diverses étapes de la construction du barrage. Pour ce premier calcul, on a supposé que les matériaux 1 à 3 étaient partiellement drainés, tandis que le matériau 4 était totalement drainé (monophasique). Les caractéristiques complètes de ce calcul sont détaillées par Sica *et al.* [SIC 02].

Les simulations meshfree de la réponse sismique du barrage ont ensuite été réalisées en transformations finies, en repartant de l'état de contraintes et des variables internes issus du calcul statique précédent. Pour ces simulations, nous avons fait l'hypothèse que le barrage était cette fois complètement drainé. Par ailleurs, les caractéristiques du modèle élastoplastique pour les différents matériaux sont regroupées dans le tableau 4.1 ci-après. Nous avons également considéré que les matériaux étaient de type hypoélastique.

La discrétisation spatiale utilisée comprenait 384 nœuds (*cf.* Fig. 4.31). La structure d'intégration de type maillage était constituée quant à elle de 389 éléments, dont 311 quadrangles et 78 triangles (*cf.* Fig. 4.32). 3x3 points de Gauss par élément ont été utilisés pour l'intégration. Du fait de la structure d'intégration considérée, nous avons utilisé des domaines d'influence de type radial avec un rayon variable au minimum égal à 2m. Nous avons par ailleurs utilisé une base polynomiale linéaire, des fonctions de pondération polynomiales modifiées pour les méthodes EFGM et RKPM (interpolation globale sur le domaine), et une fonction CSRBF de Wendland ($C^2$) pour la méthode RPIM. Le pas de temps était de 0.02 seconde pour une durée totale de séisme égale à 20 secondes. Enfin, une base rigide a été considérée pour toutes les simulations réalisées.

[1] Logiciel aux éléments finis développé par l'Ecole Centrale de Paris.

| Paramètres | *Symbole* | Noyau | Filtres | Enrochements |
|---|---|---|---|---|
| *Elastiques* | $n$ | 0.6 | 0.6 | 0.3 |
| | $K_i$ *(MPa)* | 27.5 | 55 | 48.4 |
| | $G_i$ *(MPa)* | 10 | 20 | 16.5 |
| | $p_{ref}$ *(MPa)* | 1 | 1 | 1 |
| *Plastiques et état critique* | $p_{ci}$ *(MPa)* | 0.45 | 1.2 | 1.6 |
| | $b$ | 0.9 | 0.12 | 0.12 |
| | $\beta$ | 33 | 55 | 52 |
| | $\varphi$ | 25 | 35 | 43 |
| | $\psi$ | 25 | 35 | 40 |
| *Ecrouissage déviatoire* | $a_{mon}$ | 0.001 | 0.005 | 0.012 |
| | $a_{cyc}$ | 0.0005 | 0.0025 | 0.006 |
| *Ecrouissage isotrope* | $d$ | 2 | 2 | 2 |
| | $c_{mon}$ | 0.1 | 0.004 | 0.004 |
| | $c_{cyc}$ | 0.05 | 0.002 | 0.002 |
| *Limites de domaine de comportement* | $r_{el}$ | 0.0001 | 0.02 | 0.02 |
| | $r_{hys}$ | 0.002 | 0.04 | 0.04 |
| | $r_{mob}$ | 0.02 | 0.8 | 0.8 |
| | $r_4^{el}$ | 0.0003 | 0.001 | 0.001 |
| | $m$ | 1 | 1.1 | 1.2 |

***Tabl. 4.1 : Paramètres de la loi de Hujeux pour les matériaux constitutifs du barrage (d'après Sica et al. [SIC 02])***

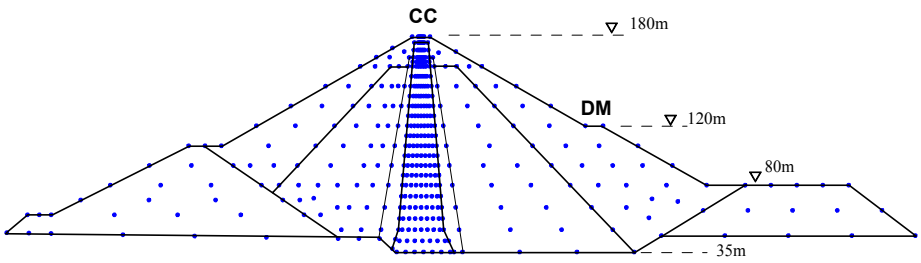



***Fig. 4.31 : Barrage de « El Infiernillo » : arrangement nodal.***

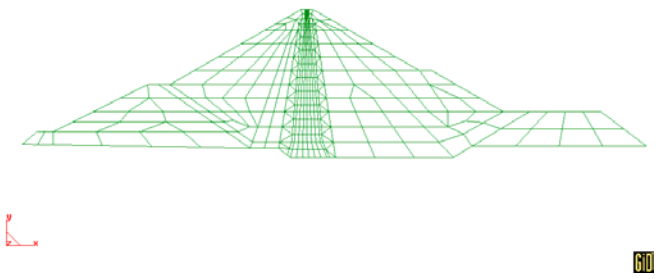


***Fig. 4.32 : Barrage de « El Infiernillo » : structure d'intégration de type maillage.***

### 4.4.3 Résultats

En comparant les valeurs de déplacements observées en crête (point CC) et à l'aval du barrage (point DM), avec les résultats issus des simulations avec les méthodes RKPM, EFGM, RPIM et FEM, on constate généralement une bonne concordance au niveau des valeurs finales de déplacement vertical calculées, et une surestimation du mouvement horizontal (*cf.* Fig. 4.33 et Fig. 4.34).

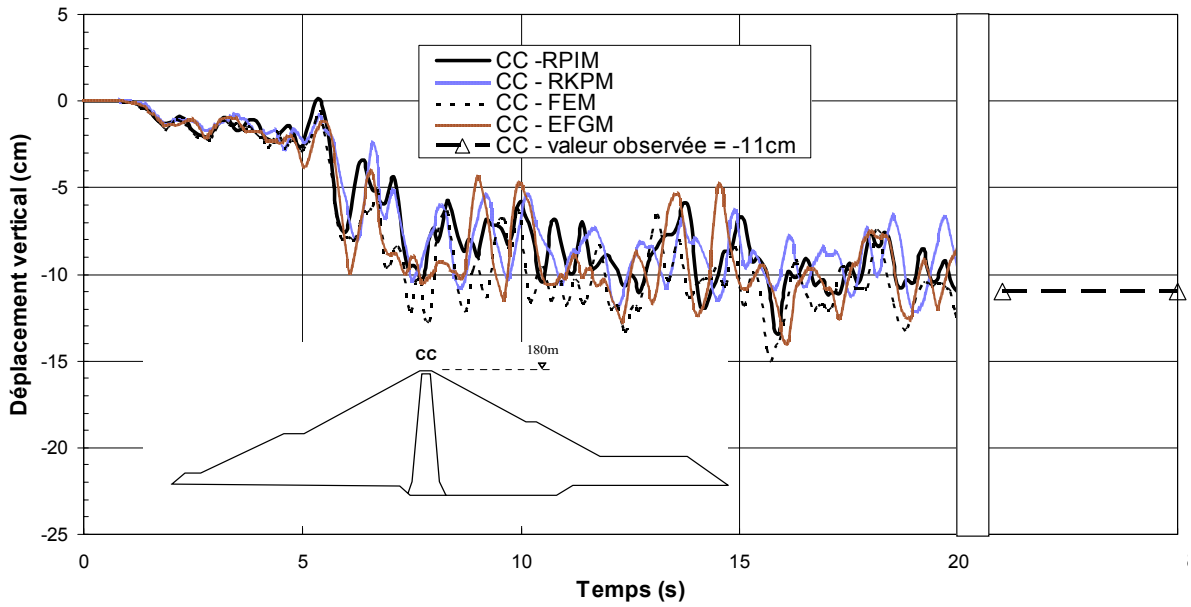



*Fig. 4.33 : Déplacements verticaux calculés et observés en crête.*

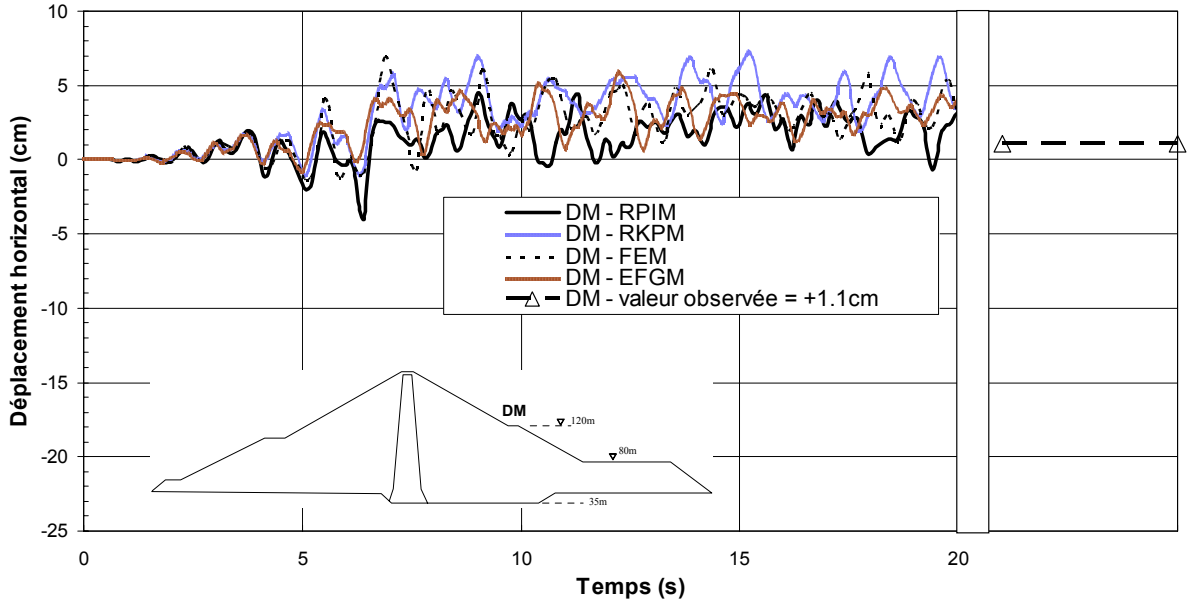



*Fig. 4.34 : Déplacements horizontaux calculés et observés en aval du barrage (point DM).*

Si l'on regarde les vecteurs déplacements calculés en fin de séisme avec les différentes méthodes (*cf.* Fig. 4.35 à Fig. 4.38), on constate une rupture de type circulaire en pied de barrage (côté DM), plus marquée pour les méthodes RPIM et FEM, que pour les autres méthodes.

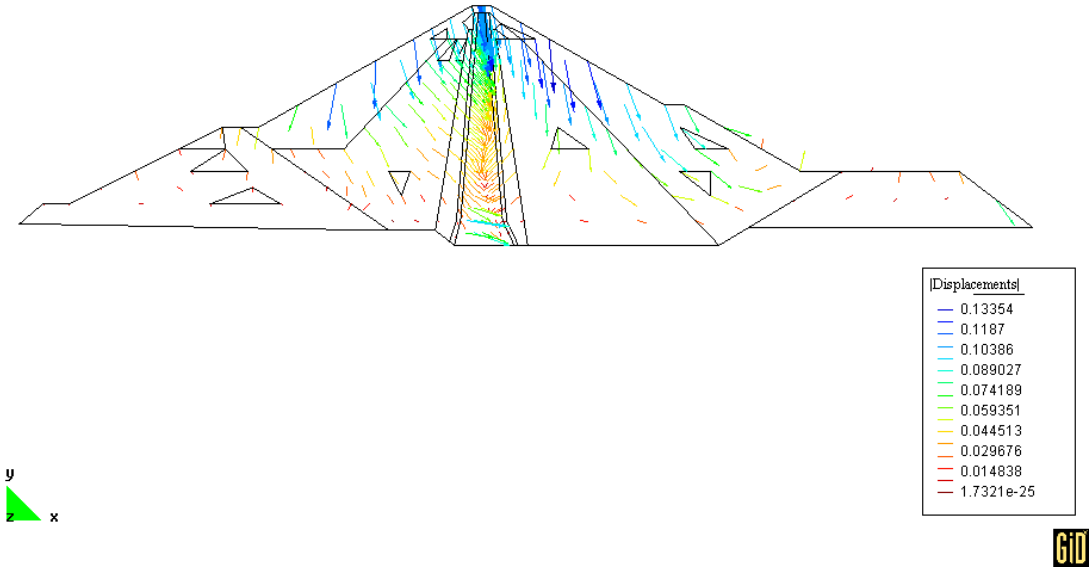



***Fig. 4.35 : Déplacements calculés en fin de séisme avec la méthode RPIM (loi de Hujeux en transformations finies).***

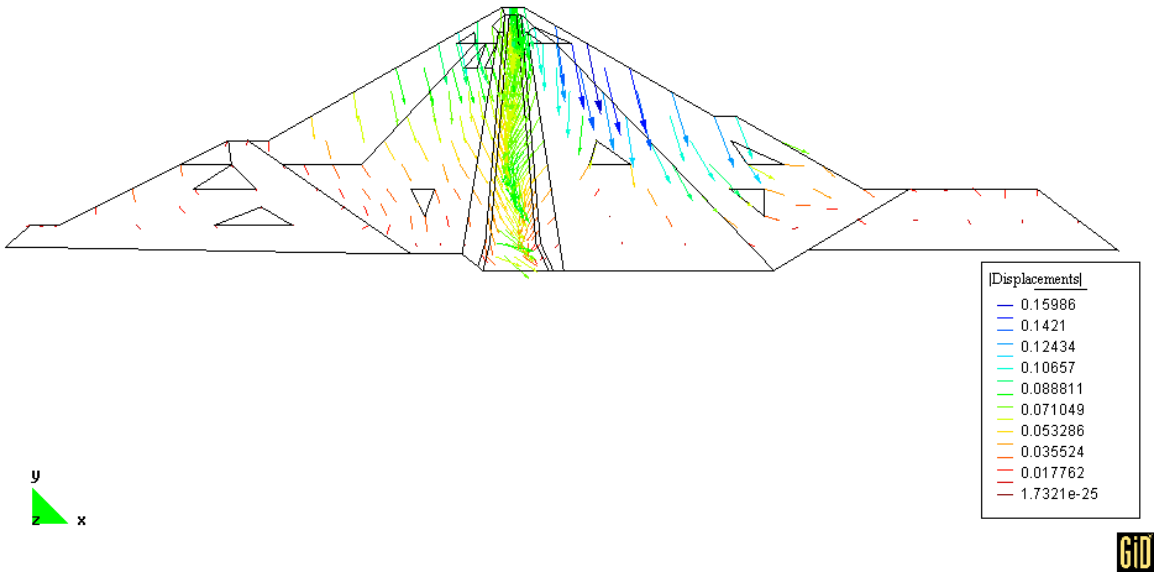



***Fig. 4.36 : Déplacements calculés en fin de séisme avec la méthode EFGM (loi de Hujeux en transformations finies).***

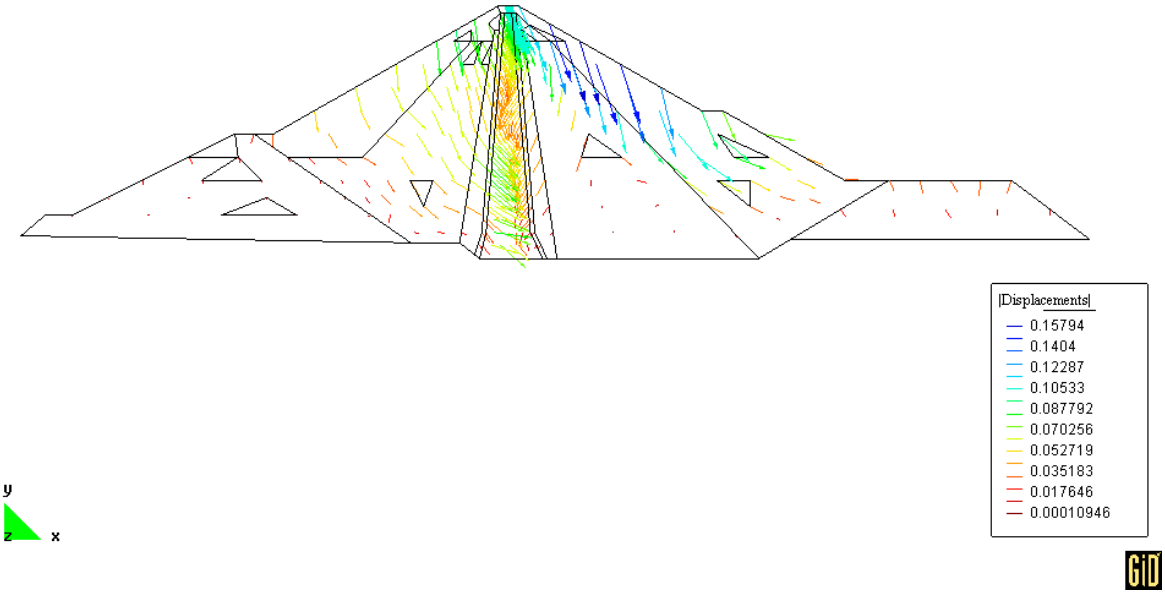



*Fig. 4.37 : Déplacements calculés en fin de séisme avec la méthode RKPM (loi de Hujeux en transformations finies).*

