Table des matières

Résumé

Le stage s'est fait au GIREF, (Groupe interdisciplinaire de recherche en éléments finis). Forintek, Institut de recherche sur les produits du bois du Canada, voulait avoir des données numériques quant à l'ordre de grandeur de la déformation d'un quart de panneau d'armoire lors du séchage. Pour ce faire, il fallait préalablement comprendre la méthode des éléments finis, apprendre à utiliser MEF++, faire un maillage pour le quart de panneau d'armoire, faire fonctionner le programme servant aux simulations et faire les simulations. Dans ce rapport, il sera question de tous ces aspects. Finalement, nous discuterons des résultats obtenus.

1 Introduction

Dans le cadre de mon stage à l'été 2006, j'ai travaillé au GIREF (Groupe interdisciplinaire de recherche en éléments finis). J'étais sous la supervision de monsieur André Fortin. Tout au long de mon stage, j'ai aussi eu l'aide de Jean Deteix (professionnel de recherche), Zanin Kazanov (étudiant au doctorat) et Christian Tibirna (professionnel de recherche du GIREF). Je voudrais remercier toutes ces personnes pour l'aide qu'elles m'ont apportée. J'ai travaillé sur un projet de Forintek, Institut de recherche sur les produits du bois du Canada.

L'industrie canadienne du bois est en constante évolution. Pour rester compétitive face aux industries d'ailleurs et être à la fine pointe de la technologie, elle doit trouver de nouveaux procédés et améliorer ceux existants. Depuis longtemps, l'industrie du bois fait face à un problème : la déformation du bois lors du séchage. Cette déformation peut être minime, mais dans certains cas, elle peut être visible à l'oeil nu.

Durant le stage, la déformation du bois lors du séchage a été étudiée, mais plus précisément dans la fabrication de panneaux de portes d'armoires. Forintek voulait avoir des données numériques concernant l'ordre de grandeur de la déformation d'un quart de panneau d'une porte d'armoire coupé au 90 $^\circ$.

Les objectifs du stage étaient :
- Comprendre la méthode des éléments finis
- Apprendre à utiliser MEF++
- Faire un maillage pour le quart de panneau d'une porte d'armoire
- Faire fonctionner le programme servant à simuler les déformations d'un quart de panneau d'armoire
- Faire des simulations
- Arriver à des résultats concluants

Organisme hôte

L'organisme hôte est le Groupe interdisciplinaire de recherche en éléments finis (GIREF). Les membres du GIREF travaillent à résoudre des problèmes provenant des industries à l'aide de simulations numériques qui utilisent la méthode des éléments finis.

Description du stage par l'organisme hôte

- Programmation en C++ dans un code d'éléments finis
- Élaboration et simulation de quelques cas tests de source industrielle
- Analyse des résultats et présentation aux industriels concernés

2 Méthode des éléments finis

Au début du stage, j'ai lu les notes de cours *Les éléments finis : de la théorie à la pratique* pour que je puisse me familiariser avec la méthode des éléments finis. Cette méthode est utilisée au GIREF. Elle est une technique d'approximation des formulations variationnelles des équations aux dérivées partielles.

2.1 Un exemple théorique

Pour pouvoir résoudre un problème avec la méthode des éléments finis, il faut commencer par trouver sa formulation variationnelle. La formulation variationnelle est aussi appelée forme faible de l'équation différentielle de départ qui elle est la forme forte. On l'appelle forme faible, car elle contient des dérivées d'ordre inférieur à la forme forte. La formulation variationnelle n'a pas besoin d'être aussi régulière que la forme forte. Elle se trouve en plusieurs étapes. Pour bien les illustrer, un exemple typique de ceux présentés dans les notes de cours est fait ici avec une équation aux dérivées partielles d'ordre 2.

Soient ν_1 et ν_2 deux constantes strictement positives
Soit f une fonction dans $L^2(\Omega)$ et Ω est un ouvert. Ici $\Omega =]0,1[$

$$\begin{cases} \nu_1 u - \nu_2 \dfrac{d^2 u}{dx^2} = f \quad \text{dans }]0,1[\\ \frac{du}{dx}(0) = \frac{du}{dx}(1) = 0. \end{cases} \tag{1}$$

Les conditions essentielles et naturelles sont sur la frontière du domaine. Ici, il n'y a aucune condition essentielle (de Diriclet) imposée, car u est une dérivé sur la frontière. Les conditions aux limites sont naturelles, donc de Neumann. L'espace fonctionnel dans lequel le problème se trouve est $V = H^1(]0,1[)$ puisque qu'il n'y a aucune condition essentielle imposée. $H^1(\Omega)$ est un espace de Sobolev, où Ω est un ouvert borné. $H^1(\Omega)$ se définit par :

$$H^1(\Omega) = \{u \in L^2(\Omega) \,|\, \frac{\partial u}{\partial x_i} \in L^2(\Omega), 1 \leq i \leq 3\}.$$

On multiplie l'équation de départ par une fonction test à support compact $\omega \in V$.

$$\nu_1 \omega u - \nu_2 \omega \frac{d^2 u}{dx^2} = f\omega \qquad \forall \omega \in V.$$

Puisque l'équation (1) se trouve dans H^1, on intègre par parties les termes d'ordre 2 pour les faire disparaître. On a de l'équation précédente

$$\int_0^1 \nu_1 \omega u dx - \int_0^1 \nu_2 \omega \frac{d^2 u}{dx^2} dx = \int_0^1 f\omega dx \qquad \forall \omega \in V.$$

Par hypothèse, $\frac{du}{dx}(0) = 0$ et $\frac{du}{dx}(1) = 0$. Il est alors possible d'intégrer et de simplifier pour obtenir :

$$\int_0^1 \nu_1 \omega(x) u(x) + \nu_2 u'(x)\omega'(x)dx = \int_0^1 f(x)\omega(x)dx \quad \forall \omega \in V. \tag{2}$$

L'équation (2) est une formulation variationnelle. Il n'y a pas de relèvement à faire sur les conditions essentielles, car il n'y en a aucune d'imposée à u.

La formulation variationnelle doit maintenant vérifier les conditions du théorème de Lax-Milgram pour avoir existence et unicité de la solution. Pour ce faire, il faut préalablement savoir ce qu'est une forme bilinéaire coercive.

Définition :
Une forme bilinéaire a est dite *coercive* s'il existe une constante strictement positive α telle que :

$$a(\omega, \omega) \geq \alpha \parallel \omega \parallel_V^2 \qquad \forall \omega \in V.$$

Théorème de Lax-Milgram

Soit V un espace de Hilbert, l une forme linéaire et a une forme bilinéaire continue sur $V \times V$ respectivement ($l \in V'$ où V' est le dual de V). Si de plus a est coercive, alors il existe une unique solution u du problème variationnel :

• trouver une fonction $u \in V$ telle que :

$$a(u, \omega) = l(\omega) \qquad \forall \omega \in V.$$

Dans le cas présent, les conditions du théorème de Lax-Milgram sont vérifiées. Alors,

$$\begin{aligned} a(u, \omega) &= \int_0^1 \nu_1 \omega(x) u(x) + \nu_2 u'(x)\omega'(x)dx \\ l(\omega) &= \int_0^1 f(x)\omega(x)dx. \end{aligned}$$

Le problème est de dimension infinie. Il reste à le discrétiser pour pouvoir le résoudre avec des méthodes numériques. On utilise les fonctions de Ritz. Le but est d'avoir N fonctions $\phi_j \in V$, $j = 1, 2, 3...N$ qui vérifient les conditions essentielles homogènes. On a donc :

$$u(x) \simeq u^N(x) = \sum_{j=1}^{N} u_j \phi_j(x).$$

On trouve les N fonctions ϕ qui forment une base pour générer toutes les autres fonctions. Ainsi, le problème devient de dimension finie N. La base trouvée est un sous-espace de V et il est noté V^N. Cela permet de résoudre le problème variationnel suivant :

• trouver $u^N \in V$ telle que :

$$a(u^N, \omega^N) = l(\omega^N) \qquad \forall \omega^N \in V^N$$

$$
\begin{aligned}
a\left(\sum_{j=1}^{N} u_j \phi_j(x), \omega^N\right) &= l(\omega^N) \qquad \forall \omega^N \in V^N \\
\sum_{j=1}^{N} u_j a(\phi_j, \omega^N) &= l(\omega^N) \qquad \forall \omega^N \in V^N.
\end{aligned}
$$

On pose $\omega^N = \widehat{\phi}_i$.

On a donc

$$
\sum_{j=1}^{N} u_j a(\phi_j, \widehat{\phi}_i) = l(\widehat{\phi}_i) \qquad 1 \leq i \leq N.
$$

Ceci est un système linéaire de la forme $A\overrightarrow{U} = \overrightarrow{F}$ de dimension N.
Un choix possible pour $\widehat{\phi}_i$ est de poser $\widehat{\phi}_i = \phi_i$.

Dans le cas du problème présenté ici, une possibilité de fonctions ϕ_i linéairement indépendantes est $\phi_i = x^i$, et $\phi_0 = 1$. On doit imposer $\phi_0 = 1$. Sinon, la fonction serait forcément nulle au point 0 et il y aurait une condition essentielle à cette extrémité.

On résout le système pour $\overrightarrow{U}$. Les fonctions U forment une famille pour approximer la formulation variationnelle.

$$
\begin{aligned}
A_{i,j} &= \int_0^1 \nu_1 \phi_i \phi_j dx + \int_0^1 \nu_2 \phi_i'(x) \phi_j'(x) dx \\
&= \int_0^1 \nu_1 x^{i+j} dx + \int_0^1 ij\nu_2 x^{i+j-2} dx \\
\overrightarrow{F} &= \left(\int_0^1 f(x) 1 dx, \int_0^1 f(x) x^1 dx, \ldots \int_0^1 f(x) x^{N-1} dx\right) \\
\overrightarrow{U} &= (u_0, u_1, \ldots, u_{N-1}).
\end{aligned}
$$

2.2 Résolution numérique

2.2.1 Le maillage

Il est nécessaire d'avoir un maillage pour résoudre un problème à l'aide des éléments finis. On peut voir un maillage comme un grillage ou un treillis. Le maillage est composé de noeuds. Les noeuds géométriques sont reliés pour former des arêtes. Ces dernières forment le contour des éléments. En 2D, les éléments sont des triangles ou des quadrilatères. En 3D, les éléments sont des hexaèdres ou des tétraèdres.

À chaque noeud de calcul, il y a un degré de liberté d'associé. Les degrés de liberté sont les inconnues du problème. Certains de ces degrés de liberté sont connus. Ce sont ceux donnés par les conditions aux limites essentielles du problème. À chaque degré de liberté, une fonction de

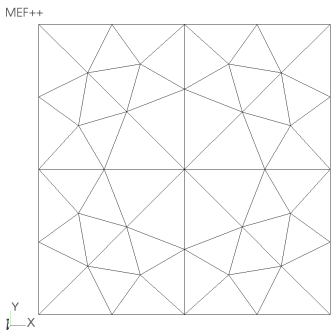


FIG. 1 – Exemple de maillage 2D avec triangles

Ritz est attribuée. Lors de la résolution numérique, des approximations de $u(x)$ sont calculées sur chaque noeud. Ce sont des noeuds d'interpolation. L'approximation de la solution est donc constituée de plusieurs fonctions dans chaque élément. Pour résoudre, on intègre sur chaque élément. Comme il peut y en avoir plusieurs, les calculs sont toujours faits sur un élément de référence.

On a besoin de fonctions d'interpolation pour trouver une solution. Il y a plusieurs types d'interpolation, comme des approximations linéaires, hiérarchiques ou quadratiques, tout dépendemment du nombre de noeuds de calculs utilisés. Pour une approximation linéaire, deux noeuds sont nécessaires tandis que pour l'approximation quadratique, il en faut trois. L'approximation hiérarchique est un mélange des deux. Une approximation quadratique est plus précise qu'une approximation linéaire.

2.2.2 Logiciel MEF++

Un didacticiel de MEF++ est disponible au GIREF. Il a été très utile pour l'apprentissage de MEF++. Ce dernier est un programme écrit en C++ et qui a été développé au GIREF pour résoudre des problèmes d'éléments finis.

Pour utiliser MEF++, il est nécessaire de fournir cinq fichiers décrivant la pièce à modéliser au programme. Il y a le fichier *préfixe*.mail qui contient le maillage et le fichier *préfixe*.geom qui contient la géométrie. Le fichier *préfixe*.enti contient les différentes entités nécessaires à la résolution du problème. Les entités sont des portions bien spécifiques de la géométrie. Par exemple, définir plus d'une entité sur une géométrie permet d'attribuer différents matériaux à certains endroits de la pièce. Pour obtenir ces trois fichiers, on les exporte à l'aide du logiciel iMEF++. Les fichiers *préfixe*.CL et *préfixe*.champs sont écrits par l'utilisateur du programme. Le fichier *préfixe*.CL contient les conditions aux limites du problème, tandis que le fichier *préfixe*.champs contient toutes les données nécessaires à la résolution du problème.