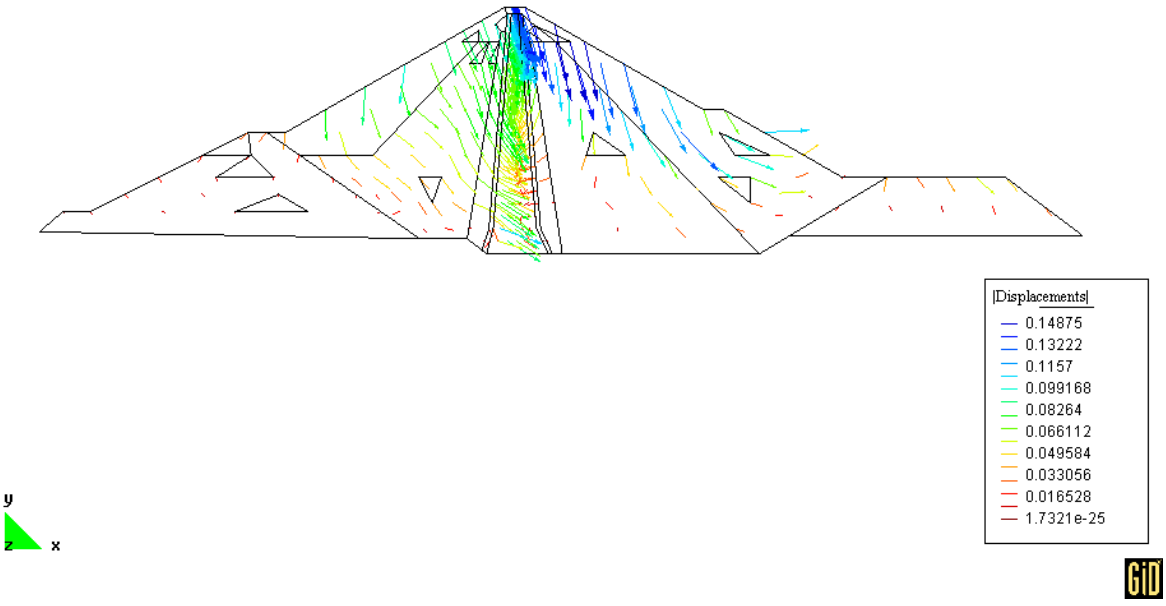



*Fig. 4.38 : Déplacements calculés en fin de séisme avec la méthode FEM (loi de Hujeux en transformations finies).*

A titre indicatif, nous fournissons ci-après les temps de calcul obtenus dans en utilisant les différentes méthodes dans le cadre des transformations finies, avec un processeur AMD Athlon à 1.2GHz fonctionnant sous Windows XP et 512Mo de RAM (*cf.* Tabl. 4.2). Ces temps comprennent la préparation des données, à savoir notamment la recherche des nœuds voisins et la création des domaines d'influence pour les méthodes meshfree.

| Méthode numérique | Temps CPU (s) | Surcoût par rapport à RPIM |
|---|---|---|
| RPIM | 25 670 | 0% |
| RKPM | 26 612 | 3.7% |
| EFGM | 32 718 | 27.5% |
| FEM | 30 700 | 19.6% |

***Tabl. 4.2 : Temps de calculs comparés pour les différentes méthodes numériques en transformations finies.***

La méthode RPIM est nettement plus rapide que toutes les autres méthodes. A notre avis, elle est tout d'abord plus rapide que les méthodes RKPM et EFGM au niveau du calcul des fonctions de forme qui sont interpolantes par construction. En effet, le calcul des fonctions de pondérations modifiées est relativement coûteux dans ce cas. Par ailleurs, on constate que la méthode FEM est l'une des méthodes les plus lentes ici. Les diverses méthodes produisant des valeurs de fonctions de forme et de dérivées de ces fonctions différentes aux points d'intégration, les surcoûts importants en temps peuvent également provenir de l'intégration de la loi de comportement non linéaire (niveaux de déformations différents entre les méthodes).

Enfin, le contenu fréquentiel de l'accélération horizontale calculée en crête du barrage (point CC) avec les méthodes RPIM, RKPM et FEM (*cf.* Tabl. 4.3 et Fig. 4.39) montre un décalage en fréquence du pic d'accélération calculé par rapport à celui de la réponse enregistrée, le décalage étant moins marqué pour la méthode RKPM. La méthode EFGM quant à elle, produit sur ce test, une amplification trop importante du mouvement et en plus du pic enregistré, elle génère deux pics supplémentaires, l'un autour de 0.4Hz et l'autre, vers 6Hz . Le décalage en fréquence constaté pour toutes les méthodes est probablement dû à notre hypothèse de milieu complètement drainé. En effet, la présence de l'eau en réalité confère un comportement non drainé au noyau du barrage (matériau peu perméable), qui tend à diminuer la vitesse réelle des ondes de cisaillement et donc à décaler le contenu fréquentiel de la réponse vers les basses fréquences.

| Méthode numérique | Fréquence (Hz) | PGA - Point CC (g) |
|---|---|---|
| Enregistré | 1.3 | 1.9 |
| RPIM | 2.7 | 1.6 |
| RKPM | 2.1 | 1.4 |
| EFGM | 0.4<br>1.5<br>5.6 | 3.3<br>2.5<br>3.2 |
| FEM | 3.5 | 1.9 |

***Tabl. 4.3 : Pics d'accélération enregistrés et calculés en crête du barrage à la fin du séisme.***

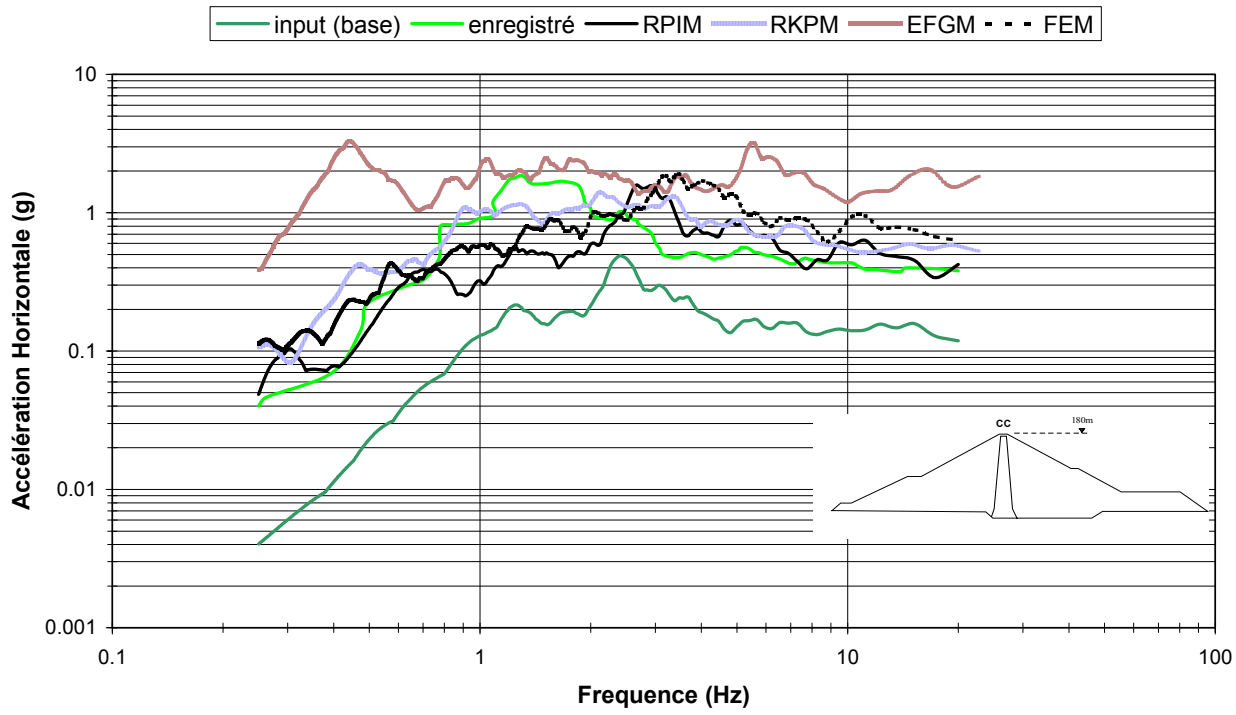



***Fig. 4.39 : Spectres de réponse en accélération : accélérations enregistrée et calculées en crête et mouvement d'entrée à la base du barrage.***

## 4.5 CONCLUSIONS

Ce chapitre nous a permis de montrer les possibilités de notre outil de calculs *MoveFree*, en l'appliquant à un certain nombre de problèmes géomécaniques en transformations infinitésimales et finies, pour des cas statiques ou dynamiques.

Pour le calcul de la réponse sismique du barrage en terre de « El Infiernillo », nous avons préféré utiliser uniquement la technique d'interpolation globale (sur tout le domaine), celle-ci étant peut-être plus coûteuse en temps de préparation des données (construction des fonctions de forme) par rapport au cas non interpolant utilisant un couplage FEM à la frontière, mais elle permet ensuite d'accélérer les procédures de sauvegarde des résultats pour l'ensemble des nœuds de la discrétisation.

Nous avons par ailleurs constaté une sensibilité importante des méthodes meshfree testées, notamment EFGM et RKPM, aux fonctions de pondération (ou lissage) utilisées pour construire les fonctions de forme. Ceci rend un calcul meshfree sur cas réel difficile, car nous avons vu que pour une application simple, le choix de certaines fonctions conduit à des résultats très éloignés de ce à quoi l'on peut s'attendre.

# Conclusion Générale

Etant donné le nombre important de méthodes meshfree actuellement à la disposition des ingénieurs pour réaliser leurs simulations numériques, il nous est apparu indispensable de réaliser une synthèse des principales méthodes existantes, afin de fournir une écriture commune à toutes ces méthodes et de mettre ainsi en évidence les spécificités de chacune. Pour ce faire, nous avons distingué cinq grandes familles de méthodes meshfree dont la formulation variationnelle repose sur une approche de type Galerkin :

- les méthodes *MLS*, construites à partir d'une approximation par moindres carrés mobiles, à savoir essentiellement les méthodes DEM, EFGM et h-p Clouds ;

- les méthodes *RKM* ou de particules avec notamment les méthodes SPH (*Smooth Particle Hydrodynamics*), RKPM (*Reproducing Kernel Particle Method*), RKI (*Reproducing Kernel Interpolation*) et MLS/RK (*Moving Least Square/Reproducing Kernel*) ;

- les méthodes basées uniquement sur le concept de partition de l'unité, telle que la méthode C-PUM (*Corrected Partition of Unity Method*) ;

- les méthodes construites à partir d'une interpolation purement polynomiale telle que la méthode PIM (*Point Interpolation Method*) ;

- et enfin les méthodes mixtes couplant des fonctions de bases radiales, soit avec une approximation par moindres carrés mobiles, telle que la méthode MLS/RBF (*Moving Least Square/Radial Basis Function*), soit avec une interpolation polynomiale (méthode *Radial* PIM).

Toutes ces méthodes, ainsi que la méthode des éléments finis, ont été implémentées dans l'outil numérique *MoveFree*, logiciel de simulations géomécaniques multidimensionnelles, développé en langage orienté objets (C++), spécifiquement dans le cadre de cette étude.

Comme nous l'avons vu au niveau du premier chapitre de ce travail, la différence principale entre toutes ces méthodes réside dans la construction des fonctions de forme et leur capacité qui en résulte, à être ou non interpolantes.

Pour les méthodes dont les fonctions ne sont pas interpolantes, il est alors nécessaire de mettre en œuvre une procédure spécifique pour pouvoir imposer des conditions aux limites essentielles. Deux procédures très utilisées consistent soit à introduire des multiplicateurs de Lagrange disposés le long de la frontière concernée, soit à réaliser un couplage local avec des éléments finis généralement de volume. Pour notre part, nous avons mis en œuvre cette dernière technique en utilisant des éléments finis linéiques (ou surfaciques en 3D). Les tests numériques réalisés montrent que ce type d'éléments fournis des résultats souvent meilleurs que les éléments de volume. Nous avons également choisi de tester une technique permettant d'obtenir une interpolation globale sur le domaine d'étude, cette technique nécessitant de

modifier les fonctions de pondération (ou de lissage) utilisées pour construire les fonctions de forme.

D'une manière générale, nous avons constaté une sensibilité importante des méthodes meshfree testées aux fonctions de pondération (ou lissage) utilisées pour construire les fonctions de forme. Ceci rend un calcul meshfree sur cas réel difficile, car nous avons vu que pour une application simple, le choix de certaines fonctions peut conduire à des résultats très éloignés de ce à quoi l'on peut s'attendre.

Nous avons également abordé les difficultés pratiques qui se posent lorsque l'on cherche à implémenter les méthodes meshfree, difficultés qui concernent essentiellement la détermination et éventuellement la gestion des domaines d'influence des nœuds de la discrétisation, ainsi que l'intégration spatiale des équations aux dérivées partielles. Nous avons vu en effet dans les exemples numériques du second chapitre, que la taille et la forme du domaine d'influence sont des paramètres essentiels au niveau de la convergence et de la précision numérique qu'il est possible d'atteindre. Notamment, l'utilisation d'un support rectangulaire permet généralement d'améliorer les performances numériques des différentes méthodes meshfree testées.

Nous avons ensuite vu qu'il existe différentes méthodes possibles pour déterminer les intégrales de volume de la formulation variationnelle, à savoir une intégration nodale ou de type quadrature de Gauss-Legendre avec utilisation de grilles déconnectées ou non des nœuds de la discrétisation. La première technique permet d'obtenir une méthode réellement meshfree, mais nécessite cependant d'utiliser des procédures de correction spécifiques, afin de maintenir un niveau de précision numérique acceptable, ces procédures étant alors très gourmandes en temps CPU. De plus, la précision ainsi obtenue demeure généralement inférieure à celle qu'il est possible d'atteindre avec une intégration de type Gauss.

Un autre aspect important de notre travail consistait à établir la formulation variationnelle du modèle dynamique non linéaire adapté aux milieux poreux saturés en transformations finies. Ce modèle a été implanté dans le logiciel *MoveFree* et testé sur un certain nombre d'applications géomécaniques 2D. En général, nous avons constaté une meilleure convergence des méthodes meshfree pour les calculs non linéaires réalisés en transformations finies, par rapport à la méthode des éléments finis, pour des temps de calculs très variables suivant les méthodes meshfree utilisées. Par ailleurs, la description lagrangienne totale a été choisie pour la mise en œuvre pratique dans *MoveFree*, afin de limiter les temps de calculs nécessaires pour construire les fonctions de forme. En effet, toute autre description (par exemple, lagrangienne réactualisée) nécessiterait une mise à jour des domaines d'influence et donc la reconstruction des fonctions de forme à chaque pas du processus itératif.

Concernant la loi de comportement de la phase solide, deux modèles élastoplastiques sont actuellement disponibles dans le logiciel : le modèle Drücker-Prager et la loi de Hujeux multimécanismes. En transformations finies, seule l'approche hypoélastique a été mise en œuvre et testée, qui conduit à une écriture incrémentale de la loi. L'autre approche usuelle consiste à se placer dans le cadre de la Thermodynamique des Processus Irréversibles en considérant la phase solide comme un matériau hyperélastique et en exprimant la loi à partir des densités volumiques d'énergie libre et de dissipation intrinsèque.

Finalement, bien que nécessitant encore des recherches complémentaires pour pallier à certaines difficultés de mise en œuvre pratique et pour améliorer leurs performances (temps de calculs, etc.), les méthodes meshfree sont bien adaptées pour simuler des problèmes

géomécaniques en transformations finies, du fait de leur plus grande souplesse en termes de discrétisation et de raffinement adaptatif local ou global, et également de leur capacité à fournir des résultats en tout point du domaine d'étude sans perte de précision. Néanmoins, les méthodes meshfree sont à l'heure actuelle encore largement sous utilisées dans la pratique de l'ingénieur, les outils numériques industriels proposant ces méthodes étant à peu près inexistants ou alors très limités au niveau de leurs applications.

Du point de vue de l'outil *MoveFree*, nous envisageons de poursuivre nos développements et validations, afin de le rendre utilisable dans un contexte opérationnel et industriel. Nous focaliserons nos efforts notamment sur les points suivants :

- Développements / améliorations complémentaires envisagés :
  - implantation de la description lagrangienne réactualisée ;
  - résolution de type hyperélastique au niveau des lois de comportement solides ;
  - ajout de nouveaux modèles de comportement, notamment dans l'optique de simuler le comportement post-liquéfaction (par exemple, des modèles viscoplastiques) ;
  - ajout de frontières absorbantes pour le calcul de la propagation des ondes dans un domaine semi-infini ;
  - implantation d'algorithmes de résolution non linéaire plus performants (par exemple BFGS) ;
  - mise en œuvre et validation de l'intégration purement nodale dans le cas des milieux poreux biphasiques ;
  - gestion automatisée de certains aspects numériques : détermination du pas de temps suivant les critères de stabilité et précision, raffinement adaptatif, etc.
- Validations complémentaires :
  - exemples tridimensionnels simples : les structures de données et les algorithmes sont en fait déjà tridimensionnels, mais seules les géométries mono et bidimensionnelles ont été validées à ce jour ;
  - exemples réels : divers sites instrumentés avec glissements actifs mobilisant de grandes masses de sols sont actuellement à l'étude, pour compléter la validation du logiciel.

# Références Bibliographiques


[AUB 82] **Aubry** D., **Hujeux** J.C., **Lassoudiere** F., **Meimon** Y., 'A double memory model with multiple mechanism for cyclic soil behaviour', *International Symposium on Numerical Models in Geomechanics*, Zurich, Suisse, 3-13, 1982.

[AUB 97] **Aubert** P., 'Méthodes meshless en Géomécanique', *Thèse de Doctorat, Ecole Centrale de Paris*, France, 1997.

[ATL 98] **Atluri** S. N. & **Zhu** T., 'A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics', *Computational Mech.*, **22**: 117-127, 1998.

[BAB 95] **Babuška** I. & **Melenk** J. M., 'The partition of unity finite element method', *Technical Report BN-1185, Institute for Physical Science and Technology, University of Maryland*, 1995.

[BAB 97] **Babuška** I. & **Melenk** J. M., 'The Partition of Unity Method', *Int. J. Num. Meth. Engng.*, **40**: 727-758, 1997.

[BAT 75] **Bathe** K. J., **Ramm** E., **Wilson** E.L., 'Finite element formulations for large deformation dynamic analysis', *Int. J. Num. Meth. Engng.*, **9**: 353-386, 1975.

[BEI 96] **Beissel** S. & **Belytschko** T., 'Nodal integration of the Element-free Galerkin method', *Computer Meth. Applied Mech. Engng.*, **139**: 49-74, 1996.