3 Le quart du panneau d'armoire

3.1 Définition du problème

Le quart du panneau d'une porte d'armoire est composé de trois morceaux. Il y en a deux pour le cadre et un pour le panneau central. Une couche de colle de 0,1 mm est mise pour tenir les deux morceaux du cadre. Le panneau du centre n'est pas collé. Lors du séchage, il y a transport d'humidité entre la pièce de bois et la colle, de même qu'entre la pièce de bois et le milieu ambiant. En séchant, le quart de panneau subit un stress de déformation.

Dans la modélisation d'un quart de panneau d'une porte d'armoire, il y a deux problèmes à résoudre, soit un hydrique pour l'humidité et un autre mécanique pour la déformation subie. Le problème mécanique est dépendant du problème hydrique.

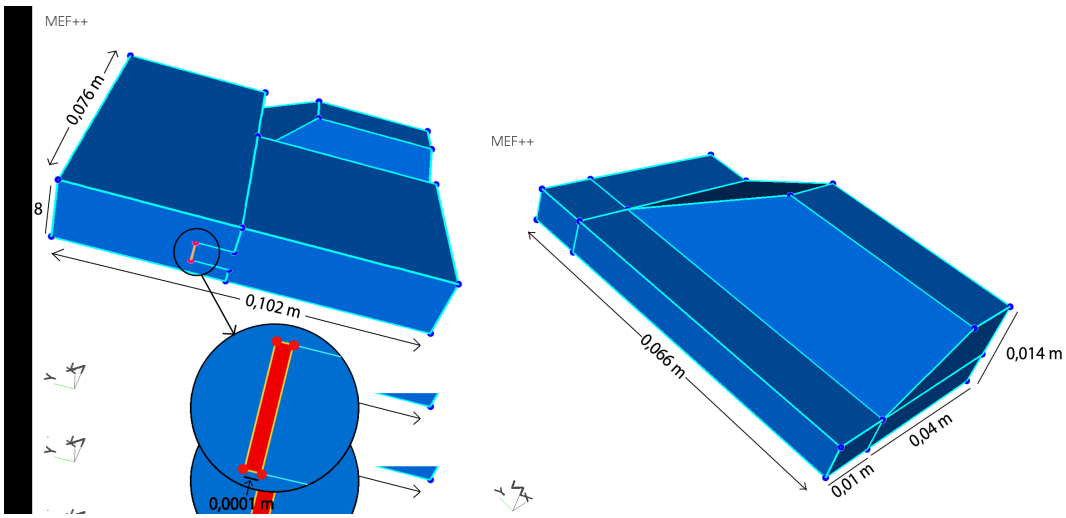
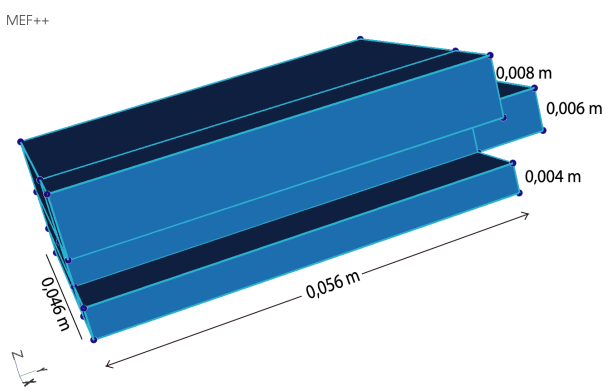


FIG. 2 – Géométrie d'un quart de panneau d'armoire et emplacement de la colle (figure à gauche)

FIG. 3 – Pièce amovible centrale (figure à droite)

Le matériau utilisé a été le bois d'érable. C'est un matériau orthotrope vivant. Cela veut dire que ses propriétés changent, dépendemment de la direction dans laquelle les axes de référence sont orientés. Le bois a trois directions privilégiées. Il y a la direction radiale, longitudinale et tangentielle.



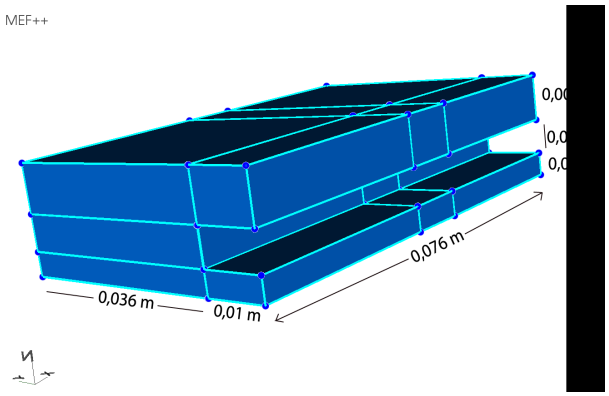


FIG. 4 – Pièce 1 du cadre (figure à gauche)

FIG. 5 – Pièce 2 du cadre (figure à droite)

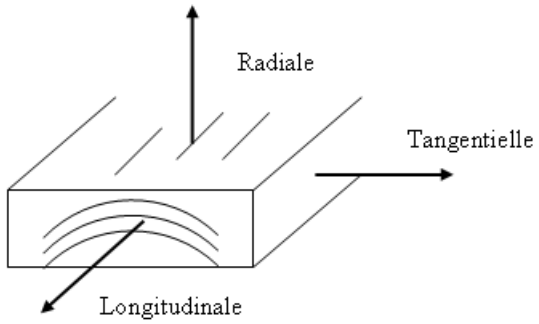


FIG. 6 – Graphique montrant les trois directions privilégiées d'un morceau de bois

3.2 Modèle du problème

3.2.1 Le modèle hydrique

Pour résoudre le problème hydrique, c'est l'équation de transport qui est utilisée pour étudier le transport de l'humidité.

$$\frac{d\mu}{dt} + \overrightarrow{a} \cdot \bigtriangledown \mu = f \tag{3}$$

L'équation 3 dit que la quantité μ est transportée à une vitesse $\overrightarrow{a}$. Si div $\overrightarrow{a}$ est zéro alors, avec des manipulations, il est possible de l'adapter au problème hydrique étudié pour le quart du panneau d'armoire. On obtient :

$$\frac{db}{100\%}\frac{\partial M}{\partial t} - \nabla \cdot \left(\frac{Ddb}{100\%} \nabla M \right) = f \tag{4}$$

où
$db =$ densité basale $\left(\frac{\text{kg}}{\text{m}^3}\right)$

$M =$ humidité en %
$D =$ diffusivité effective $\left(\frac{\mathrm{m}^2}{\mathrm{s}}\right)$
$\frac{Ddb}{100\%} =$ conductivité hydrique
$f =$ le terme source (dans le cas présent, $f = 0$).

L'équation 4 est celle utilisée dans le programme.

Au bord de la pièce, il y a une condition limite hydrique.

$$\frac{dbD}{100\%}\nabla M \cdot \overrightarrow{n} = h_m \left(M - M_\infty\right)$$

où
$M_\infty =$ taux d'humidité du milieu ambiant (%)
$h_M =$ coefficient de transfert de masse dans le cas de l'humidité $\left(\frac{\mathrm{kg}}{\mathrm{ms}\%}\right)$.

Cette condition aux limites veut dire qu'il y a échange d'humidité entre le milieu ambiant et la pièce de bois. Cette condition en est donc une de Neumann. Sur les autres faces, par défaut, l'échange est 0.

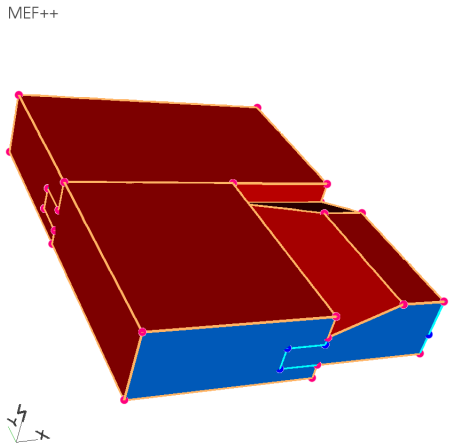


FIG. 7 – Les faces en contact avec le milieu ambiant (en rouge)

3.2.2 Le modèle mécanique

Pour résoudre le problème mécanique, on résout l'équation d'élasticité. On a :

$$\begin{aligned} \sigma &= E : \varepsilon\left(U\right) - \beta_M\left(\Delta M\right) \\ \mathrm{div}\sigma &= F \\ -\nabla \cdot \left(E : \varepsilon\left(U\right) - \beta_M\left(\Delta M\right)\right) &= F \end{aligned}$$

où
E = tenseur d'élasticité du bois (GPa)
β_M = coefficient de dilatation (%)
$U = (U_x, U_y, U_z)$ le vecteur de déplacement (m).

C'est cette dernière équation qu'on résout dans le programme.

La formulation variationnelle à résoudre est :

$$\int \nabla \cdot (E : \gamma(U) - \beta_M(\Delta M)) : \gamma(\omega)\, dx = \int \vec{F}\,\vec{\omega}\, dx.$$

Les conditions limites imposées à la mécanique sont un axe de symétrie en x, un autre en y et un point fixe en (0,076, -0,102, 0,009).

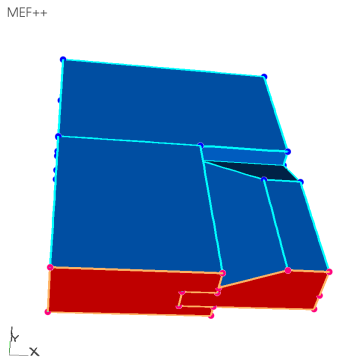


FIG. 8 – L'axe de symétrie en x (en rouge, figure à gauche)

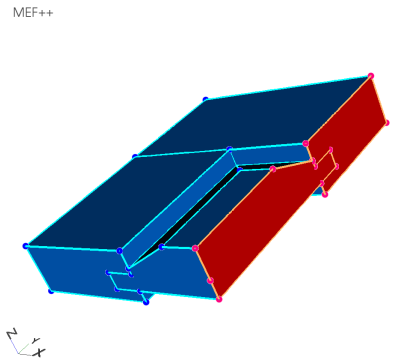


FIG. 9 – L'axe de symétrie en y (en rouge, figure au centre)

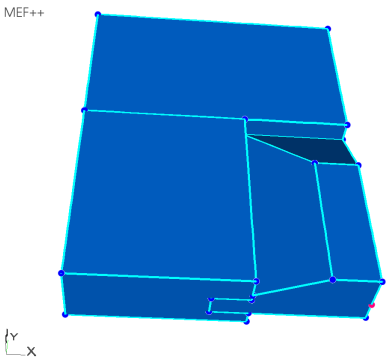


FIG. 10 – Le point fixe (en rouge, figure à droite)

3.3 Création du maillage

Pour pouvoir utiliser le programme qui simule la déformation causée lors du séchage, on fournit un maillage. La création du maillage a posé beaucoup plus de difficulté que prévu. Ce qui devait être deux semaines de travail, s'est avéré deux mois.

Pour qu'un maillage 2D soit correct, les noeuds et les arêtes de deux entités adjacentes doivent tous coïncider. En 3D, les noeuds, les arêtes et les surfaces de toutes les entités adjacentes doivent coïncider. La géométrie 3D à construire était complexe, puisqu'elle était composée de trois pièces emboîtées. La construction de chaque pièce séparément n'était pas problématique. La difficulté arrivait lors de l'emboîtement des trois pièces, car il fallait garder en tête que les éléments adjacents des différentes entités devaient tous coïncider.

Lorsqu'un maillage est fait, il faut l'importer dans iMEF++ pour la création des fichiers *préfixe*.mail, *préfixe*.geom et *préfixe*.enti. Une difficulté rencontrée est que iMEF++ ne détecte pas toutes les erreurs de maillage, comme certains dédoublements d'arêtes ainsi que des éléments qui ne coïncident pas pour deux entités adjacentes. Puisqu'il n'était pas su que iMEF++ ne détectait pas ces erreurs, de mauvais maillages ont longtemps été utilisés et ce, sans le savoir. Par conséquent, l'algorithme du programme divergeait et l'erreur n'était pas cherchée au bon endroit. Le programme a été vérifié plutôt que le maillage.

Un premier maillage grossier d'un quart de panneau de porte d'armoire a été fait dans Gmsh. Une particularité de Gmsh est que pour donner la longueur des arêtes des éléments, il faut le faire lors de la déclaration des points. On donne une longueur caractéristique de l'arête. À partir de cette information, Gmsh maille la pièce. Pour s'assurer que les éléments de chaque pièce adjacente coïncideraient, les trois pièces ont été créées dans un même fichier. De plus, pour mailler correctement, il fallait découper chacune des pièces en plusieurs petits solides de manière à ce qu'il n'y ait jamais de noeuds géométriques arrivant au milieu d'une arête.

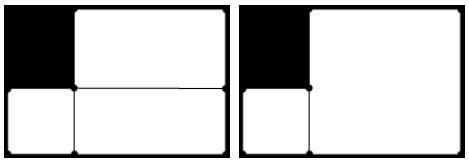


FIG. 11 – Bon maillage à gauche, mauvais maillage à droite

Malgré cela, le maillage était incorrect. Le problème venait de la manière que les différentes parties de la pièce étaient faites. Les noeuds ne coïncidaient pas s'il y avait un mélange d'extrusion et de coordonnées de sommets données. Il a donc fallu créer tous les solides de la même manière soit par extrusion. De cette manière, un maillage grossier a été fait.