[BEL 94] **Belytschko** T., **Lu** Y. Y., **Gu** L., 'Element-free Galerkin methods', *Int. J. Num. Meth. Engng.*, **37**: 229-256, 1994.

[BEL 95] **Belytschko** T., **Organ** D., **Krongauz** Y., 'A coupled finite element-element-free Galerkin method', *Computational Mech.*, **17**:186-195, 1995.

[BEL 96] **Belytschko** T., **Krongauz** Y., **Organ** D., **Fleming** M., **Krysl** P., 'Meshless methods: an overview and recent developments', *Computer Meth. Applied Mech. Engng.*, **139**: 3-48, 1996.

[BEL 00] **Belytschko** T., **Guo** Y., **Liu** W. K., **Xiao** S. P., 'A unified stability analysis of meshless particle methods', *Int. J. Num. Meth. Engng.*, **48**: 1359-1400, 2000.

[BEN 74] **Benzley** S. E., 'Representation of singularities with isoparametric finite elements', *Int. J. Num. Meth. Engng.*, **8**: 537-545, 1974.

[BIO 41] **Biot** M.A., 'General theory of three-dimensional consolidation', *J. Appl. Physics.*, **12**: 155-164, 1941.

[BIO 55] **Biot** M.A., 'Theory of elasticity and consolidation for a porous anisotropic solid', *J. Appl. Physics.*, **26(2)**: 182-185, 1955.

[BIO 77] **Biot** M.A., 'Variational Lagrangian Thermodynamics of Non-Isothermal Finite Strain Mechanics of Porous Solids and Thermomolecular Diffusion', *Int. J. Solids Struct.*, *ASCE*, **13**: 579-597, 1977.

[BON 00] **Bonet** J. & **Kulasegaram** S., 'Correction and stabilization of smooth particle hydrodynamics methods with applications in metal forming simulations', *Int. J. Num. Meth. Engng.*, **47**: 1189-1214, 2000.

[BOR 98] **Borja** R.I. & **Tamagnini**, C., 'Cam-Clay plasticity, Part III: Extension of the infinitesimal model to include finite strains', *Computer Meth. Applied Mech. Engng.*, **155**: 73-95, 1998.

[BOU 97] **Bourgeois** E., 'Mécanique des milieux poreux en transformation finie : formulation des problèmes et méthodes de résolution', *Thèse de doctorat, Ecole Nationale des Ponts et Chaussées*, Paris (1997).

[BRE 00] **Breitkopf** P., **Rassineux** A., **Touzot** G., **Villon** P., 'Explicit form and efficient computation of MLS shape functions and their derivatives', *Int. J. Num. Meth. Engng.*, **48**: 451-466, 2000.

[BUH 01] **Buhmann** M. D., 'A new class of radial basis functions with compact support', *Mathematics of Computation*, **70**: 307-318, 2001.

[CAR 91] **Carlson** R. E. & **Foley** T. A., 'The parameter $R^2$ in Multiquadrics interpolation', *Computer Math. Applic.*, **21**(9): 29-42, 1991.

[CHE 00] **Chen** J. K. & **Beraun** J. E., 'A generalized smoothed particle hydrodynamics method for nonlinear dynamic problems', *Computer Meth. Applied Mech. Engng.*, **190**: 225-239, 2000.

[CHE 01] **Chen** J. S., **Wu** C. T., **Yoon** S., **You** Y., 'A stabilized conforming nodal integration for Galerkin mesh-free methods', *Int. J. Num. Meth. Engng.*, **50**: 435-466, 2001.

[CHE 02a] **Chen** J. S., **Golberg** M. A., **Schaback** R. S., 'Recent developments of the dual reciprocity method using compactly supported radial basis functions', *to appear in: Transformation of Domain Effects to the Boundary, ed. Y.F. Rashed, Advances in Boundary Elements, WIT Press,* Boston, 2002.

[CHE 02b] **Chen** J. S., **Yoon** S., **Wu** C. T., 'Non-linear version of stabilized conforming nodal integration for Galerkin mesh-free methods', *Int. J. Num. Meth. Engng.*, **53**: 2587-2615, 2002.

[CHE 03] **Chen** C. S., **Han** W., **You** Y., **Meng** X., 'A reproducing kernel method with nodal interpolation property', *Int. J. Num. Meth. Engng.*, **56**: 935-960, 2003.

[CHO 92] **Chopra** M. B. & **Dargush** G. F., 'Finite-element analysis of time-dependent large-deformation problems', *Int. J. Num. Anal. Meth. Geomech.*, **16**: 101-130, 1992.

[COU 91] **Coussy** O., 'Mécanique des milieux poreux', *Technip*, Paris, 1991.

[CUI 92] **Cuitiño** A. & **Ortiz** M., 'A material-independent method for extending stress update algorithms from small-strain plasticity to finite plasticity with multiplicative kinematics', *Engineering Computations*, **9**: 437-451, 1992.

[DAF 83] **Dafalias** Y. F., 'Co-rotational rates for kinematic hardening at large plastic deformations', *J. Applied Mech.*, **50**: 561-565, 1983.

[DIE 79] **Dienes** J. K., 'On the analysis of rotation and stress rate in deforming bodies', *Acta Mechanica*, **32**: 217-232, 1979.

[DOG 84] **Dogui** A. & **Sidoroff** F. 'Rhéologie anisotrope en grandes déformations', *Actes du 19ème Colloque G.F.R.*, Paris, 1984.

[DOL 98] **Dolbow** J. E. 'Numerical integration in meshfree methods', *Master's Thesis report, Northwestern University*, 1998.

[DUA 95] **Duarte** C. A. & **Oden** J. T., '*H-p* clouds: an *h-p* meshless method to solve boundary-value problems', *Technical Report 95-05, TICAM, University of Texas at Austin*, 1995.

[DUA 96a] **Duarte** C. A. & **Oden** J. T., '*H-p* clouds: an *h-p* meshless method', *Num. Meth. Partial Diff. Eq.*, **12**: 673-706, 1996.

[DUA 96b] **Duarte** C. A. & **Oden** J. T., 'An *h-p* adaptive method using clouds', *Computer Meth. Applied Mech. Engng.*, **139**: 237-262, 1996.

[DUB 94] **Dubal** M. R., 'Domain decomposition and local refinement for multiquadrics approximations. I: Second order equations in one dimension', *J. Appl. Sci. Comp.*, **1**: 146-171, 1994.

[DYK 97] **Dyka** C. T., **Randles** P. W., **Ingel** R. P., 'Stress points for tension instability in SPH', *Int. J. Num. Meth. Engng.*, **40**: 2325-2341, 1997.

[ETE 90] **Eterovic** A. L. & **Bathe** K. J., 'A hyperelastic-based large strain elasto-plastic constitutive formulation with combined isotropic-kinematic hardening using the logarithmic stress and strain measures', *Int. J. Num. Meth. Engng.*, **30**: 1099-1114, 1990.

[EVE 94] **Eve** R. A. & **Reddy** B. D., 'The variational formulation and solution of problems of finite-strain elastoplasticity based on the use of a dissipation function', *Int. J. Num. Meth. Engng.*, **37**: 1673-1695, 1994.

[FAS 97] **Fasshauer** G. E., 'Solving partial differential equations by collocation with radial basis functions', *Proceedings of Chamonix 1996, Vanderbilt University Press*, Nashville, 1-8, 1997.

[FAS 98] **Fasshauer** G. E. & **Schumaker** L. L., 'Scattered data fitting on the sphere', in *Mathematical Methods for Curves and Surfaces II*, *M. Daehlen, T. Lyche & L.L. Schumaker*, *Eds., Vanderbilt University Press*, Nashville, 117-166, 1998.

[FAS 99] **Fasshauer** G. E., 'Solving partial differential equations with radial basis functions: multilevel methods and smoothing', *Advances in Computational Mathematics*, **11**: 139-159, 1999.

[FAS 01] **Fasshauer** G. E., 'Non symmetric multilevel RBF collocation within an operator Newton framework for nonlinear PDEs', in *Trends in Approximation Theory, K. Kopotun, T. Lyche & M. Neamtu*, *Eds.*, *Vanderbilt University Press*, Nashville, 103-112, 2001.

[FAS 02] **Fasshauer** G. E., 'Approximate moving least-squares approximation with compactly supported radial weights', *Lecture Notes in Computer Science and Engineering: Meshfree Methods for Partial Differential Equations*, **26**: 105-116, 2002.

[FLO 96] **Floater** M. S. & **Iske** A., 'Multistep scattered data interpolation using compactly supported radial basis functions', *J. Comput. Applied Math.*, **73**: 65-78, 1996.

[FRA 82] **Franke** R., 'Scattered data interpolation: test of some methods', *Mathematics of Computations*, **38**: 181-200, 1982.

[FRA 98a] **Franke** R. & **Schaback** R., 'Solving partial differential equations by collocation using radial basis functions', *J. Applied Mathematics and Computations*, **93**: 73-82, 1998.

[FRA 98b] **Franke** R. & **Schaback** R., 'Convergence order estimates of meshless collocation methods using radial basis functions', *Advances in Computational Mathematics*, **8**: 381-399, 1998.

[GAV 00] **Gavete** L., **Benito** J. J., **Falcon** S. & **Ruiz** A., 'Penalty functions in constrained variational principles for element free Galerkin method', *Eur. J. Mech. A/Solids*, **19**: 699-720, 2000.

[GIN 77] **Gingold** R. A. & **Monaghan** J. J., 'Smoothed particle hydrodynamics: theory and applications to non-spherical stars', *Mon. Not. R. Astr. Soc.*, **181**: 375-389, 1977.

[GOS 96] **Gosz** J. & **Liu** W. K., 'Admissible approximations for essential boundary conditions in the reproducing kernel particle method', *Computational Mechanics*, **19**: 120-135, 1996.

[GRE 71] **Green** A. E., **Naghdi** P. M., 'Some remarks on elastic-plastic deformation at finite strain', *Int. J. Engng. Sc.*, **9**: 1219-1229, 1971.

[GU 01] **Gu** Y. T. & **Liu** G. R., 'A boundary point interpolation method (BPIM) using radial function basis', *1st MIT Conference of Computational Fluid and Solid Mech.*, *MIT*, Massachusetts, 1590-1592, Juin 2001.

[GU 02] **Gu** Y. T. & **Liu** G. R., 'A boundary point interpolation method for stress analysis, of solids', *Computational Mech.*, **28**: 47-54, 2002.

[HIB 70] **Hibbitt** H. D., **Marcal** P. V., **Rice** J. R., 'A finite-element formulation for problems of large strain and large displacement', *Int. J. Solids and Structures*, **6**: 1069-1086, 1970.

[HIL 50] **Hill** R., 'The Mathematical Theory of Plasticity', *Clarendon Press*, Oxford, 1950.

[HOF 90] **Hofstetter** G. & **Taylor** R. L., 'Non-associative Drücker-Prager plasticity at finite strains', *Comm. Applied Num. Meth.*, **6**: 583-589, 1973.

[HUE 00] **Huerta** A. & **Fernández-Méndez** S., 'Enrichment and coupling of the finite element and meshless methods', *Int. J. Num. Meth. Engng.*, **48**: 1615-1636, 2000.

[HUJ 85] **Hujeux** J.C., 'Une loi de comportement pour les chargements cycliques des sols', in *Génie Parasismique, Davidovici, ed.*, *Presse ENPC*, Paris, 1985.

[HUG 80a] **Hughes** T. J. R., 'Generalization of selective integration procedures to anisotropic nonlinear media', *Int. J. Num. Meth. Engng.*, **15**: 1413-1418, 1980.

[HUG 80b] **Hughes** T. J. R. & **Winget** J., 'Finite rotation effects in numerical integration of rate constitutive equations arising in large deformation analysis', *Int. J. Num. Meth. Engng.*, **15**: 1862-1867, 1980.

[ISK 02] **Iske** A. & **Levesley** J., 'Multilevel scattered data approximation by adaptive domain decomposition', *Technical Report No. 2002/17*, University of Leicester, 2002.

[JOH 96] **Johnson** G. R. & **Beissel** S. R., 'Normalized smoothing functions for SPH impact computations', *Int. J. Num. Meth. Engng.*, **39**: 2725-2741, 1996.

[KAN 90a] **Kansa** E. J., 'Multiquadrics: a scattered data approximation scheme with applications to computational fluid-dynamics – I. Surface approximations and partial derivative estimates', *Comput. Math. Applic.*, **19**: 127-145, 1990.

[KAN 90b] **Kansa** E. J., 'Multiquadrics: a scattered data approximation scheme with applications to computational fluid-dynamics – II. Solutions to hyperbolic, parabolic and elliptic partial differential equations', *Comput. Math. Applic.*, **19**: 147-161, 1990.

[KIO 85] **Kiousis** P. D. & **Voyiadjis** G. Z., 'Lagrangian continuum theory for saturated porous media', *J. Engng. Mech.*, **111**(10): 1277-1289, 1985.

[KIO 88] **Kiousis** P. D., **Voyiadjis** G. Z. & **Tumay** M.T., 'A large strain theory and its application in the analysis of the cone penetration mechanism', *Int. J. Num. Anal. Meth. Geomech.*, **12**: 45-60, 1988.

[KRO 96] **Krongauz** Y. & **Belytschko** T., 'Enforcement of essential boundary conditions in meshless approximations using finite elements', *Computer Meth. Applied Mech. Engng.*, **131**: 133-145, 1996.

[KRY 00] **Krysl** P. & **Belytschko** T., 'An efficient linear-precision partition of unity basis for unstructured meshless methods', *Comm. Num. Meth. Engng.*, **16**(4): 239-255, 2000.

[LAN 81] **Lankaster** P. & **Salkauskas** K., 'Surfaces generated by moving least squares methods', *Math. Comput.*, **37:** 141-158, 1981.

[LAO 96] **Laouar** T., 'Contribution à l'étude de l'approximation diffuse : auto - adaptativité en éléments diffus', *Thèse de Doctorat, Université de Technologie de Compiègne,* 1996.

[LEE 69] **Lee** E. H., 'Elastic-plastic deformation at finite strains', *J. Applied. Mech. ASME*, **36:** 1-6, 1969.

[LIU 95a] **Liu** W. K. & **Chen** Y., 'Wavelet and multiple scale reproducing kernel methods', *Int. J. Num. Meth. Fluids*, **21**: 901-932, 1995.

[LIU 95b] **Liu** W. K., **Jun** S. & **Zhang** Y. F., 'Reproducing kernel particle methods', *Int. J. Num. Meth. Engng.*, **20**: 1081-1106, 1995.

[LIU 96a] **Liu** W. K., **Chen** Y., **Chang** C. T. & **Belytschko** T., 'Advances in multiple scale kernel particle methods', *Computational Mechanics*, **18**: 73-111, 1996.

[LIU 96b] **Liu** W. K., **Chen** Y., **Uras** R.A. & **Chang** C. T., 'Generalized multiple scale reproducing kernel particle methods', *Computer Meth. Applied Mech. Engng.*, **139**: 91-158, (1996).

[LIU 97] **Liu** W. K., **Li** S., **Adee** J. & **Belytschko** T., 'Moving least square reproducing kernel methods: (i) methodology and convergence', *Computer Meth. Applied Mech. Engng*, **143**: 113-154, 1997.

[LIU 99] **Liu** G. R. & **Gu** Y. T., 'A point interpolation method', *Proceedings of the 4th Asia-Pacific Conference on Computational Mechanics*, Singapore, 1009-1014, Dec. 1999.

[LIU 00a] **Liu** W. K., **Hao** S., **Belytschko** T., **Li** S. & **Chang** C. T., 'Multi-scale methods', *Int. J. Num. Meth. Engng.*, **47**: 1343-1361, 2000.

[LIU 00b] **Liu** G. R. & **Gu** Y. T., 'Coupling of element free Galerkin method with boundary point interpolation method', in *Advances in Computational Engng. & Science*, *Atluri S. N. & Brust F. W.*, *Eds.*, *ICES'2K*, Los Angeles, 1427-1432, Août 2000.

[LIU 01a] **Liu** G. R. & **Gu** Y. T., 'A point interpolation method for two-dimensional solids', *Int. J. Num. Meth. Engng.*, **50**: 937-951, 2001.

[LIU 01b] **Liu** G. R. & **Gu** Y. T., 'A local point interpolation method for stress analysis of two-dimensional solids', *Struct. Engng. Mech.*, **11**: 221-236, 2001.

[LIU 01c] **Liu** G. R. & **Gu** Y. T., 'A local radial point interpolation method (LR-PIM) for free vibration analyses of 2-D solids', *J. Sound Vib.*, **246**: 29-46, 2001.

[LIU 03] **Liu** G. R., 'Mesh free methods: moving beyond the finite element method', *4th ed., Vols. 1 and 2, CRC Press*, London, 2003.

[LU 94] **Lu** Y. Y., **Belytschko** T., **Gu** L., 'A new implantation of the element-free Galerkin method', *Computer Meth. Applied Mech. Engng.*, **113**: 397-414, 1994.

[LUC 77] **Lucy** L., 'A numerical approach to testing the fission hypothesis', *Astron. J.*, **82**: 1013-1024, 1977.

[MAN 72] **Mandel** J., 'Plasticité classique et viscoplasticité', *CISM No. 97, Udine 1971, Springer-Verlag*, Vienna, New-York (1972).

[MAS 00] **Masud** A., 'A multiplicative finite strain finite element framework for the modelling of semicrystalline polymers and polycarbonates', *Int. J. Num. Meth. Engng.*, **47:** 1887-1908 (2000).

[MAU 99] **Mauget** B. & **Perré** P., 'A large displacement formulation for anisotropic laws', *Eur. J. Mech. A/Solids*, **18:** 859-877, 1999.

[MCK 75] **McKeeking** R. M. & **Rice** J. R., 'Finite-element formulations for problems of large elastic-plastic deformation', *Int. J. Solids and Structures*, **11:** 601-616, 1975.

[MEL 96] **Melenk** J. M. & **Babuška** I., 'The Partition of Unity Method: basic theory and applications', *Computer Meth. Applied Mech. Engng.*, **139**: 289-314, 1996.