Par la suite, il fallait raffiner le maillage dans Gmsh en changeant la longueur caractéristique des éléments. Gmsh arrivait à mailler en 1D et en 2D, mais plus en 3D. Gmsh est un logiciel libre. En faisant des recherches dans Internet, la littérature disait qu'il y avait des améliorations à apporter au logiciel pour son utilisation en 3D et qu'il était possible que pour certaines longueurs caractéristiques, Gmsh ne fasse pas le maillage 3D.

Pour contrer le problème du maillage trop grossier, ce dernier a été importé quand même dans iMEF++ et à partir de iMEF++, un maillage plus fin a été généré. Chaque arête a été divisée par deux. Un maillage plus fin a été obtenu, mais ayant huit fois plus d'éléments. Il avait approximativement 20 000 éléments. En faisant cela, il y avait des éléments extrêmement petits dans la couche de colle comparativement aux autres éléments.

Jean Deteix a donc suggéré de faire un maillage plus fin avec un autre logiciel. En même temps, des expérimentations avec Patran, un logiciel commercial de maillage ont été faites, puisqu'à partir d'une géométrie, Patran permet qu'on lui donne le nombre d'éléments sur chaque arête. Ainsi,

un maillage satisfaisant a été obtenu. Par contre, lors de l'importation dans iMEF++, ce dernier ne détectait pas les différents solides créés dans Patran. Il était impossible de définir les différentes entités composant le quart du panneau d'armoire. Il a donc fallu abandonner ce maillage.

Le maillage utilisé a donc été celui que Jean Deteix a fait.

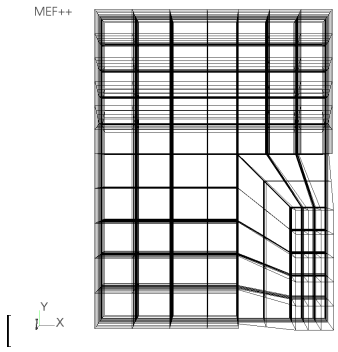


FIG. 12 – Maillage utilisé en hexaèdres

3.4 Modification du code

Jean Deteix a envoyé le code que Mathieu Arbour a utilisé lors de son stage. Avec l'aide de Jean Deteix, le code de Mathieu Arbour a été adapté pour être utilisable pourle problème du quart de panneau d'amoire. Le tableau suivant contient les données utilisées pour les fichiers .champs, mec.CL et hyd.CL. Dans le tableau, L=longitudinal, R=radial et T=tangentiel en faisant référence aux directions privilégiées du bois. Les fichiers .champs, mec.CL et hyd.CL sont en annexe.

Propriété	Définition	Érable	Colle
$db(\frac{kg}{m^3})$	Densité basale	597	1500
$D_L(\frac{m^2}{s})$	Diffusité effective	1,3e-08	1,0e-15
$D_R(\frac{m^2}{s})$	Diffusité effective	1,8e-11	1,0e-15
$D_T(\frac{m^2}{s})$	Diffusité effective	1,8e-11	1,0e-15
$h_M\left(\frac{kg}{m^2 s\%}\right)$	Coefficient de transfert de masse	3,2e-04	3,2e-04
$\beta_{eff}(\%)$	Coefficient de dilatation		1,9e-02
$\beta_L(\%)$	Coefficient de dilatation	1,842e-04	
$\beta_R(\%)$	Coefficient de dilatation	1,9053e-03	
$\beta_T(\%)$	Coefficient de dilatation	2,7579e-03	
$\alpha_L(\%)$	Coefficient de dilatation	1,4715e-04	
$\alpha_R(\%)$	Coefficient de dilatation	2,1382e-03	
$\alpha_T(\%)$	Coefficient de dilatation	3,3495e-03	
$E_L(Gpa)$	Module de Young	13,810	10
$E_R(Gpa)$	Module de Young	1,311	10
$E_T(Gpa)$	Module de Young	0,678	10
$G_{LR}(Gpa)$	Cisaillement	1,013	
$G_{RT}(Gpa)$	Cisaillement	0,255	
$G_{LT}(Gpa)$	Cisaillement	0,753	
ν_{TL}	Coefficient de Poisson	0,025	0,35
ν_{TR}	Coefficient de Poisson	0,43	0,35
ν_{LR}	Coefficient de Poisson	0,46	0,35
ν_{LT}	Coefficient de Poisson	0,50	0,35
M (%)	Humidite	6.3	6.3
M_∞ (%)	Humidité ambiante	5	5

3.5 Résolution du problème

3.5.1 Les problèmes rencontrés dû au mauvais maillage

Il est très important qu'un maillage soit bon avant de l'utiliser, sinon, toutes les données recueillies sont fausses, ou bien l'algorithme du programme divergera, comme cela est arrivé dans les premières simulations. Avant de savoir que le maillage était problématique, toutes sortes de choses ont été essayées pour contrer la divergence.

Les tolérances pour les solveurs linéaires et non linéaires ont été modifiées. Elle a été mise moins sévère pour que la convergence ait lieu plus tôt. Cela n'a pas amélioré les résultats.

La résolution en mode direct pour les maillages avec peu d'éléments a été faite lorsque c'était possible au lieu de la méthode de d'en gradient conjugué. La méthode directe utilise une factorisation LU pour résoudre un système linéaire. On se disait que la méthode de gradient conjugué avait peut-être certaines limites. Avec résolution en mode direct, il y avait des pivots nuls dans la matrice du système. Par conséquent, il n'y avait tout simplement pas de solution.

Plusieurs préconditionneurs ont été essayés pour améliorer le conditionnement de la matrice du système à résoudre. Cette dernière est multipliée par la matrice du préconditionneur et cela modifient ses entrées. Chaque préconditionneur utilise une matrice différente. Le changement de préconditionneur n'a pas amélioré les résultats.

Le type d'interpolation a été changé. Le programme est passée de l'interpolation quadratique à hiérarchique et ensuite à linéaire. Il se peut que l'algorithme d'un programme diverge avec une interpolation quadratique et converge avec une interpolation linéaire. En interpolation quadratique, il y a plus de degrés de liberté, donc plus d'inconnues. En changeant le type d'interpolation, la divergence se produisait, mais à un pas de temps différent.

L'orientation des pièces de bois a été modifiée. Il fallait savoir si la divergence était due au matériau. Cela n'était pas le cas.

L'ordre de grandeur de certaines constantes a été changé. Par exemple, pour la colle, la diffusivité effective est de l'ordre de 10e-15. Elle a été mise à 10e-5 pour vérifier si la divergence provenait de la limite numérique de l'ordinateur. Malgré ce changement, il y avait quand même divergence.