[MER 95] **Meroi** E. A., **Schrefler** B. & **Zienkiewicz** O. C., 'Large strain static and dynamic semisaturated soil behaviour', *Int. J. Num. Anal. Meth. Geomech. Engng.*, **19**: 81-106, 1995.

[MOD 87] **Modaressi** H., 'Modélisation numérique de la propagation des ondes dans les milieux poreux anélastiques', *Thèse de Doctorat, Ecole Centrale de Paris*, 1987.

[MOD 95] **Modaressi** H. & **Aubert** P., 'Coupled FEM-EFGM for consolidation modelling', *Proceedings of the 5th International Symposium on Numerical Models in Geomechanics (NUMOG V)*, Davos, Suisse, 417-422, Sept. 1995.

[MOD 96] **Modaressi** H. & **Aubert** P., 'A diffuse element-finite element technique for transient coupled analysis', *Int. J. Num. Meth. Engng.*, **39**:3809-3838, 1996.

[MOD 97] **Modaressi** H. & **Aubert** P., 'Meshless Numerical Methods in Geomechanics', *International Conference of the International Association for Computer Methods and Advances in Geomechanics (IACMAG IX)*, Wuhan, Chine, 1997.

[MOD 98] **Modaressi** H. & **Aubert** P., 'Element-free Galerkin method for deforming multiphase porous media', *Int. J. Num. Meth. Engng.,* **42**:313-340, 1998.

[MON 82] **Monaghan** J. J., 'Why particle methods work', *SIAM J. Sci. Stat. Comput.*, **3**: 422-433, 1982.

[MON 88] **Monaghan** J. J., 'An introduction to SPH', *Comput. Physics Comm.*, **48**: 89-96, 1988.

[MUK 97a] **Mukherjee** Y. X. & **Mukherjee** S., 'The boundary node method for potential problems', *Int. J. Num. Meth. Engng.*, **40**: 797-815, 1997.

[MUK 97b] **Mukherjee** Y. X. & **Mukherjee** S., 'On boundary conditions in the element-free Galerkin method', *Comput. Mechanics*, **19**: 264-270, 1997.

[NAG 81] **Nagtegaal** J. C. & **De Jong** J. E., 'Some computational aspects of elastic-plastic large strain analysis', *Int. J. Num. Meth. Engng.*, **17**: 15-41, 1981.

[NAG 99] **Nagashima** T., 'Node-by-node meshless approach and its applications to structural analyses', *Int. J. Num. Meth. Engng.*, **46**: 341-385, 1999.

[NAY 91a] **Nayroles** B., **Touzot** G. & **Villon** P., 'L'approximation diffuse', *C.R. Acad. Sci. Paris*, t313, Série II : 133-138, 1991.

[NAY 91b] **Nayroles** B., **Touzot** G. & **Villon** P., 'Generalizing the finite element method: diffuse approximation and diffuse elements', *Computational Mechanics*, **10**: 307-318, 1992.

[NAY 92] **Nayroles** B., **Touzot** G. & **Villon** P., 'Generalizing the finite element method: diffuse approximation and diffuse elements', *Computational Mechanics*, **10**: 307-318, 1992.

[NEM 79] **Nemat-Nasser** S., 'Decomposition of strain measures and their rates in finite deformation elastoplasticity', *Int. J. Solids and Structures.*, **15**: 155-166, 1979.

[NEM 82] **Nemat-Nasser** S., 'On finite deformation elasto-plasticity', *Int. J. Solids and Structures.*, **18**: 857-872, 1982.

[ORG 96] **Organ** D. J., 'Numerical solutions to dynamic fracture problems using the Element-Free Galerkin method', *Ph.D.*, *Northwestern University*, Evanston, Illinois, 1996.

[PER 99a] **Perić** D. & **Souza Neto** E. A. (de), 'A new computational model for Tresca plasticity at finite strains with an optimal parameterization in the principal space', *Computer Meth. Applied Mech. Engng.*, **171**: 463-489, 1999.

[PER 99b] **Perić** D., **Vaz Jr.** M., **Owen** D. R. J. 'On adaptive strategies for large deformations of elasto-plastic solids at finite strains: computational issues and industrial applications', *Computer Meth. Applied Mech. Engng.*, **176**: 279-312, 1999.

[PIN 83] **Pinsky** P. M., **Ortiz** M. & **Pister** K. S., 'Numerical integration of rate constitutive equations in finite deformation analysis', *Computer Meth. Applied Mech. Engng.*, **40:** 137-158, 1983.

[POW 90] **Powell** M. J. D., 'The theory of radial basis function approximation in 1990', *University of Cambridge Numerical Analysis Reports*, DAMTP 1990/NA11.

[RAN 96] **Randles** P. W. & **Libertsky** L. D., 'Smoothed particle hydrodynamics: some recent improvements and applications', *Computer Meth. Applied Mech. Engng.*, **139**: 375-408, 1996.

[RAS 96] **Rashid** M. M. & **Thorne** B. J., 'Incremental objectivity in cyclic shearing deformations', *Comm. Num. Meth. Engng.*, **12:** 863-871, 1996.

[ROD 97] **Rodríguez-Ferran** A., **Pegon** P. & **Huerta** A., 'Two stress update algorithms for large strains: accuracy analysis and numerical implementation', *Int. J. Num. Meth. Engng.*, **40:** 4363-4404, 1997.

[ROD 98] **Rodríguez-Ferran** A., **Vila** A. & **Huerta** A., 'Comparing two algorithms to add large strains to a small strain finite element code', *ASCE J. Engng. Mechanics*, **124:** 939-948, 1998.

[SCH 00] **Schaback** R. & **Wendland** H., 'Adaptive greedy techniques for approximate solution of large RBF systems', *Numerical Algorithms*, **24**: 239-254 (2002).

[SHE 68] **Shepard** D., 'A two-dimensional function for irregularly spaced data', *ACM National Conference*, 517-524, 1968.

[SIC 02] **Sica** S., **Pagano** L., **Modaressi**, A., 'Numerical analysis of a zoned earth dam by a coupled approach', *12th European Conference in Earthquake Engineering*, London, 2002.

[SID 84] **Sidoroff** F., 'Mécanique et thermodynamique des milieux continus', *Cours de 3ème Année / DEA de l'École Centrale de Lyon*, Ecully, 1984.

[SIM 85] **Simo** J. C. & **Ortiz** M., 'A unified approach to finite deformation elastoplasticity analysis based on the use of hyperelastic constitutive equations', *Computer Meth. Applied Mech. Engng.*, **49**: 221-245, 1985.

[SIM 88a] **Simo** J. C., 'A framework for finite strain elastoplasticity based on maximum plastic dissipation and the multiplicative decomposition. Part I: continuum formulation', *Computer Meth. Applied Mech. Engng.*, **66**: 199-219, 1988.

[SIM 88b] **Simo** J. C., 'A framework for finite strain elastoplasticity based on maximum plastic dissipation and the multiplicative decomposition. Part II: computational aspects', *Computer Meth. Applied Mech. Engng.*, **68**: 1-31, 1988.

[SIM 92] **Simo** J. C., 'Algorithms for static and dynamic multiplicative plasticity that preserves the classical return mapping schemes of the infinitesimal theory', *Computer Meth. Applied Mech. Engng.*, **99**: 61-112, 1992.

[STO 92] **Stolle** D. F. E. & **Schad** H., 'An updated reference configuration formulation for large-deformation problems', *Int. J. Num. Anal. Meth. Geomech.*, **16**: 295-306, 1992.

[SWE 93] **Swegle** J. W., **Attaway** S. W., **Heinstein** M. W., **Mello** F. J. & **Hicks** D, 'An analysis of smoothed particle hydrodynamics', *Technical Report SAND93-2513 UC-705*, Sandia National Laboratories, 1993.

[SWE 95] **Swegle** J. W., **Hicks** D & **Attaway** S. W., 'Smoothed particle hydrodynamics stability analysis', *J. Comp. Physics*, **116**: 123-134, 1995.

[VOY 89] **Voyiadjis** G. Z. & **Kattan** P. I., 'Eulerian constitutive model for finite deformation plasticity with anisotropic hardening', *Mech. Materials*, **7**: 279-293, 1989.

[VOY 97] **Voyiadjis** G. Z. & **Abu-Farsakh** M. Y., 'Coupled theory of mixtures for clayey soils', *Computers and Geotechnics*, **20**: 195-222, 1997.

[WAN 00] **Wang** G. J. & **Liu** G. R., 'Radial point interpolation method for elastoplastic problems', *Proceedings of the 1st Int. Conf. On Structural Stability and Dynamics*, Taipei, Taiwan, 703-708, Dec. 2000.

[WAN 02a] **Wang** G. J., **Liu** G. R., **Lin** P., 'Numerical analysis of Biot's consolidation process by radial point interpolation method', *Int. J. Solids Struct.*, **39**: 1557-1573, 2002.

[WAN 02b] **Wang** G. J. & **Liu** G. R., 'A point interpolation method based on radial basis functions', *Int. J. Num. Meth. Engng.*, **54**: 1623-1648, 2002.

[WEN 95] **Wendland** H., 'Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree', *Advances in Computational Mathematics*, **4**: 389-396, 1995.

[WEN 99] **Wendland** H., 'Meshless Galerkin methods using radial basis functions', *Mathematics of Computation*, **68**: 1521-1531, 1999.

[WEN 01] **Wendland** H., 'Gaussian interpolation revisited', *Trends in Approximation Theory, K. Kopotun, T. Lyche & M. Neamtu*, *Eds.*, *Vanderbilt University Press*, Nashville, 417-426, 2001.

[WEN 02] **Wendland** H., 'Fast evaluation of radial basis functions: methods based on Partition of Unity', *Approximation Theory X: Wavelets, Splines and Applications, C. Chui L.L. Schumaker & J. Stöckler*, *Eds., Vanderbilt University Press*, Nashville, 473-483, 2002.

[WU 93] **Wu** Z. & **Schaback** R. A., 'Local error estimates for radial basis function interpolation of scattered data', *IMA J. Num. Anal.*, **13**: 13-27, 1993.

[WU 95] **Wu** Z., 'Compactly supported positive definite radial functions', *Advances in Computational Mathematics*, **4**: 283-292, 1995.

[ZAB 00] **Zabaras** N., **Bao** Y., **Srikanth** A., **Frazier** W.G., 'A continuum Lagrangian sensitivity analysis for metal forming processes with applications to die design problems', *Int. J. Num. Meth. Engng.*, **48**: 679-720, 2000.

[ZER 00] **Zerroukat** M., **Djidjeli** K. & **Charafi** A., 'Explicit and implicit meshless methods for linear advection-diffusion-type partial differential equations', *Int. J. Num. Meth. Engng.*, **48**: 19-35, 2000.

[ZHA 00] **Zhang** X., **Song** K. Z., **Lu** M. W., **Liu** X., 'Meshless methods based on collocation with radial basis functions', *Computational Mech.*, **26**: 333-343, 2000.

[ZIE 80] **Zienkiewicz** O. C., **Chang** C. T., **Bettess** P., 'Drained, undrained, consolidating and dynamic behavior assumption in soils', *Géotechnique*, **30:**385-395, 1980.

[ZIE 84] **Zienkiewicz** O. C. & **Shiomi** T., 'Dynamic behaviour of saturated porous media; The generalized Biot formulation and its numerical solution', *Int. J. Num. Anal. Meth. Geomech.*, **8:**71-96, 1984.

[ZIE 91] **Zienkiewicz** O. C. & **Taylor** R. L., 'The Finite Element Method', *4th ed., Vols. 1 and 2, McGraw-Hill*, London, 1991.

[ZHU 98] **Zhu** T., **Zhang** S., **Atluri** S. N. 'A local boundary integral equation (LBIE) in computational mechanics and a meshless discretization approach', *Computational Mech.*, **21**: 223-235, 1998.

# Annexes

# Annexe A

# Aperçu du langage C++

## A.1 INTRODUCTION

L'objectif de cette section est de fournir un rapide aperçu du langage C++. Le lecteur désireux d'en connaître davantage se reportera notamment à l'ouvrage de Bjarne Stroustrup « *The C++ Programming Language* » (1987).

Le langage C++ est un langage orienté objet, c'est-à-dire qu'il utilise des entités autonomes (objets) permettant de représenter des abstractions du monde réel. C'est un langage structuré, dont la syntaxe s'apparente à celle du C. De même qu'en C, chaque instruction se termine par un point-virgule « ; » et un bloc d'instructions est encadré par des accolades {…}. Les variables sont typées et les types sont soit prédéfinis, soit définis par le programmeur . Quant à la visibilité des variables ou des fonctions, la portée d'une déclaration ne dépasse pas le bloc immédiatement englobant, à moins d'utiliser explicitement l'attribut « extern » pour signifier que la déclaration effective est réalisée à un autre endroit.

## A.2 TYPES FONDAMENTAUX

La notion de type permet de définir d'une part les opérations pouvant être appliquées à une entité donnée (variable, fonction, objet, etc.), et d'autre part la signification de ces opérations. Un certain nombre de type fondamentaux existe en C++, permettant de représenter des nombres entiers ou réels :

| Nombres | Type |
|---|---|
| *Entiers* | *char* |
| | *short* |
| | *int* |
| | *long* |
| *réels* | *float* |
| | *double* |

***Tabl. 1 : Types fondamentaux***

Par définition, la taille occupée en mémoire par un *char* est égale à un byte. Celle occupée par tout autre nombre est un multiple de la taille de *char*. Les types indiqués dans le tableau 1 sont rangés par ordre de taille croissante.

Hormis ces types fondamentaux, le programmeur peut créer des types nouveaux ou dérivés des types fondamentaux. Il peut pour ce faire utiliser le mot clé ***typedef***, ce qui par exemple donne :

***typedef const char**** LPCTSTR;

Il peut également utiliser le mot clé ***enum*** pour créer un sous-type énuméré constitué d'une suite croissante de nombres entiers contigus. Par exemple, on définira le type énuméré *mes_entiers* par :

***enum*** *mes_entiers* {*PREMIER* = 1, *SECOND*, *TROISIEME*} ;

Dans ce cas, *SECOND* = 2, etc. Par ailleurs, lorsque ce n'est pas précisé, la valeur du premier nombre est zéro.

## A.3 CLASSES ET OBJETS

Ecrire un programme C++ consiste à définir un ensemble de classes ou modèles pour les objets réels, dont les instances (objets) communiquent entre elles. Définir une classe *T* consiste à spécifier les constituants de cette classe, en distinguant d'une part les *attributs*, c'est-à-dire la structure des données membres permettant de représenter un objet réel de type *T*, et d'autre part les *méthodes*, c'est-à-dire l'ensemble des opérations (fonctions ou procédures) admissibles sur cet objet et permettant de le manipuler.

Plus généralement, les méthodes d'une classe exploitent et éventuellement modifient les attributs de la classe. Par ailleurs, quel que soit l'endroit où ils sont déclarés, les attributs et méthodes d'une même classe sont utilisables dans toute méthode de cette classe.

Par exemple, si l'on veut représenter des points de coordonnées réelles $(x, y, z)$ dans l'espace euclidien $\mathbb{R}^3$, on peut définir une classe *Point3D* de la manière suivante[1] :

```
class Point3D
{
private:
   // Attributs[2]
   float x, y, z;

public:
   // Méthodes
   …
};
```

[1] Les mots réservés du langage sont indiqués en gras et italique.
[2] Les doubles barres // indiquent une ligne de commentaires. Il est possible d'encadrer des commentaires dépassant plusieurs lignes par /* et */ (comme en C).

C++ offre trois niveaux de connaissance (*encapsulation*) sur les constituants d'une classe, vis-à-vis du « monde extérieur » :

- Une connaissance complète (mot clé : ***public***) : l'utilisateur connaît le composant et son type et peut les manipuler directement (pour une classe dont tous les composants sont publics, on préfère utiliser le mot clé ***struct***) ;

- Aucune connaissance : un ou plusieurs constituants (attributs ou méthodes) de la classe sont entièrement cachés à l'utilisateur et ne sont accessibles qu'à l'intérieur de la classe (mot clé : ***private***) ;

- Une connaissance sélective aux classes dérivées : l'accès à un ou plusieurs constituants de la classe est limité à la classe elle-même (« mère ») et à ses classes dérivées (« filles ») par l'utilisation du mot clé : ***protected***.

Afin d'initialiser correctement les objets construits à partir d'une classe donnée, il est nécessaire de définir un ou plusieurs constructeurs de cet objet. Ceux-ci se présentent comme des méthodes, à la différence qu'ils doivent obligatoirement porter le même nom que la classe et qu'ils ne sont pas précédés d'un nom de type.

Lorsque la création d'un objet est effectuée dans un bloc[3] sans faire appel à l'opérateur ***new***[4], la destruction se fait automatiquement à la fin d'exécution du bloc. Le programmeur n'a donc pas à s'en préoccuper. En revanche, tout objet crée par allocation dynamique existe jusqu'à sa destruction explicite par l'opérateur ***delete***. Cependant, la destruction explicite de l'espace mémoire ne concerne que celui effectivement occupé par les propres composants de l'objet, et non pas celui pouvant être adressé par des pointeurs. La libération effective de l'espace occupé par les objets pointés devra dans ce cas être réalisée explicitement par le programmeur au moyen d'une opération que l'on nomme destructeur. De même que pour le(s) constructeur(s), celui-ci est défini en utilisant le nom de la classe précédé de ~.

Si l'on complète l'exemple précédent, on aura :

```
class Point3D
{
private:
   float x, y, z; // Attributs

public:
   Point3D() {x = 0; y = 0; z = 0;} // Constructeur par défaut
   Point3D(float a, float b, float c) {x = a; y = b; z = c;} // Autre constructeur possible
   // Constructeur par copie
   Point3D(const Point3D& the_point) {x = the_point.x; y = the_point.y; z = the_point.z;}

   ~Point3D () ; // Destructeur : déclaration sans implémentation[5]
   …
```

[3] Le bloc est l'ensemble des instructions ou opérations situées entre deux accolades : {…}.
[4] L'opérateur ***new*** appliqué à un objet de type *T*, permet de créer un pointeur **p** sur cet objet, c'est-à-dire qu'il permet d'allouer dynamiquement un espace mémoire de taille suffisante pour contenir la représentation de l'objet en question. L'adresse de cet espace est alors conservée au niveau du pointeur **p**.
[5] Le corps d'une méthode peut être fourni soit au niveau de sa déclaration, soit à un autre endroit du programme.

};

Ainsi, pour créer un point de type *Point3D*, on écrira :

*Point3D M* ; // utilisation du premier constructeur
*Point3D M'*$(1,3,2)$ ; // utilisation du second constructeur
*Point3D M"*$(M)$ ; // utilisation du constructeur par copie
*Point3D* pM* = ***new*** *Point3D*$(2,4,6)$ ; // Allocation dynamique avec le second constructeur (*pM* : pointeur sur *Point3D*)
***delete*** *pM* ; // Utilisation explicite du destructeur : libération de l'espace mémoire pointé par *pM* .

Remarques :

- En C++, il est possible de spécifier des valeurs initiales à prendre pour les arguments directement au niveau de la déclaration d'une méthode, ces valeurs étant en fait des valeurs par défaut. Ce type de déclaration permet lors de l'instantiation d'un objet, d'omettre la liste des arguments dont la valeur initiale correspond la valeur spécifiée par défaut. Par exemple, au lieu d'utiliser deux constructeurs pour la classe *Point3D* précédente, nous pouvions également définir le constructeur unique suivant :

  *Point3D* (***float*** *a=0,* ***float*** *b=0,* ***float*** *c=0*) $\{x = a; y = b; z = c;\}$

  Ainsi, la déclaration « *Point3D M* ; » est équivalente à écrire : *Point3D M* $(0,0,0)$ ;

- Le langage permet à toute fonction non membre d'une classe, mais déclarée comme ***friend***, d'accéder directement aux composants privés de la classe. Ceci est par exemple utile lorsque l'on souhaite définir une méthode commune à deux classes différentes.

## A.4 GENERICITE ET CONTROLE DE TYPE

C++ permet de définir une famille de classes à partir d'une classe paramétrée appelée classe générique ou « patron » (***template***). Un tel paramétrage permet au programmeur d'énoncer toutes les fonctionnalités attendues pour la réalisation de la classe. Ces fonctionnalités ne seront vérifiées qu'au moment de la compilation, lors de l'instanciation effective de la classe, i.e. lorsque les types, valeurs et fonctions sur lequel le patron doit être appliqué, seront précisés.

Par ailleurs, C++ permet une grande souplesse au niveau des conversions de type (*cast*). La conversion d'un objet d'une classe dans une autre est utilisée notamment lorsqu'il s'agit de restreindre la visibilité de certains composants de la classe de départ ou l'accès à certaines de ses fonctionnalités (par exemple, conversion d'un objet d'une classe fille vers une classe mère). Le programmeur peut par ailleurs définir explicitement les opérations de conversion pour les autres cas désirés.

Les conversions concernant les pointeurs sont encore plus souples : il est en effet possible de convertir un pointeur vers un type donné en un pointeur vers un autre type, quel qu'il soit.

## A.5 HERITAGE ET POLYMORPHISME

L'héritage est une propriété essentielle des langages orientés objets, puisqu'il consiste à définir l'ensemble des classes générées (classes filles) à partir d'une ou plusieurs classes données (classes mères). Une classe fille hérite de tous les attributs et méthodes non privés de sa ou ses classes mères, et enrichit généralement cet héritage en apportant ses propres spécificités. Par analogie à la généalogie humaine, on choisit généralement de représenter la notion d'héritage sous forme d'arbre, appelé *arbre d'héritage*, dans lequel la racine est constituée par la classe la plus primitive (niveau le plus élevé) et les sous-niveaux représentant les diverses classes dérivées (*cf*. l'annexe B).

Le polymorphisme est également une notion importante pour les langages orientés objets. Il consiste à considérer une classe mère comme un ensemble incluant toutes ses classes filles (*cf*. Fig. 1). Ainsi, un objet de la classe fille *F* peut être converti directement en un objet d'une de ses classes mères *M* et inversement, tout objet de type *M* peut désigner un objet de type *F* et utiliser ainsi les attributs et méthodes non privés associés. Ceci ne peut néanmoins se faire qu'à condition de déclarer les attributs et méthodes en question comme *virtuels* (ou différés) au niveau de la classe mère *M*.

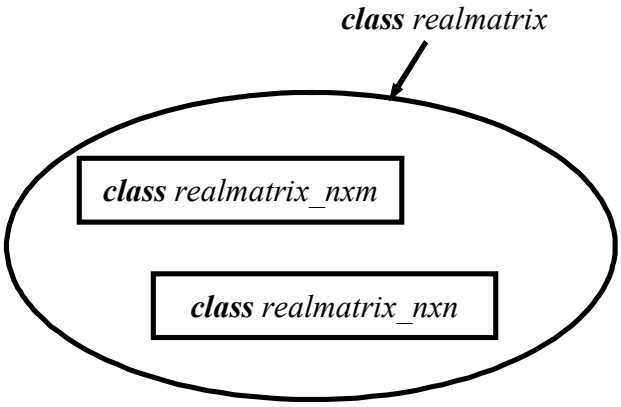



***Fig. 1 : Visualisation du polymorphisme***

## A.6 SURCHARGE DES OPERATEURS ET DES METHODES

D'une manière générale, C++ autorise la surcharge ou la modification de la plupart des fonctions ou opérations, qu'elles soient prédéfinies ou non par le langage. Cette technique est utilisée couramment au niveau de l'implémentation des classes dérivées, pour modifier l'héritage des méthodes d'une classe ancêtre. Elle est également utilisée pour redéfinir les opérateurs arithmétiques de base (« + », « - », « * », « / », etc.) et les opérateurs booléens pour qu'ils s'appliquent à des objets de types nouveaux (classes définies par le programmeur). On peut par exemple surcharger l'opérateur « + » pour pouvoir l'utiliser avec deux matrices

réelles *M* et *M'* (classe ici notée *realmatrix*), en procédant à une addition matricielle classique :

*realmatrix* ***operator****+(****const*** *realmatrix& M,* ***const*** *realmatrix& M')*;

Cette technique permet ainsi de définir plusieurs fonctions ou opérations homonymes se distinguant par les types de leurs paramètres, facilitant de ce fait la tâche de l'utilisateur. Par ailleurs, ce procédé permet d'avoir un meilleur contrôle sur les opérations erronées ou non autorisées.

## A.7 STRUCTURES DE CONTROLE

Les structures de contrôle comprennent[6] :

*le test « si ... alors… » :*

***if*** (expression booléenne) { blocs d'instructions ; }
[***else if*** { blocs d'instructions ; }]

*les boucles :*

*« pour ... faire… » :*
***for***([déclaration type] indice = valeur initiale[,(autre [déclaration type] indice = valeurs initiales,…] ; expression booléenne (arrêt) ; instructions)
[{ blocs d'instructions ; }]

*« tant que ... faire… » :*
***while*** (expression booléenne) [{instructions ;}]

*« faire…tant que… » :*
***do*** { instructions ; } ***while***(expression booléenne) ;

*le sélecteur « Pour variable = valeur, faire… » :*

```
switch(variable) {
        case valeur : { blocs d'instructions ; }
                [break ;]
        [case  autres valeurs : { blocs d'instructions ; }
                [break ;] ]
         [default : { blocs d'instructions ; }
                [break ;] ]
                }
```

Enfin, comme dans le cas des langages FORTRAN et C, C++ autorise l'utilisation du mot clé ***goto*** associé à une étiquette ou label, pour atteindre directement une ligne précise du programme dans une fonction donnée, mais cette pratique devrait néanmoins être proscrite dans ce type de langage.

[6] Les crochets […] indiquent des expressions facultatives.

# Annexe B

# Structure de données du logiciel *MoveFree*

## B.1 INTRODUCTION

Cette section a pour objet de présenter l'organisation générale de la structure de données existant dans le logiciel *MoveFree*, en indiquant la hiérarchie des diverses classes et en fournissant une description détaillée des attributs et méthodes d'un certain nombre de ces classes. D'une manière générale, nous ne redonnons pas au niveau d'une classe dérivée, les attributs et méthodes héritées de la classe mère (y compris les méthodes virtuelles effectivement implémentées à ce niveau), mais uniquement celles apportées spécifiquement par la classe fille.

## B.2 PRINCIPE D'ORGANISATION

Nous avons choisi d'organiser la structure de données du logiciel suivant la démarche habituelle de modélisation, qui consiste à décrire le système réel par l'intermédiaire de différentes composantes ou modèles théoriques le simplifiant. Pour notre part, nous avons considéré les trois modèles théoriques suivants :

- le *modèle géométrique* : idéalisation ou simplification du domaine réel à modéliser ;
- le *modèle mécanique* : formulation mathématique utilisée pour le problème aux limites ;
- le *modèle numérique* : approximation discrète des modèles mécanique et géométrique (formulation variationnelle).

Lors de l'élaboration de la structure de données informatique de *MoveFree*, nous nous sommes attaché à suivre cette démarche, tout en permettant les communications et interactions entre les différents modèles théoriques.

Nous présentons ci-après la description et les arbres d'héritage pour les principaux objets informatiques utilisés, tels les objets mathématiques de base ou ceux pour la modélisation, ces derniers étant regroupés suivant les modèles théoriques énoncés ci-dessus. Par ailleurs, nous indiquons à titre d'exemple, l'implémentation détaillée de quelques classes mathématiques.

## B.3 LES TYPES ENUMERES

D'une manière générale, un type énuméré permet de restreindre les valeurs que peut prendre une variable du même type. Dans *MoveFree*, un certain nombre de types énumérés ont été définis, essentiellement pour contrôler la communication entre les objets. Suivant le niveau de communication visé, la définition peut être globale ou locale, c'est-à-dire à l'intérieur ou à l'extérieur de la déclaration d'une classe donnée. Par exemple, pour simplifier la communication entre des objets appartenant à des modèles théoriques différents, on préfère utiliser des types énumérés définis au niveau global (*cf*. Tabl. 2). Pour la communication spécifique entre des objets d'une même généalogie, on préfère en revanche utiliser la définition locale (*cf*. Tabl. 3).

| Types énumérés globaux | Description |
|---|---|
| *algoKind* | Définition de la méthode de résolution des systèmes non linéaires (par exemple : Newton-Raphson modifiée ou non) |
| *analysisGeom* | Géométrie de l'analyse : 1D, 2D (déformations ou contraintes planes, analyse axisymétrique) ou 3D |
| *analysisKind* | Type d'analyse : statique ou dynamique en transformations infinitésimales ou finies |
| *analysisStress* | Calcul mécanique en contraintes totales (non drainés uniquement) ou effectives |
| *cellKind* | Type de cellules (intégration numérique ou éléments finis) |
| *fieldKind* | Type de champ :<br>- cinématique du solide (déplacement, vitesse, accélération),<br>- fluides (pressions, flux),<br>- chargements (forces ponctuelles, 1inéiques, forces de surface/volume),<br>- contraintes ou déformations du solide, etc. |
| *funcKind* | Fonctions de pondération ou lissage. |
| *integKind* | Définition de la méthode d'intégration numérique utilisée (Gauss, « bucket » ou nodale) |
| *lawKind* | Types de lois de comportement disponibles (Hooke, Hujeux, etc.) |
| *nodeKind* | Méthode numérique utilisée (définition au niveau des nœuds) :<br>- FEM, EFGM, EFG-FEM, h-p Clouds,<br>- RKPM, RKPM-FEM, RKI, RKI-FEM, MLS/RK,<br>- C-PUM,<br>- PIM, RPIM, MLS/RBF. |
| *pointKind* | Types de points pour les calculs : point d'intégration, nœud ou point sur lequel sont définis des multiplicateurs de Lagrange |
| *problemKind* | Cinématique du problème (analyse statique ou dynamique) |
| *saturState* | Etat de saturation au niveau d'un point ou d'un sous-domaine du domaine d'étude (valeurs : sec, saturé, non saturé, multi-phase) |

***Tabl. 2 : Exemples de types énumérés globaux***

| Types énumérés locaux | Description |
|---|---|
| *weightfunc::supportKind* | Définition du type de support d'une fonction de pondération (radial ou rectangulaire) |
| *load::loadKind* | Définition du type de courbe de chargement (constante, à pas constant ou à pas variable) |

***Tabl. 3 : Exemples de types énumérés locaux***

## B.4 OBJETS MATHEMATIQUES DE BASE

Les objets mathématiques implémentés dans *MoveFree* sont les vecteurs réels ou entiers à indices entiers, reposant sur la classe générique *Vecteur*, et les matrices réelles à indices entiers, s'appuyant sur la classe mère *realmatrix*.

Nous n'avons pour le moment par cherché à optimiser les aspects concernant le stockage des matrices en mémoire (stockage profil, etc.). En effet, tous les éléments, y compris les éléments nuls, sont actuellement stockés (également vrai pour les vecteurs).

### B.4.1 La classe *Vecteur*

Cette classe générique nous permet de définir des vecteurs de type *T* quelconque (*cf*. Fig. 2), *T* désignant aussi bien des objets que des types numériques fondamentaux. Les classes dérivées que nous avons implémentées à partir de cette classe, utilise essentiellement ces derniers types.

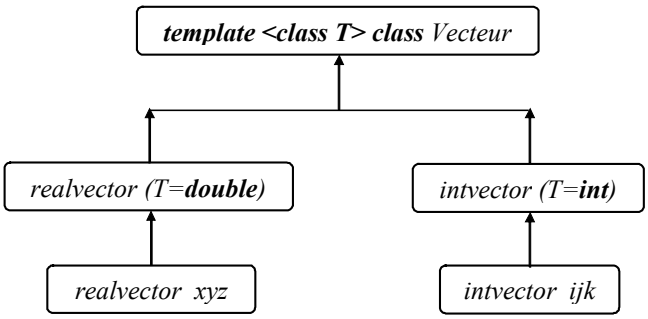



***Fig. 2 : Hiérarchie de la classe « vecteur »***

| | |
|---|---|
| *Déclaration C++ :* | ***template <class T> class** Vecteur* |
| *Données membres protégées :* | *Structure interne de stockage de l'objet vecteur (pointeur sur objet de type T) et nombre d'éléments dans la structure (entier naturel).* |
| *Constructeurs :* | |
| *Vecteur (int = 0)* | *Crée un objet vecteur de type T quelconque.*<br>*⇨ Argument : dimension d'allocation du vecteur* |
| *Vecteur (const Vecteur&)* | *Crée un objet vecteur de type T par copie d'un autre objet vecteur.*<br>*⇨ Argument : vecteur à copier* |
| *Destructeur :* | |
| *~Vecteur ()* | *Détruit cet objet (méthode virtuelle).* |
| *Attributs et méthodes publiques :* | |
| *GetSize ()* | *Donne la dimension de ce vecteur.*<br>*⇨ Retour : entier naturel* |
| *SetSize (int)* | *Redimensionne ce vecteur.*<br>*⇨ Argument : nouvelle dimension* |
| *data ()* | *Donne accès à la structure interne de stockage de ce vecteur.*<br>*⇨ Retour : pointeur la structure (objet de type T*)* |
| *operator[] (int)*<br>*value_at (int)* | *Donne accès à un élément de ce vecteur pour le manipuler.*<br>*⇨ Argument : indice de l'élément*<br>*⇨ Retour : élément non modifiable (type T)* |
| *operator[] (int)* | *Donne accès à un élément de ce vecteur pour modifier sa valeur.*<br>*⇨ Argument : indice de l'élément*<br>*⇨ Retour : référence sur l'élément (objet de type T&)* |
| *value_at (int, const T&)* | *Modifie la valeur d'un élément de ce vecteur (méthode virtuelle).*<br>*⇨ Argument : indice de l'élément et nouvelle valeur* |
| *add (T)* | *Ajoute un élément à la fin de ce vecteur.*<br>*⇨ Argument : élément à ajouter*<br>*⇨ Retour : indice de l'élément ajouté (entier naturel)* |
| *InsertAt (int, const T&)* | *Insert un élément dans ce vecteur.*<br>*⇨ Arguments : indice d'insertion et élément à insérer*<br>*⇨ Retour : indice de l'élément inséré (entier naturel)* |
| *is_empty ()* | *Vérifie si ce vecteur est vide.*<br>*⇨ Retour : booléen* |
| *copy (const Vecteur&)* | *Remplace ce vecteur par la copie d'un autre vecteur réel.*<br>*⇨ Argument : vecteur à copier* |
| *init (int)* | *(Re-)Initialise ce vecteur.*<br>*⇨ Argument : nouvelle dimension* |
| *clear ()* | *Annule toutes les composantes de ce vecteur.* |
| *destroy ()* | *Libère l'espace mémoire alloué pour ce vecteur.* |
| *add_at (int, const T&)* | *Ajoute une valeur à un élément de ce vecteur.*<br>*⇨ Arguments : indice de l'élément et valeur à lui ajouter (type T)* |
| *x_at (int, const T&)* | *Multiplie un élément de cet objet par une valeur donnée.*<br>*⇨ Arguments : indice de l'élément et valeur à lui multiplier (type T)* |

### B.4.2 La classe *realvector*

Cette classe correspond à la classe « vecteur réel » de l'espace euclidien (dimension quelconque). Elle hérite de tous les attributs et méthodes de la classe mère générique et implémente un certain nombre de méthodes spécifiques que nous décrivons ci-après. Par ailleurs, nous ne rappelons pas les méthodes virtuelles de la classe mère devant être implémentées à ce niveau.