Dans un premier temps, la diffusivité effective radiale et tangentielle de l'érable étaient une fonction en terme de l'humidité qui devenait négative pour certaines valeurs. Il en résultait des calculs erronés puisqu'à un certain moment, le taux d'humidité devenait négatif. Cette fonction a été remplacée par des constantes positives. Il était raisonnable de la changer puisque la fonction s'appliquait pour le problème de Mathieu Arbour, mais pas pour celui du quart de panneau d'armoire. Les constantes utilisées sont celles de la thèse de monsieur Pierre Blanchet.

Par la suite, seul le problème hydrique à été testé. Il fallait savoir où se trouvait plus précisément le problème, dans l'hydrique ou bien dans la mécanique. En fait, le problème hydrique ne fonctionnait pas.

Pour tous ces tests, le maillage avec des hexaèdres a été utilisé. Par la suite, le maillage avec des tétraèdres fait avec Gmsh a été utilisé. En changeant de maillage, l'algorithme du programme divergeait encore. Finalement, le programme a été lancé avec les deux maillages en mode développement au lieu d'en mode optimisé. Le mode développement fait plus de tests de vérification que le mode optimisation. En mode développement, il y a eu divergence avec chacun des maillages. Cela voulait dire que les deux maillages étaient problématiques.

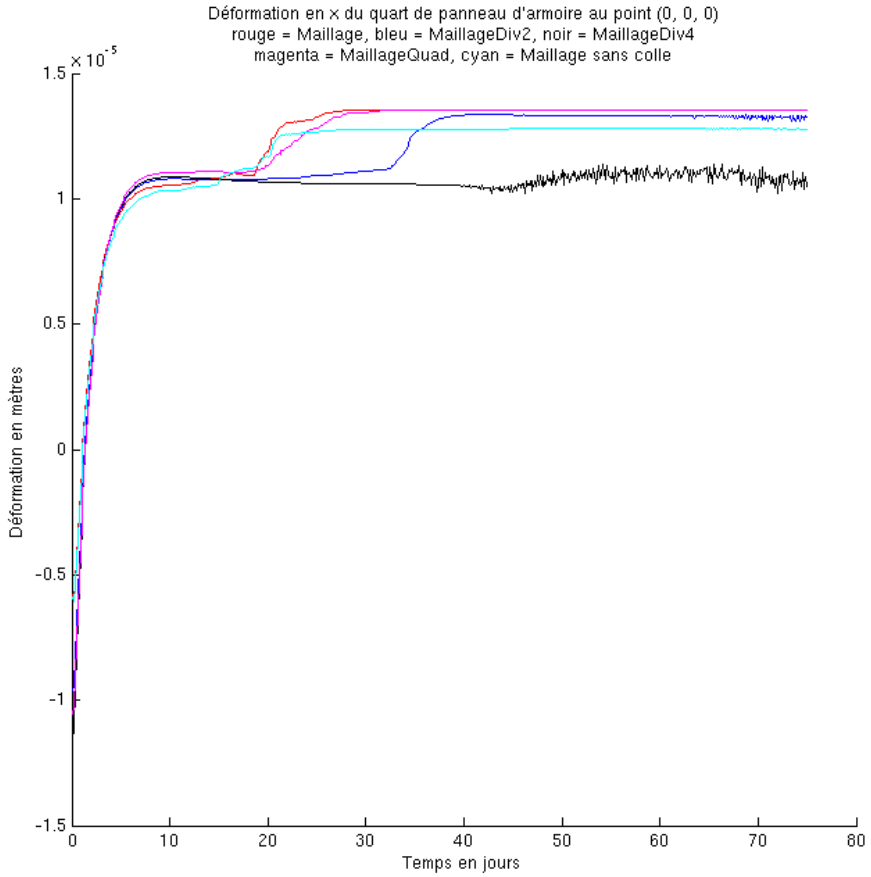
En regardant de plus près le maillage avec des hexaèdres dans iMEF++, il a été trouvé qu'il y avait un dédoublement d'arêtes à trois endroits. Cela impliquait qu'il y avait des conditions aux limites à l'intérieur du quart de panneau d'armoire et cela n'a pas de sens. En ce qui concerne le maillage avec des tétraèdres, il a été découvert que les arêtes et les noeuds du prisme rectangulaire de la pièce amovible ne coïncidaient pas avec ceux de la partie en pente de cette même pièce. Il était donc normal que l'algorithme du programme diverge.

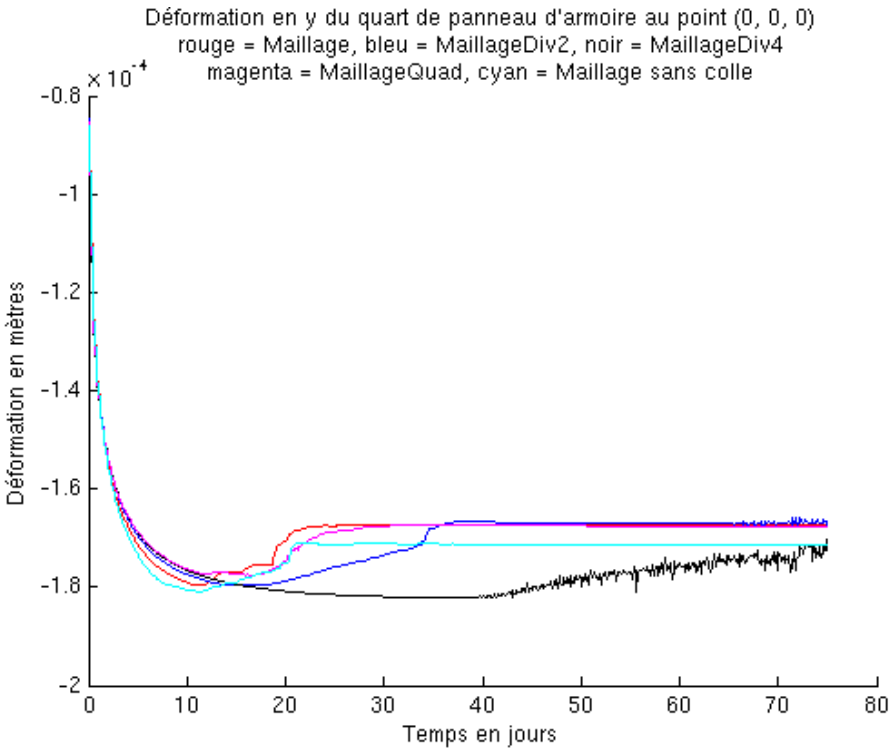
3.5.2 L'adimensionnalisation

Lors de la résolution en méthode directe, il est souvent arrivé d'avoir un pivot nul dans la matrice du système à résoudre et donc d'être dans l'impossibilité de résoudre numériquement. En fait, le pivot n'était pas nul, mais très petit. Dû à la précision de l'ordinateur, ce dernier considérait le pivot comme nul. Pour contrer cela, le problème hydrique et mécanique ont été adimensionnalisés. Les calculs et le fichiers .champs adimensionnalisés sont en annexe. En adimendisionnalisant, toutes les valeurs ont un ordre de grandeur plus près les unes des autres. Ainsi, on ne fait plus de calculs en utilisant de très petites et de très grandes valeurs.