| | |
|---|---|
| *Déclaration C++ :* | ***class*** *realvector* : ***public*** *Vecteur* ***<double>*** |
| *Constructeurs:* | |
| *realvector (int, double*)* | *Crée un objet vecteur réel par copie d'un certain nombre d'éléments pris dans un tableau de réels (éléments pris à partir du début du tableau).*<br>⇨ *Argument : dimension d'allocation du vecteur et tableau de réels à copier* |
| *realvector (const realvector&)* | *Crée un objet vecteur réel par copie d'un autre objet vecteur réel.*<br>⇨ *Argument : vecteur réel à copier* |
| *realvector (int, const realvector&)* | *Crée un objet vecteur réel par copie d'un certain nombre d'éléments pris dans un autre objet vecteur réel (à partir du début du vecteur).*<br>⇨ *Argument : dimension d'allocation du vecteur, i.e. nombre de termes et vecteur réel utilisé pour la copie* |
| *Attributs et méthodes publiques :* | |
| *tol_is (const double&)* | *Annule les éléments dont la valeur est strictement inférieure à une tolérance donnée.*<br>⇨ *Argument : valeur de tolérance* |
| *unit ()* | *Rend ce vecteur unitaire.* |
| *x_tensor (const realvector&)* | *Effectue le produit tensoriel avec un autre vecteur réel.*<br>⇨ *Retour : matrice rectangulaire résultante (type realmatrix_nxm)* |
| *operator=* | *Remplace ce vecteur par un autre objet vecteur réel.*<br>⇨ *Argument : vecteur à assigner (type realvector)*<br>⇨ *Retour : référence sur le vecteur modifié (type realvector&)* |
| *operator-()* | *Renvoie l'opposé de ce vecteur réel (opérateur unaire)*<br>⇨ *Retour : référence sur le vecteur modifié (type realvector&)* |
| *operator <=, operator >=, operator <, operator <* | *Compare les composantes de ce vecteur réel à une valeur donnée.*<br>⇨ *Argument : valeur pour la comparaison (réel)*<br>⇨ *Retour : booléen* |
| *Méthodes globales :* | |
| *dist (const realvector&, const realvector&)* | *Calcule la distance entre deux vecteurs réels.*<br>⇨ *Retour : réel* |
| *Prod_scal (const realvector&, const realvector&)* | *Calcule le produit scalaire entre deux vecteurs réels.*<br>⇨ *Retour : réel* |
| *abs (const realvector&)* | *Calcule la valeur absolue des éléments d'un vecteur réel.*<br>⇨ *Retour : vecteur résultant (type realvector)* |
| *concat (const realvector&, const realvector&)* | *Concatène deux vecteurs réels.*<br>⇨ *Retour : vecteur résultant (type realvector)* |
| *norm_max (const realvector&)* | *Calcule la valeur absolue de la borne supérieure sur l'ensemble des éléments d'un vecteur réel.*<br>⇨ *Retour : réel* |
| *norm (const realvector&)*<br>*norm_sqr (const realvector&)* | *Calcule la norme euclidienne et la norme carrée d'un vecteur réel.*<br>⇨ *Retour : réel* |
| *operator ==*<br>*operator !=* | *Teste l'égalité ou non sur deux vecteurs réels.*<br>⇨ *Retour : booléen* |

| | |
|---|---|
| *operator +*<br>*operator -* | *Additionne ou soustrait deux vecteurs réels.*<br>⇨ *Retour : vecteur résultant (type realvector)* |
| *operator **<br>*operator /* | *Multiplie ou divise deux vecteurs réels termes à termes.*<br>⇨ *Retour : vecteur résultant (type realvector)* |
| *operator ** | *Multiplie un vecteur réel par une valeur donnée (produit à droite ou à gauche).*<br>⇨ *Arguments : vecteur réel et valeur (réel)*<br>⇨ *Retour : vecteur résultant (type realvector)* |
| *operator /* | *Divise un vecteur réel par une valeur donnée.*<br>⇨ *Arguments : vecteur réel et valeur (réel)*<br>⇨ *Retour : vecteur résultant (type realvector)* |
| *operator <<* | *Lecture d'un vecteur réel à partir d'un objet stream (par exemple un fichier)*<br>⇨ *Arguments : références sur le stream (type istream&) et le vecteur réel lus*<br>⇨ *Retour : référence sur le stream lu (type istream&)* |
| *operator >>* | *Ecriture d'un vecteur réel sur un objet stream.*<br>⇨ *Arguments : référence sur le stream modifié (type ostream&) et vecteur réel à écrire*<br>⇨ *Retour : référence sur le stream modifié (type ostream&)* |

### B.4.3 La classe *realvector_xyz*

Cette classe correspond à la classe « vecteur réel » de dimension 3 (déclaration C++ : ***class*** *realvector_xyz* : ***public*** *realvector*). Elle hérite de tous les attributs et méthodes de la classe *realvector* et nous fournissons ci-après, uniquement les méthodes spécifiques à cette classe (pas de rappel des méthodes virtuelles héritées et effectivement implémentées ici).

| | |
|---|---|
| *Constructeurs:* | |
| *realvector_xyz (double, double = 0, double = 0)* | *Crée un objet vecteur réel 3D en fournissant ses 3 composantes.* |
| *realvector_xyz (const realvector_xyz&)* | *Crée un objet vecteur réel 3D par copie d'un autre objet vecteur réel 3D.*<br>⇨ *Argument : vecteur réel à copier* |
| *realvector_xyz (const realvector&)* | *Crée un objet vecteur réel 3D par copie d'un objet vecteur réel.*<br>⇨ *Argument : dimension d'allocation du vecteur, i.e. nombre de termes (entier naturel) et vecteur réel à copier (type realvector)* |
| *Attributs et méthodes publiques :* | |
| *is_colinear_to (const realvector_xyz&)* | *Evalue si ce vecteur est colinéaire à un autre vecteur.*<br>⇨ *Retour : booléen* |
| *max_is ()*<br>*min_is ()* | *Calcule la borne supérieure ou inférieure des composantes de ce vecteur.*<br>⇨ *Retour : réel* |
| *Méthodes globales :* | |
| *Prod_vect (const realvector&, const realvector&)* | *Calcule le produit vectoriel entre deux vecteurs réels 3D.*<br>⇨ *Retour : réel* |
| *Coord_max ()*<br>*Coord_min ()* | *Fournit le vecteur réel dont chaque composante est calculée en prenant la valeur maximale ou minimale des composantes de deux vecteurs réels 3D.*<br>⇨ *Retour : vecteur réel 3D résultant* |

### B.4.4 Les classe *intvector* et *intvector_ijk*

Ces deux classes que nous ne détaillerons pas ici, correspondent respectivement aux classes « vecteur entier » et « triplet d'entiers ». Nous utilisons cette dernière classe essentiellement pour définir les différents repères (global ou locaux) d'une analyse. Par exemple, on notera (0,1,-1) le triplet correspondant au plan OXY du repère global de l'espace.

Ces classes héritent de tous les attributs et méthodes de la classe mère générique *Vecteur* et sont déclarées de la manière suivante en C++ :

***class*** *intvector* : ***public*** *Vecteur* ***<int>***

***class*** *intvector_ijk* : ***public*** *intvector*

### B.4.5 La classe *realmatrix*

Cette classe constitue la classe de base pour définir tout type de matrice réelle (*cf*. Fig. 3). Le stockage des éléments est réalisé en utilisant un objet de type « vecteur réel » (dimension quelconque). Pour accéder à un élément $a_{ij}$ donné de la matrice (ligne $i$, colonne $j$), il est nécessaire d'utiliser la méthode virtuelle *index_is* qui renvoie l'indice correspondant à la position réelle de l'élément dans le vecteur de stockage, suivant la classe dérivée réellement instantiée (*cf*. Fig. 4 pour un exemple de calcul pour une matrice réelle rectangulaire).

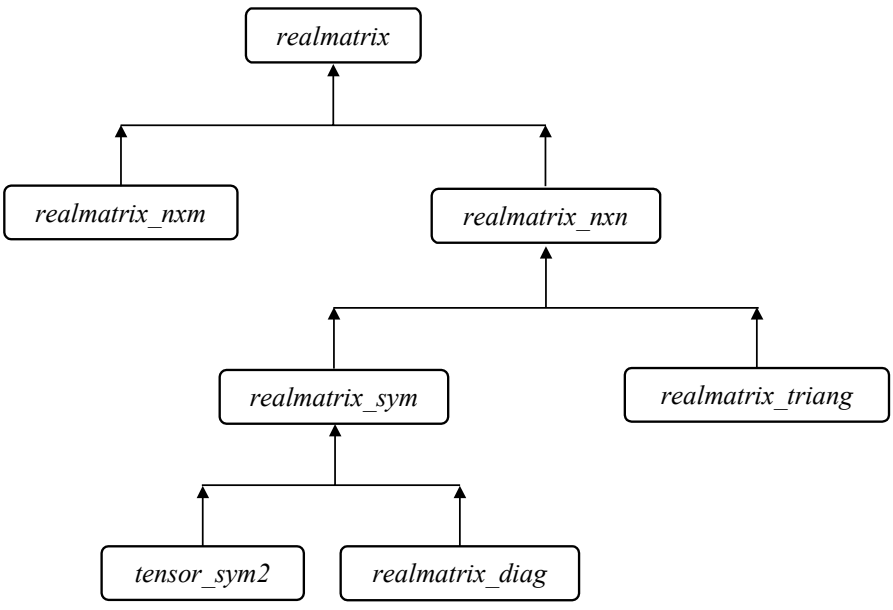



***Fig. 3 : Hiérarchie de la classe « matrice réelle »***

| | |
|---|---|
| *Données membres protégées :* | *Structure interne de stockage de l'objet matrice (objet de type realvector).* |
| *Constructeurs:* | |
| *Realmatrix (int = 0, int = 0)* | *Crée un objet matrice réelle.*<br>*⇨ Arguments : nombre de lignes et de colonnes de la matrice* |
| *realmatrix (const realmatrix&)* | *Crée un objet matrice réelle par copie d'un autre objet matrice réelle.*<br>*⇨ Argument : objet matrice réelle à copier* |
| *Destructeur:* | |
| *~ realmatrix ()* | *Détruit cet objet (méthode virtuelle).* |
| *Attributs et méthodes publiques :* | |
| *GetSize_col ()*<br>*GetSize_row ()* | *Donne le nombre de colonnes ou de lignes de cet objet matrice (méthodes virtuelles).*<br>*⇨ Retour : entier naturel* |
| *SetSize (int, int)* | *Redimensionne cet objet en fournissant les nouvelles dimensions.* |
| *size_col (int)*<br>*size_row (int)* | *Modifie respectivement le nombre de colonnes ou de lignes de cet objet (méthodes virtuelles).*<br>*⇨ Argument : nouvelle dimension* |
| *data ()* | *Donne accès à la structure interne de stockage de cet objet.*<br>*⇨ Retour : pointeur sur un vecteur réel (type realvector*)* |
| *operator() (int, int)*<br>*value_at (int, int)* | *Donne accès à un élément de la matrice pour le manipuler (méthodes virtuelles).*<br>*⇨ Argument : indices des ligne et colonne de l'élément*<br>*⇨ Retour : valeur de l'élément (réel)* |
| *operator() (int, int)* | *Donne accès à un élément de la matrice pour modifier sa valeur (méthode virtuelle).*<br>*⇨ Argument : indices des ligne et colonne de l'élément*<br>*⇨ Retour : référence sur l'élément à modifier (type double&)* |
| *value_at (int,int ,const double&)* | *Modifie directement la valeur d'un élément de la matrice (méthode virtuelle).*<br>*⇨ Arguments : indices des ligne et colonne de l'élément et nouvelle valeur (réel)* |
| *row_at (int)*<br>*col_at (int)* | *Donne accès à une ligne ou une colonne de la matrice (méthodes virtuelles).*<br>*⇨ Arguments : indice de la ligne ou de la colonne*<br>*⇨ Retour : vecteur réel résultant (type realvector)* |
| *row_at (int, const realvector&)*<br>*col_at (int, const realvector&)* | *Modifie directement une ligne ou une colonne de la matrice (méthodes virtuelles).*<br>*⇨ Arguments : indice de la ligne ou de la colonne et nouvelles valeurs (type realvector)* |
| *determinant ()* | *Calcule le déterminant de la matrice (méthode virtuelle).*<br>*⇨ Retour : réel* |
| *is_empty ()* | *Vérifie si la matrice est vide.*<br>*⇨ Retour : booléen* |
| *type ()* | *Indique le type de cette matrice (rectangulaire, carrée, etc.).*<br>*⇨ Retour : type énuméré matriceKind* |
| *clear ()* | *Annule toutes les valeurs de la matrice.* |
| *destroy()* | *Libère l'espace mémoire alloué pour la matrice.* |
| *add_at (int, int, const double&)* | *Ajoute une valeur à un élément de la matrice (méthode virtuelle).*<br>*⇨ Arguments : indices de l'élément et valeur à lui ajouter (réel)* |
| *swap_at (int, int)* | *Echange de deux éléments de la matrice (méthode virtuelle).*<br>*⇨ Arguments : indices des ligne et colonne à échanger* |
| *x_at(int, int, const double&)* | *Multiplie un élément de la matrice par une valeur donnée (méthode virtuelle).* |

| | |
|---|---|
| | ⇨ *Arguments : indices de l'élément et valeur à lui multiplier (réel)* |
| *index_is (int, int)* | *Donne la position d'un élément dans le vecteur de stockage de la matrice réelle (méthode virtuelle).*<br>⇨ *Arguments : indices des ligne et colonne de l'élément*<br>⇨ *Retour : indice dans le vecteur de stockage (entier naturel)* |
| *Méthodes globales :* | |
| *norm_max (const realmatrix&)* | *Calcule la valeur absolue de la borne supérieure sur les éléments d'une matrice réelle.*<br>⇨ *Retour : réel* |
| *trace (const realmatrix&)* | *Calcule la trace d'une matrice réelle.*<br>⇨ *Retour : réel* |
| *Surcharge des opérateurs :* | |
| *operator ==*<br>*operator !=* | *Teste l'égalité de deux matrices réelles.*<br>⇨ *Retour : booléen* |
| *operator *, operator +*<br>*operator -* | *Multiplie, additionne ou soustrait deux matrices réelles.*<br>⇨ *Retour :matrice réelle résultante* |
| *operator ** | *Multiplie une matrice réelle par un vecteur réel (à droite ou à gauche).*<br>⇨ *Arguments : matrice et vecteur réels*<br>⇨ *Retour : vecteur réel résultant* |
| *operator ** | *Multiplie une matrice réelle par une valeur donnée (à droite ou à gauche).*<br>⇨ *Arguments : matrice réelle et valeur (réel)*<br>⇨ *Retour :matrice réelle résultante* |
| *operator /* | *Divise une matrice réelle par une valeur donnée.*<br>⇨ *Arguments : matrice réelle et valeur (réel)*<br>⇨ *Retour :matrice réelle résultante* |
| *operator <<* | *Lecture d'une matrice réelle à partir d'un objet stream (par exemple un fichier)*<br>⇨ *Arguments : référence sur le stream (type istream&) et pointeur sur la matrice réelle lus*<br>⇨ *Retour : référence sur le stream lu (type istream&)* |
| *operator >>* | *Ecriture d'une matrice réelle sur un objet stream.*<br>⇨ *Arguments : référence sur le stream modifié (type ostream&) et pointeur sur la matrice réelle à écrire*<br>⇨ *Retour : référence sur le stream modifié (type ostream&)* |

$$\underbrace{\begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{1n} & \cdots & a_{nm} \end{pmatrix}}_{realmatrix\_nxm} \Rightarrow \underbrace{\begin{bmatrix} b_1 = a_{11} \\ \vdots \\ b_k = a_{ij} \\ \vdots \\ b_N = a_{nm} \end{bmatrix}}_{\text{vecteur réel de stockage}} \quad \Rightarrow k = index\_is(i,j) = (i-1)\times m + j$$

***Fig. 4 : Exemple de calcul de l'indice de l'élément $a_{ij}$ dans le vecteur servant au stockage de la classe « matrice réelle rectangulaire ».***

### B.4.6 Les classes *realmatrix_nxm* et *realmatrix_nxn*

Ces classes correspondent aux types « matrice réelle rectangulaire » et « matrice réelle carrée ». Elles héritent de tous les attributs et méthodes de la classe *realmatrix* (déclarations C++ : ***class*** *realmatrix_nxm* : ***public*** *realmatrix* et ***class*** *realmatrix_nxn* : ***public*** *realmatrix*). La classe *realmatrix_nxm* possède un argument supplémentaire indiquant le nombre de colonnes de la matrice.

Nous ne fournissons ci-après, que les attributs et méthodes spécifiques à la classe *realmatrix_nxn*. Comme pour les autres classes, nous ne rappelons pas les méthodes virtuelles de la classe mère devant être implémentées à ce niveau.

| | |
|---|---|
| *Données membres privées :* | *Indicateur de décomposition LU (booléen) et vecteur contenant les indices utilisés pour la décomposition (type intvector).* |
| *Constructeurs:* | |
| *realmatrix_nxn (int = 0, double** = 0)* | *Crée un objet matrice réelle carrée par copie d'un certain nombre d'éléments pris dans un tableau de réels à deux dimensions (éléments pris à partir du début du tableau).*<br>*⇨ Argument : dimension de la matrice et tableau de réels à copier* |
| *realmatrix_nxn (const realmatrix_nxn&)* | *Crée un objet matrice réelle carrée par copie d'un autre objet matrice réelle carrée.*<br>*⇨ Argument : objet matrice réelle carrée à copier* |
| *Attributs et méthodes publiques :* | |
| *is_lu ()* | *Indique si la matrice est décomposée sous forme LU (booléen).* |
| *operator=* | *Remplace cette matrice par un autre objet matrice réelle (si non carrée, on ne conserve que la partie de dimension égale à la dimension minimale lignes/colonnes).*<br>*⇨ Argument : matrice réelle à assigner*<br>*⇨ Retour : référence sur la matrice réelle carrée modifiée (realmatrix_nxn&)* |
| *Méthodes globales :* | |
| *transpose (const realmatrix_nxn&)* | *Transpose une matrice réelle carrée.*<br>*⇨ Retour :matrice transposée (type realmatrix_nxn)* |
| *inverse (const realmatrix_nxn&)* | *Calcule l'inverse d'une matrice réelle carrée.*<br>*⇨ Retour :matrice inverse (type realmatrix_nxn)* |
| *lu_dcmp(realmatrix_nxn&)* | *Décompose la matrice sous forme LU (résolution de système linéaire).*<br>*⇨ Argument : référence sur la matrice à décomposer (type realmatrix_nxn&)*<br>*⇨ Retour : booléen indiquant la réussite de l'opération* |
| *lu_sb(realmatrix_nxn&, realvector&)* | *Calcule la solution du système linéaire AX = B (décomposition LU).*<br>*⇨ Arguments : références sur la matrice du système à résoudre et le vecteur réel contenant le 2nd membre du système (en entrée) ou la solution (en sortie)* |

### B.4.7 Les classes *realmatrix_sym*, *realmatrix_triang* et *realmatrix_diag*

Ces classes correspondent respectivement aux classes « matrice réelle symétrique » (déclaration C++ : ***class*** *realmatrix_sym* : ***public*** *realmatrix_nxn*), « matrice réelle triangulaire » (déclaration : ***class*** *realmatrix_triang* : ***public*** *realmatrix_nxn*) et « matrice réelle diagonale » (déclaration : ***class*** *realmatrix_diag* : ***public*** *realmatrix_nxn*). Elles héritent de tous les attributs et méthodes de la classe *realmatrix_nxn*.