3.6 Les résultats

La géométrie utilisée n'est pas problématique. Avec différents maillages et différents types d'interpolation, les déformations sont extrêmement petites et non visibles à l'oeil nu. Par contre, on se rend compte que l'ordre de grandeur des déformations n'est pas la même, dépendemment de l'orientation des fibres de bois. Les déformations de six points ont été étudiées. Par exemple, le point (0, 0, 0) et (0, -0,1, 0) n'ont pas le même type de déformation.



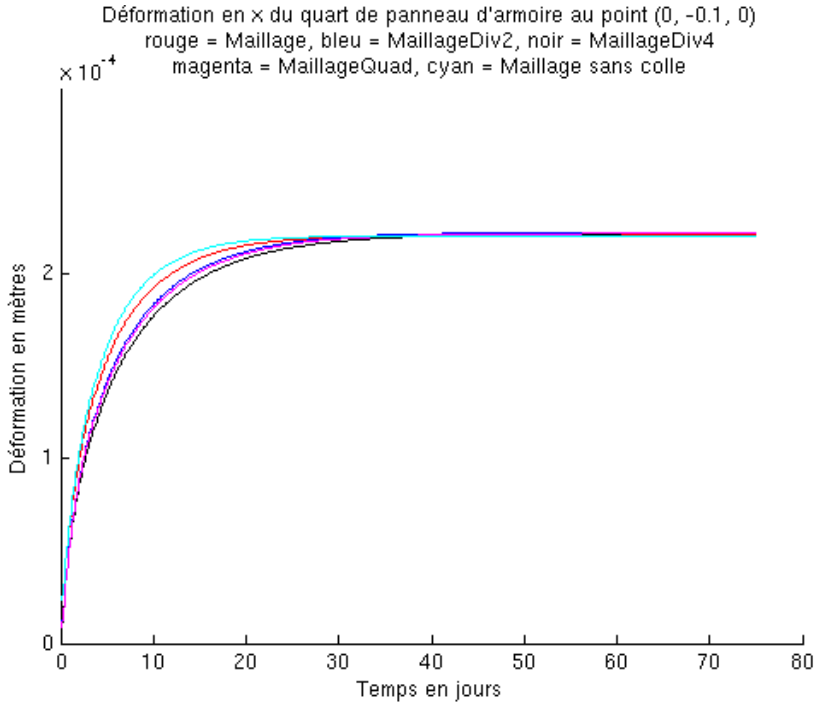


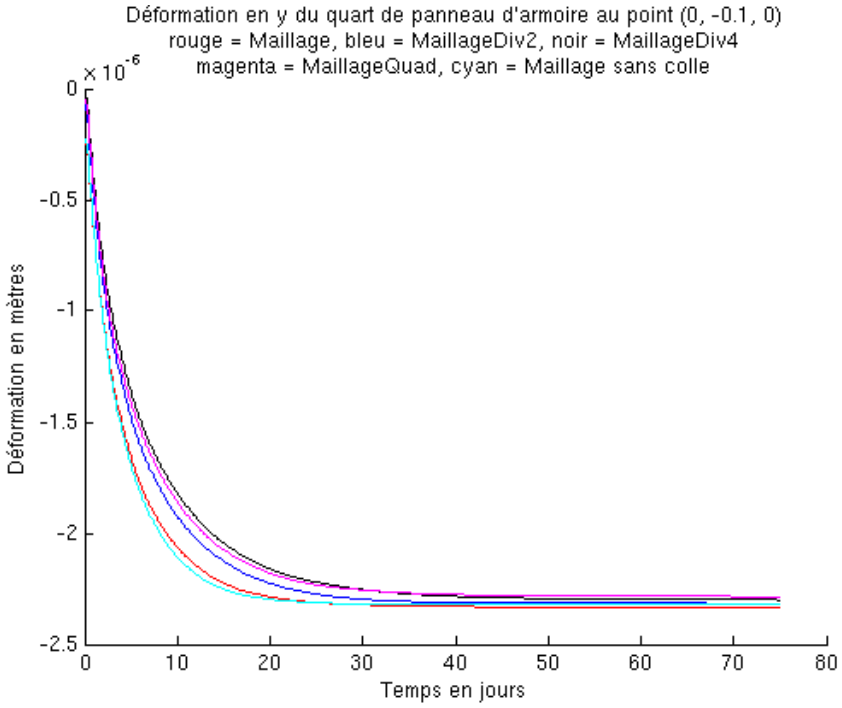
On remarque que pour le point (0, 0, 0), la déformation en x est de l'ordre de 10^{-5}, tandis que celle en y est de l'ordre de 10^{-4}. Ceci est tout à fait normal, dû à l'orientation des fibres de bois à cet endroit. Il est possible de dire qu'il y a une plus grande déformation dans le sens tangentiel que longitudinal. On pourrait penser que le saut dans la courbe est problématique. Puisque l'ordre de grandeur des déformations est très petit, il n'y a pas lieu de s'inquiéter. La courbe bleue qui représente le maillage en hexaèdre divisé par deux et le maillage en noir qui représente le maillage divisé en quatre ont des oscillations. Cela est causé par la précision numérique finie de l'ordinateur. Dans les premières simulations, les oscillations étaient plus grandes. En augmentant la précision des solveurs linéaires, elle ont diminué.

On remarque que pour le point (0, -0,1, 0), la déformation en x est de l'ordre de 10^{-4}, tandis que celle en y est de l'ordre de 10^{-6}. Ceci coïncide avec les résultats du point (0, 0, 0). Les fibres de bois sont orientées perpendiculairement au point (0, -0,1, 0) par rapport au point (0, 0, 0). La courbe à ce dernier point n'a pas de saut. Plus le point étudié est loin de la couche de colle, plus le saut diminue.

On peut conclure qu'il y a de plus grandes déformations mécaniques dans le sens tangentiel de la pièce de bois que longitudinal, mais il faut toutefois garder une certaine réserve. D'autres simulations devraient être faites pour valider.