Seule la classe *realmatrix_triang* possède un argument supplémentaire qui indique si la matrice est triangulaire supérieure ou inférieure.

Les classes *realmatrix_diag* et *realmatrix_triang* implémentent également une méthode de conversion vers la matrice réelle carrée complète correspondante.

### B.4.8 La classe *tensor_sym2*

Cette classe correspond à la classe « tenseur symétrique réel d'ordre 2 » (dimension 3). Elle hérite de tous les attributs et méthodes de la classe *realmatrix_sym* (déclaration C++ : ***class*** *tensor_sym2* : ***public*** *realmatrix_sym*). Nous fournissons ci-après les attributs et méthodes spécifiques à cette classe.

| | |
|---|---|
| *Données membres privées :* | - *Dimension du tenseur dépendant de la dimension cinématique des calculs.*<br>- *Type d'analyse : contraintes ou déformations planes, etc. (type énuméré analysisGeom)*<br>- *Triplet d'entiers représentant les axes de l'analyse (type intvector_ijk).* |
| *Constructeurs:* | |
| *tensor_sym2 (const int& = 2,const intvector_ijk& = intvector_ijk(0,1)))* | *Crée un objet tenseur symétrique réel d'ordre 2*<br>⇨ *Arguments : dimension cinématique (2D par défaut) et axes de l'analyse (plan OXY par défaut)* |
| *tensor_sym2 (const tensor_sym2&)* | *Crée un objet tenseur symétrique d'ordre 2 par copie d'un autre tenseur du même type.* |
| *tensor_sym2 (const realvector_xyz&, const realvector_xyz&)* | *Crée un objet tenseur symétrique d'ordre 2 en fournissant les deux vecteurs réels 3D contenant les termes diagonaux et hors diagonaux.* |
| *Attributs et méthodes publiques :* | |
| *geomkind_is()* | *Indique le type d'analyse en cours (type énuméré analysisGeom)* |
| *geomkind_is(analysisGeom)* | *Modifie le type d'analyse en cours.*<br>⇨ *Argument : nouveau type* |
| *axes_of()* | *Renvoie les axes pour l'analyse en cours (type intvector_ijk).* |
| *dimension_of()* | *Renvoie la dimension cinématique de l'analyse en cours (entier naturel).* |
| *realvector_of(bool = true)* | *Convertit le tenseur sous forme de vecteur réel contenant tous les termes diagonaux, puis les termes hors diagonaux.*<br>⇨ *Argument : booléen indiquant si l'on récupère toutes les composantes (6 composantes) ou seulement les composantes correspondant à la cinématique réelle de l'analyse (par exemple, uniquement 3 composantes en déformations planes)*<br>⇨ *Retour : vecteur réel* |

| | |
|---|---|
| *diagonal_terms()*<br>*outdiagonal_terms()* | *Récupère tous les termes diagonaux ou hors diagonaux.*<br>⇨ *Retour : vecteur réel 3D* |
| *diagonal_terms(const realvector_xyz&)*<br>*outdiagonal_terms(const realvector_xyz&)* | *Modifie les termes diagonaux ou hors diagonaux.*<br>⇨ *Argument : nouvelles valeurs (type vecteur réel 3D)* |

## B.5 OBJETS RELATIFS AU MODELE GEOMETRIQUE

Trois types d'objets peuvent être recensées à ce niveau :

- la structure *Cpoint*, servant à identifier un point de l'espace au moyen d'un identifiant unique (entier) et par ses coordonnées spatiales ($x$, $y$, $z$) ;
- la classe *subdomain*, partie géométrique incluse dans le domaine d'étude simplifié, délimitée par sa frontière ;
- la classe *domain*, ensemble des sous-domaines (type collection d'objets *subdomain*).

La structure *Cpoint* participe également au modèle numérique comme classe de base (*cf.* Fig. 9). Les classes *subdomain* et *domain* participent quant à elles, aux modèles mécanique et numérique par l'intermédiaire de leurs attributs. Par exemple, à chaque sous-domaine est associé un matériau solide et éventuellement un ou plusieurs matériaux fluides, ainsi que des paramètres numériques spécifiques : le type de discrétisation spatiale (FEM, meshfree ou mixte) ; l'ordre de la base primaire et le type de support (radial ou rectangulaire) dans le cas meshfree ; le type d'intégration numérique ; les nœuds, points et éventuellement les cellules d'intégration (éléments finis ou buckets ou autres) inclus dans le sous-domaine, etc.

## B.6 OBJETS RELATIFS AU MODELE MECANIQUE

Les types d'objets regroupés à ce niveau sont essentiellement les degrés de liberté, les conditions aux limites et courbes de chargement éventuelles, ainsi que les matériaux et les lois de comportement.

### B.6.1 La classe *load*

Cette classe de base permet de définir une courbe de chargement de type fonction à une variable (généralement réelle) : $y = f(x)$. Les classes *load_const*, *load_stepped*, *load_variable* héritent de cette classe (*cf.* Fig. 5) et définissent respectivement un chargement constant, à pas constant ou à pas variable.

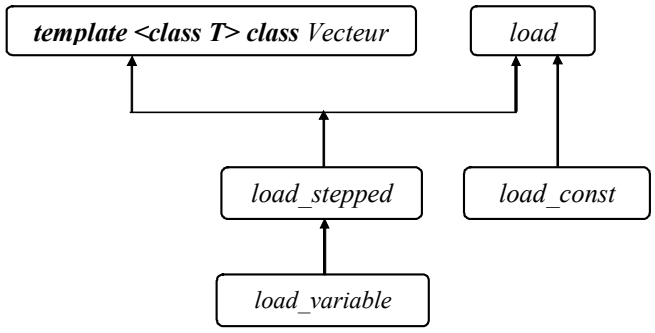



***Fig. 5 : Hiérarchie de la classe « courbe de chargement »***

### B.6.2 La classe *constitutive_law*

Cette classe (*cf.* Fig. 6) permet de définir les attributs et méthodes communes aux lois de comportement en général (pour les solides ou les fluides). Ses attributs comprennent :

- le type de loi implémentée (type énuméré *lawKind*) ;
- une structure de stockage des paramètres de la loi (y compris l'histoire éventuelle) ;
- le type d'analyse : 3D, 2D en contraintes ou déformations planes, etc. (type énuméré *analysisGeom*) : nous avons en effet considéré une seule classe globale par loi de comportement, capable de tenir compte du type d'analyse en cours ;
- un booléen indiquant si les transformations sont infinitésimales ou finies ;
- un indicateur du type de formulation finie utilisée (Lagrangienne totale ou réactualisée, etc.) ;
- une structure de stockage de la matrice globale tangente (objet de type « tenseur constitutif »).

Par ailleurs, elle fournit une méthode virtuelle permettant de calculer l'opérateur tangent et une autre pour déterminer le tenseur des contraintes résultant d'un état de déformations (et taux de déformations) donné et mettre à jour l'histoire en un point d'intégration donné.

A partir de cette classe sont dérivées les classes correspondant aux lois de comportement pour les solides et celles pour les fluides. Ces dernières ne sont pas encore implémentées dans le logiciel, mais nous envisageons notamment de compléter notre approche par des lois de type pseudoplastique et de type viscoplastique. Concernant les lois solides, seules la loi de Hooke (élasticité linéaire) et deux lois élastoplastiques pour les sols (Drücker-Prager et Hujeux) sont opérationnelles pour le moment.

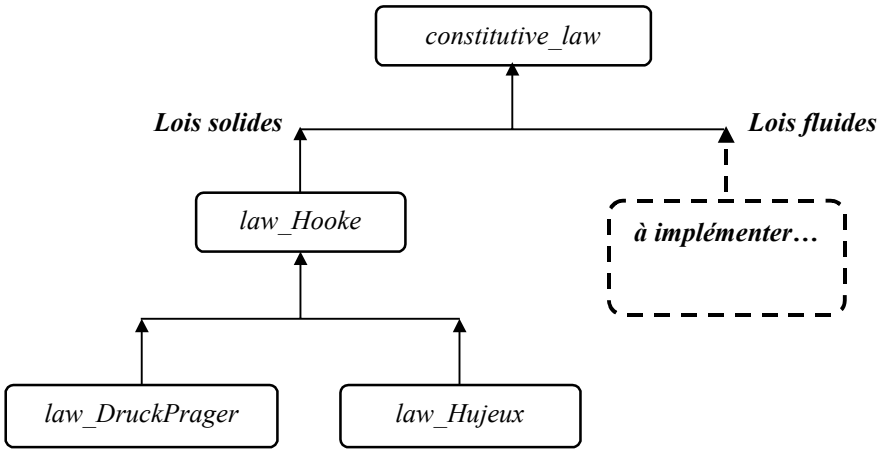



***Fig. 6 : Hiérarchie de la classe « loi de comportement »***

### B.6.3 La classe *material*

Cette classe (*cf*. Fig. 7) permet de définir les caractéristiques de base d'un matériau, qu'il soit de type solide (sol ou roche essentiellement pour nous) ou fluide. Hormis la structure de stockage des paramètres physiques du matériau, elle possède une référence sur un objet de type « loi de comportement » (type *constitutive_law**).

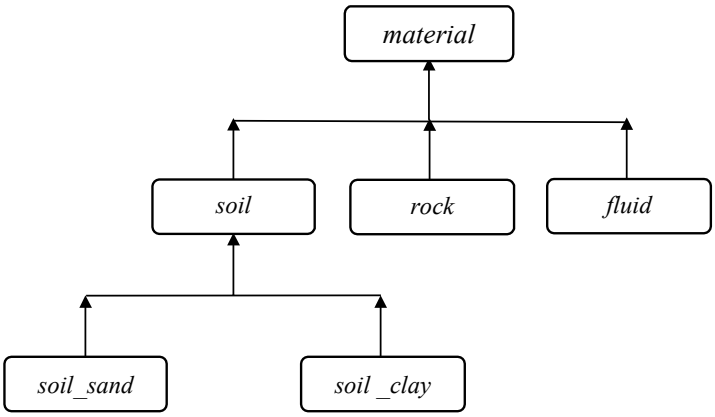



***Fig. 7 : Hiérarchie de la classe « matériau »***

### B.6.4 La classe *field*

Cette classe permet de définir un champ physique quelconque, scalaire ou vectoriel (Fig. 8). Nous l'avons utilisée comme classe de base pour les degrés de liberté et les conditions aux limites (Dirichlet ou Neumann).

Les attributs spécifiques de la classe *field* comprennent essentiellement :

- le nom du champ ;
- son type : déplacement, pression, contrainte, etc. (type énuméré *fieldKind*) ;
- son vecteur directeur : référence sur un vecteur réel 3D dans le repère global (référence nulle si le champ est scalaire) ;
- un booléen indiquant si le champ est actif ou non pour les calculs ;
- un ensemble de courbes associées à ce champ (type loads) : chargements si conditions aux limites, évolution temporelle d'un degré de liberté (imposée ou calculée), etc.

Cette classe fournit toutes les méthodes permettant d'accéder ou de modifier les attributs, ainsi que de réaliser des opérations de lecture ou d'écriture sur un stream (par exemple, un fichier).

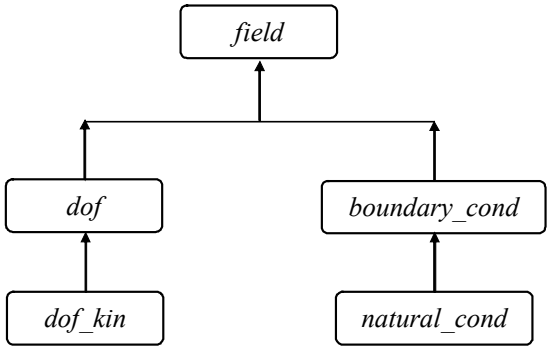



***Fig. 8 : Hiérarchie de la classe « champ »***

### B.6.5 Les classes *dof* et *dof_kin*

La classe de base *dof* permet de définir un degré de liberté quelconque (déclaration C++ : ***class*** *dof :* ***public*** *field*). Elle hérite de tous les attributs et méthodes de la classe *field*. La classe *dof_kin*, classe dérivée de *dof*, permet de définir un degré de liberté, capable de connaître ses dérivées premières et secondes, contre seulement la dérivée première pour la classe *dof*. Cette distinction est faite pour faciliter la mise en œuvre de certains schémas d'intégration numériques.

Ces classes utilisent la classe *varstep* qui permet de définir un objet « variable réelle » connaissant à tout moment, sa valeur à l'étape précédente, à l'étape ou itération courante (si calcul itératif) et à l'itération précédente (si calcul itératif).

Les attributs spécifiques de la classe *dof* comprennent essentiellement :

- un entier désignant le numéro d'équation globale ;
- des objets de type varstep contenant les valeurs de ce degré de liberté et de sa dérivée première, aux étapes ou itérations précédente et courante ;
- un entier pour désigner l'état du degré de liberté : imposé non nul (-1), libre (0) ou bloqué (1).

Hormis les méthodes héritées de la classe mère et celles permettant d'accéder ou de modifier les attributs, cette classe proposent des méthodes qui permettent les actions suivantes :

- imposer une valeur (réel) au degré de liberté ;
- libérer ce degré de liberté ;
- déterminer l'incrément de valeurs calculé entre les étapes précédente et courante ;
- initialiser la valeur de ce degré de liberté en début d'étape courante (méthode virtuelle) ;
- stocker la valeur de ce degré de liberté, calculée à l'étape (fin) ou itération courante à la place de celle de l'étape ou itération précédente (méthode virtuelle) ;
- fournir la valeur de la dérivée seconde de ce degré de liberté, aux étapes ou itérations précédente et courante (méthode virtuelle implémentée au niveau de la classe *dof_kin*).

### B.6.6 Les classes *boundary_cond* et *natural_cond*

La classe de base *boundary_cond* permet de définir des conditions aux limites de type Dirichlet (déclaration C++ : ***class*** *boundary_cond :* ***public*** *field*). Elle hérite de tous les attributs et méthodes de la classe *field*. La classe *natural_cond* (déclaration C++ : ***class*** *natural_cond :* ***public*** *boundary_cond*) permet de définir des conditions aux limites de type Neumann.

Ces classes utilisent la structure *condition_id* qui permet de définir un objet « appliquer une condition » dont les attributs sont : l'identifiant du nœud, point ou cellule d'application de la condition, et un vecteur réel qui suivant le type de condition, représente soit la direction spatiale d'application (condition initiale vectorielle), soit des composantes tensorielles (condition initiale en contraintes ou en déformations), ou soit des facteurs réels à appliquer aux courbes de chargement (condition aux limites).

Le seul attribut apporté par la classe *boundary_cond* est la collection d'objets « appliquer une condition » (type Vecteur d'objets condition_id).

Hormis les méthodes héritées de la classe mère et celles permettant d'accéder ou de modifier les attributs, cette classe proposent les méthodes suivantes :

- donner l'accès à un objet « appliquer une condition » de la collection ;
- fournir un vecteur réel dont les composantes contiennent les valeurs de cette condition aux limites (suivant le type de condition, nombre de composantes différent), pour un objet « appliquer une condition » et une date donnés ;
- donner l'accès à la discrétisation spatiale de la frontière (cellules) sur laquelle s'applique cette condition aux limites (méthode virtuelle) ;

- trouver l'indice d'un objet « appliquer une condition » pour un identifiant donné (méthode virtuelle) ;

## B.7 OBJETS RELATIFS AU MODELE NUMERIQUE

Les types d'objets regroupés à ce niveau sont essentiellement les points, cellules et schémas d'intégration numériques (éléments finis ou non), les nœuds, les fonctions de pondération, les méthodes de résolution (linéaires ou non), et plus largement le modèle numérique global permettant de faire communiquer tous les objets entre eux.

### B.7.1 La classe *loopPoint*

L'une des classes les plus importantes du modèle numérique est la classe générique que nous appèlerons «  point numérique ». Cette classe permet en effet de définir un point en tant que point d'intégration (par exemple point de Gauss, « *stress point* », centre d'un *bucket*, ), en tant que nœud au sens des méthodes meshfree ou éléments finis, ou également en tant que multiplicateur de Lagrange (*cf*. Fig. 9).

Ses attributs comprennent essentiellement :

- la référence sur le sous-domaine (parent) auquel appartient ce point (type *subdomain**) ;
- la référence sur la cellule contenant éventuellement ce point (type *cell**) ;
- l'état de saturation du point (type énuméré *saturState*) ;
- un booléen indiquant si ce point est actif ou non pour les calculs ;
- un booléen indiquant si ce point est sur la frontière ou à l'intérieur du domaine d'étude ;
- l'ensemble des voisins de ce point (points et/ou nœuds) ;
- l'ensemble des opérateurs matriciels (gradients de la transformation, dérivation, rigidité tangente, fonctions de forme, etc.) calculés sur un voisinage donné pour différentes configurations possibles (par exemple, suivant le type d'analyse, configurations de référence, actuelle, etc.) ;
- les différents types de tenseurs associés à ce point, suivant  le type d'analyse (contraintes de Cauchy et/ou Piola-Kirchhoff , déformations de Lagrange, Almansi, etc.) pour un certain nombre d'étapes (référence, courante, etc.) ;
- l'histoire éventuelle du matériau en ce point.

Par ailleurs, parmi les méthodes qu'elle propose, un certain nombre de méthodes virtuelles existent qui dépendent des méthodes de discrétisation spatiale (meshfree ou *FEM*) ou d'intégration (nodale, Gauss, etc.) utilisées, et qui permettent de calculer :

- l'ensemble des opérateurs matriciels en ce point, en fonction du type d'analyse (statique ou dynamique, transformations infinitésimales ou finies) ;
- le poids pour l'intégration numérique en ce point ;

- l'ensemble des voisins de ce point ;
- les valeurs de la fonction de forme et de ses dérivées spatiales associées, calculées sur le voisinage de ce point et pour différentes configurations ;
- les contributions apportées suivant le type d'analyse, par les degrés de liberté appartenant aux nœuds voisins de ce point (incréments de déformations, contraintes, etc.).

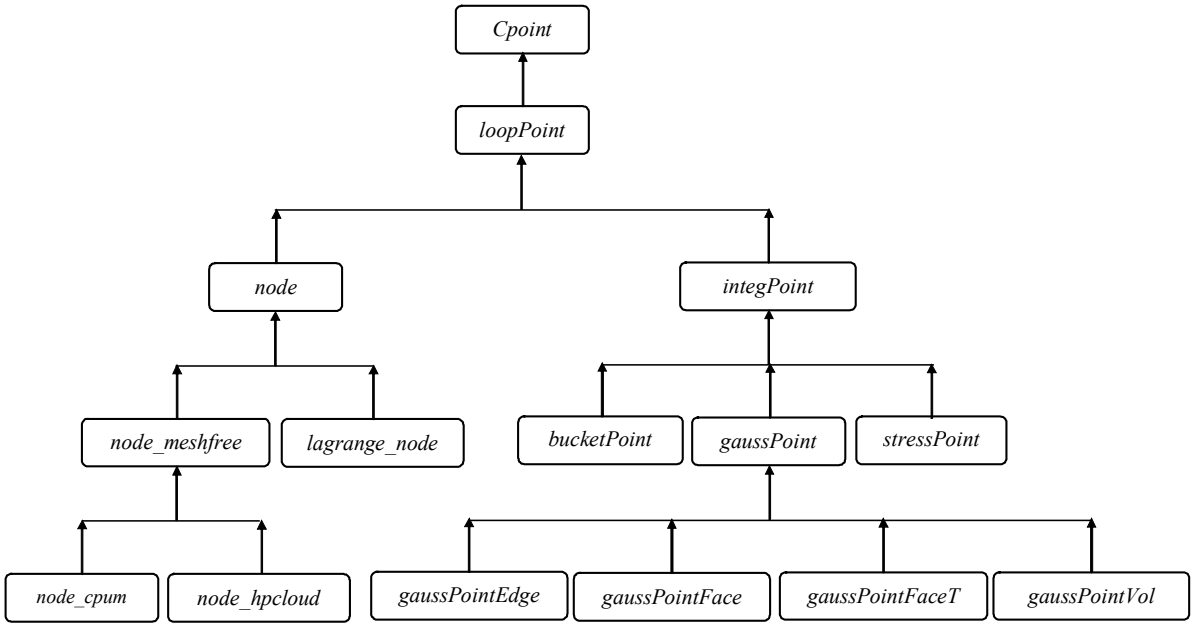



***Fig. 9 : Hiérarchie de la classe « point numérique »***

### B.7.2 La classe *node*

La classe *node* permet de définir les caractéristiques communes des objets de type « nœud » pour les méthodes meshfree et éléments finis (*FEM*). Les attributs spécifiquement apportés par cette classe sont les suivants :

- le type de méthode de dicrétisation auquel appartient ce nœud (type énuméré *nodeKind*) : la valeur prise par défaut est *FEM* ;
- une collection d'objets de type « degré de libertés » ;
- le type « mécanique » de ce nœud (type énuméré *fieldKind*) : nœud possédant des degrés de liberté uniquement solides ou fluides (analyse non couplée), ou mixtes (analyse couplée) ;
- un booléen pour indiquer si ce nœud possède une ou plusieurs conditions aux limites ;
- un booléen pour indiquer si ce nœud possède une ou plusieurs conditions initiales ;
- un booléen pour indiquer si ce nœud est à l'interface entre deux domaines utilisant des méthodes de discrétisation différentes (meshfree – *FEM*) ;
- l'ensemble des points d'intégration (Gauss, « stress point », etc.) voisins de ce nœud.

Hormis les méthodes permettant d'accéder ou de modifier les attributs, cette classe propose essentiellement des méthodes virtuelles implémentées au niveau des classes dérivées (nœuds meshfree ou *FEM*) et qui permettent de calculer :

- la valeur de la fonction de pondération ou de sa dérivée (si meshfree) pour un point d'évaluation dont le voisinage contient ce nœud ;
- la position actuelle de ce nœud suivant la valeur de ses degrés de liberté cinématiques ;
- les valeurs prédites ou corrigées de ses degrés de liberté, suivant la nature de ceux-ci et le schéma de discrétisation temporelle utilisé.

### B.7.3 La classe *cell*

Cette classe permet de définir le type générique « cellule d'intégration » (bucket, élément fini, etc.). Elle est utilisée comme élément de base d'une grille ou d'un maillage (*cf*. Fig. 10).

Ses attributs comprennent :

- un identifiant unique (entier) de la cellule ;
- la référence sur le sous-domaine (parent) contenant cette cellule (type *subdomain**) ;
- la liste des objets de type « point géométrique » associés à cette cellule (type *points_xyz*) ;
- un booléen indiquant si cette cellule est sur la frontière ou à l'intérieur du domaine d'étude ;
- un tableau d'entiers contenant les numéros des points sommets de la cellule : ces numéros sont uniques et attribués au niveau global de l'analyse, lors de la création de la grille ou la lecture du maillage ;
- un tableau d'entiers contenant éventuellement les numéros des nœuds *FEM* de la cellule : ce tableau est utilisé dans le cas d'une analyse mixte *FEM* – meshfree ;
- la liste des nœuds voisins dont le domaine d'influence contient cette cellule (type *nodes*) ;
- la liste des nœuds (type *nodes*) contenus dans cette cellule (y compris sur les arêtes) ;
- la liste des cellules voisines, *i.e.* ayant un sommet voisin avec cette cellule (type *grid*) ;
- un triplet d'entiers définissant l'ordre d'intégration numérique dans les trois directions du repère local de la cellule, utile pour l'intégration de Gauss ;

La majorité des méthodes apportées par cette classe sont virtuelles et permettent notamment les actions suivantes :

- manipuler ou modifier les attributs ;
- fournir le type de la cellule (type énuméré *cellKind*) ;
- pour une analyse 1D ou 2D, fournir la normale à la cellule dans le repère global ; dans le cas 3D, on fournit la normale à d'une face donnée ;
- fournir la boîte englobante de cette cellule ;
- pour la méthode *FEM*, calculer :
    - les fonctions de forme et leurs dérivées en tout point géométrique défini dans le repère de référence de la cellule ;

  - la matrice jacobienne, son inverse et le jacobien aux points de Gauss de la cellule ;
  - les coordonnées globales d'un point géométrique défini dans le repère de référence de la cellule.
- tester si un point géométrique appartient à cette cellule ;
- suivant la dimension géométrique de l'analyse (1D, 2D ou 3D), calculer la dimension de cette cellule (longueur, surface ou volume) ;
- fournir le barycentre de la cellule ;
- calculer la fonction de transition et sa dérivée, en tout point géométrique de la cellule (cas d'une analyse mixte *FEM* – meshfree) ;
- fournir la liste des points d'intégration de la cellule (Gauss ou autres).

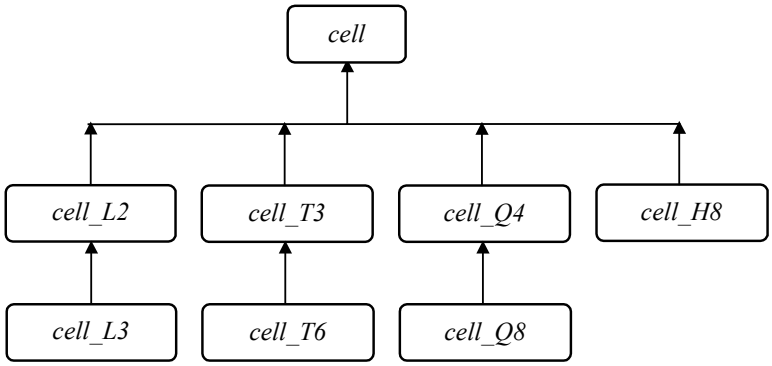



***Fig. 10 : Hiérarchie de la classe « cellule d'intégration »***

### B.7.4 La classe *weightfunc*

Cette classe permet de définir le type générique « fonction de pondération » dans le cas d'une analyse meshfree (*cf*. Fig. 11). Elle utilise un seul attribut

Son seul attribut comprend un booléen indiquant si cette fonction est interpolante ou non. Cette classe définit également le type énuméré *supportKind* (*cf*. Tabl. 3) permettant d'indiquer si le support utilisé pour cette fonction est radial ou rectangulaire.

Hormis les méthodes manipulant la variable membre, les méthodes apportées par cette classe sont virtuelles et permettent notamment les actions suivantes :

- fournir le type de la fonction (type énuméré *funcKind*) ;
- fournir le type du support (type énuméré *supportKind*) ;
- donner ou modifier la valeur du rayon du support : valeur scalaire dans le cas radial, vectorielle dans le cas rectangulaire ;
- donner ou modifier la valeur des paramètres utilisés pour définir de la fonction de pondération ;
- tester si un point géométrique appartient au support de la fonction ;
- calculer les valeurs de la fonction et de ses dérivées en un point géométrique donné.

Les classes *weightfunc_ball* et *weightfunc_square* héritent de tous les attributs et méthodes de cette classe et correspondent respectivement aux fonctions à support radial ou rectangulaire. Elles possèdent de ce fait un attribut supplémentaire qui est la valeur du rayon (scalaire ou vectoriel). La classe *weightfunc_square* possède en outre en référence, un objet de type « fonction de pondération à support radial » (type *weightfunc_ball**) à associer à chaque direction du support rectangulaire.

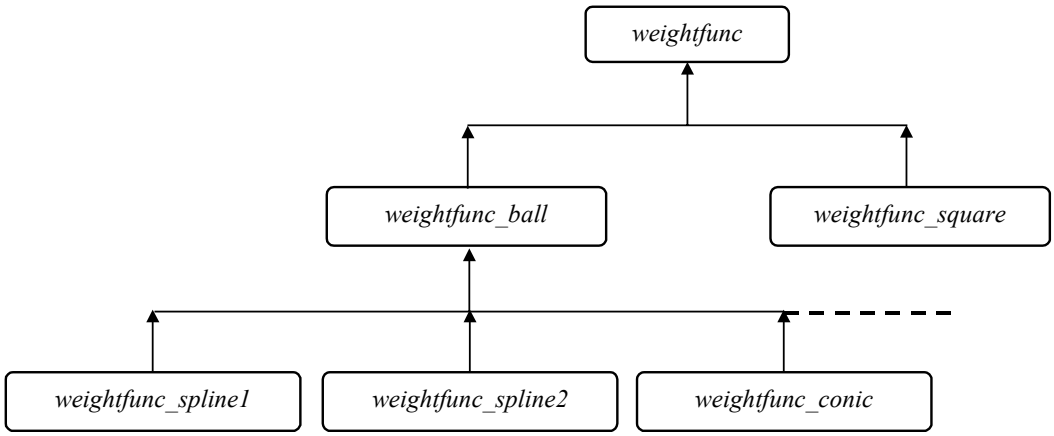



***Fig. 11 : Hiérarchie de la classe « fonction de pondération »***

### B.7.5 Les classes de plus haut niveau

Un certain nombre de classes pouvant être associées au modèle numérique, interviennent au plus haut niveau dans la démarche de modélisation. La classe *problem* par exemple, regroupe les caractéristiques générales d'un problème donné, à savoir le type de problème (statique ou dynamique), le domaine d'étude, les conditions initiales et aux limites, ainsi que la liste des courbes de chargement utilisées à tous les niveaux de l'analyse.

La classe *analysis* permet quant à elle de définir toutes les données de l'analyse, à savoir le problème lui-même et le type d'analyse (géométrie 1D, 2D ou 3D, transformations infinitésimales ou finies), mais également les données numériques, comprenant la liste des points d'intégration utilisés pour réaliser les calculs (liste de points numériques de type *loopPoints,* les objets *loopPoint* étant alloués dynamiquement à ce niveau), le type d'intégration (objet de type *integration**, *cf*. Fig. 12) et le schéma associé (objet de type *numeric**), les données algorithmiques pour la résolution (objet de type *algorithme**) et enfin celles relatives au scénario et à la chronologie de l'étude (objet de type *scenario**). Ces dernières regroupent : l'intervalle de temps sur lequel est réalisée de l'étude, les différents pas de temps envisagés par l'utilisateur, les données de reprise à partir d'un calcul précédent, ainsi que celles permettant éventuellement de gérer le pas de temps adaptatif.

C'est notamment au niveau de la classe *analysis* que s'effectuent :

- la création des degrés de liberté associés aux nœuds de l'analyse ;
- la mise à jour éventuelle des données de pas de temps adaptatifs (communications avec l'objet *scenario*) ;

- les opérations éventuelles de prédiction - correction réalisées au niveau des nœuds de l'analyse et suivant le schéma d'intégration en temps choisi (communications avec l'objet *numeric*) ;

- la boucle générale réalisée sur les points d'intégration de l'analyse (points de Gauss ou nœuds), permettant d'assembler et de résoudre le système global (communications avec l'objet *algorithme*) : que l'analyse utilise la méthode des éléments finis, une méthode meshfree ou une méthode mixte, on procède de la même manière au niveau de la boucle générale ;

- écrire les résultats intermédiaires et finaux sur des streams (en général des fichiers).

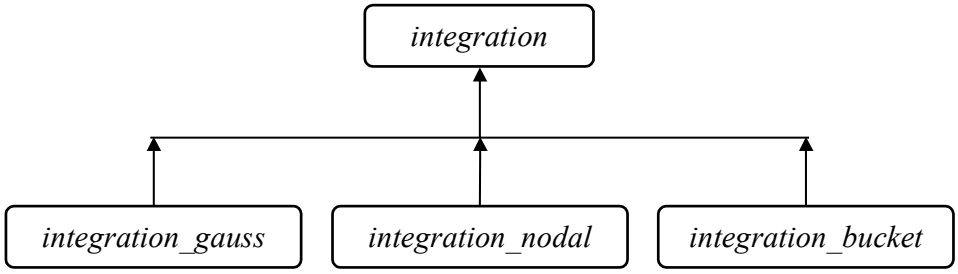



***Fig. 12 : Hiérarchie de la classe « intégration numérique »***

Enfin, la classe *interface* est la classe « interface utilisateur » : ses méthodes permettent de tenir compte du type de système d'exploitation (Windows, Linux, etc.) et de la façon dont l'utilisateur entrera ses données. Dans *MoveFree*, il est en effet possible d'entrer les données sous forme graphique (interface Windows), sous forme de ligne de commandes à l'écran ou encore sous forme de script.

Remarque :

Les classes de haut niveau permettent par ailleurs d'allouer une fois pour toute, des objets globaux qui seront ensuite référencés par les autres objets intervenant dans le modèle. Par exemple, l'allocation de la plupart des listes (objets de type « collection d'objets ») tels que les courbes, matériaux, points géométriques, nœuds, etc. (*cf*. Fig. 13), est réalisée une seule fois au niveau de la classe *interface*.

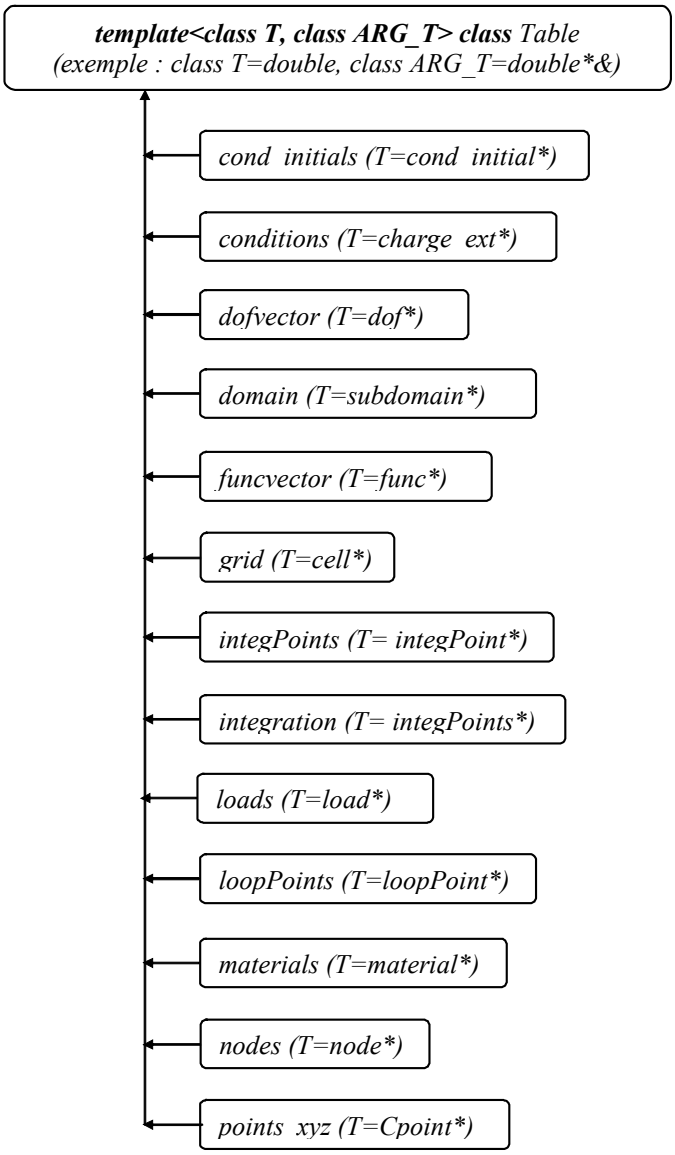



***Fig. 13 : Hiérarchie de la classe générique Table ou « collection d'objets »***