Les résultats obtenus ne sont pas concluants. Pour en avoir des représentatifs, en gardant la diffusivité effective radiale et tangentielle de l'érable comme des constantes, il aurait fallu refaire tourner le programme avec un maillage d'une géométrie plus grande, puisque celle utilisée cet été ressemble plus à un coin d'armoire qu'à un quart de panneau d'armoire. Dû à la contrainte





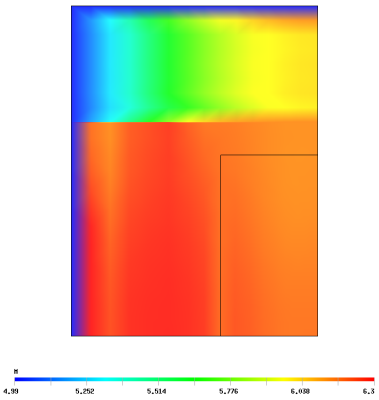
de temps, (la du fin du stage), il n'y avait pas assez de temps pour obtenir un maillage d'une géométrie plus grosse pour ensuite le faire tourner. Ainsi, les déformations, si problématiques, seraient plus importantes. Une autre contrainte est le temps de calcul. Certains calculs prennent

plusieurs jours à tourner, comme celui du maillage grossier divisé par quatre et avec une tolérance accrue pour les solveurs linéaires.

Pour avoir un quart de panneau d'armoire, il faudrait que la pièce à modéliser ait au moins 0,2 m par 0,25 m. Avec ces dimensions, il serait intéressant de vérifier s'il y a des déformations de l'ordre du millimètre, entre autres dans le sens tangentiel. De plus, il faudrait essayer la modélisation avec des matériaux différents. L'érable est un bois franc qui n'a pas les mêmes propriétés par exemple, que le pin qui est un bois mou. Il serait intéressant de comparer les déformations avec les différentes essences de bois. De plus, il faudrait regarder l'impact de la colle dans des panneaux plus grands.

Ce qui a été montré cet été est que les fichiers à utiliser pour modéliser un quart de panneau d'armoire fonctionnent. De plus, avec les données recueillies et le logiciel Vu, il est possible de voir comment le bois sèche dans le temps. Grâce aux images dans le logiciel Vu, il est possible de voir que le bois sèche plus vite dans le sens tangentiel que longitudinal.

Pour les quatre dessins, une coupe horizontale a été prise à $z = 0,009$ m. On voit qu'il y a deux directions privilégiées de séchage, selon l'orientation du bois. La colle a un effet isolant. Le bois sèche moins vite près de la colle.



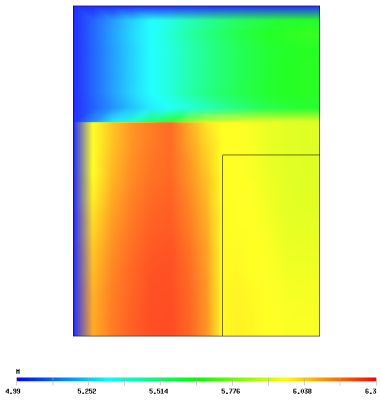


FIG. 13 – Taux d'humidité après 1 jour de séchage (figure de gauche)

FIG. 14 – Taux d'humidité après 2 jours de séchage (figure de droite)

Dans la partie centrale (celle dans le rectangle), il y a moins épais de bois. En plus de sécher dans le sens tangentiel, le bois sèche dans la direction radiale. On remarque aussi que le bois sèche dans le sens longitudinal près de la partie amovible, puisqu'il n'y a pas de colle.

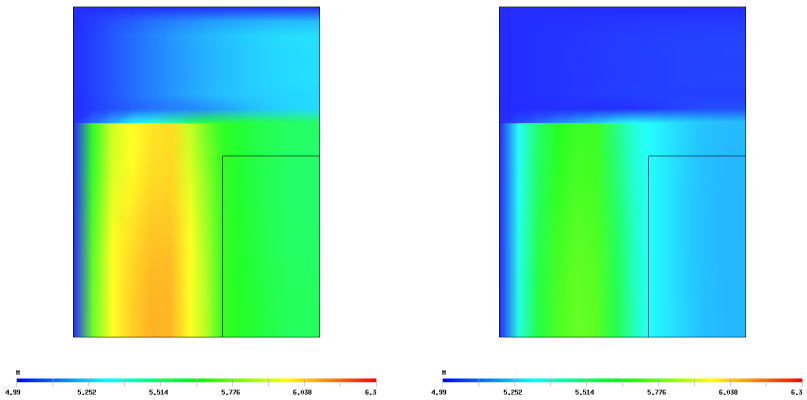


FIG. 15 – Taux d'humidité après 4 jours de séchage

FIG. 16 – Taux d'humidité après 8 jours de séchage

4 Conclusion

Ce stage se voulait un début pour montrer aux industries du bois la pertinence d'utiliser la méthode des éléments finis dans la fabrication d'objets en bois. Par exemple, une compagnie peut décider de ne pas partir une ligne de montage si elle sait qu'un objet aura des déformations majeures. La déformation d'un quart de panneau d'armoire a été simulée et elle a été montrée non problématique pour la géométrie utilisée. Dans les simulations futures, il serait intéressant d'utiliser une autre géométrie et d'autres matériaux. Il serait pertinent de comparer ces résultats avec ceux de la géométrie utilisée dans ce projet.

Du côté personnel, le stage au GIREF a été très formateur. J'ai beaucoup appris sur les éléments finis, domaine mathématique que je ne connaissais pas du tout. J'ai maintenant plus d'outils pour poursuivre mes études en mathématiques.

5 Références

Notes de cours *Les éléments finis : de la théorie à la pratique*, André Fortin, André Gagnon, 1997-2004

Rapport de stage de Mathieu Arbour

Fortin André, Analyse numérique pour ingénieurs, Presses internationales Polytechnique, 2001

6 Annexe

6.1 Fichier .champs

```
# Ceci est le fichier .champs qui sera utilisé pour résoudre
  le probleme du panneau d'amoire



#==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION
# #      temps     --> seconde # dimension --> mètre #
pression-->Gpa #
#==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION


# ----------------------------------------------------------------
# Nom des fichiers de sorties
  chainecar nomsortie resultats1
# ----------------------------------------------------------------

# Par défaut, le problème résout un problème d'humidité # on choisit
ensuite les autres types de problèmes que l'on veut
  résoudre
# Pour le panneau d'armoire, on veut aussi résoudre la Mécanique

booleen Thermique f
booleen Mecanique v

# ----------------------------------------------------------------
# On choisit de quelle manière on veut résoudre le problème

# constantes


# Temps de depart:                    T_0
# Temps final:                        T_F

scalaire T_0 0
scalaire T_F 3542400

# # Si les données ont des discontinuités en temps on va forcer une
  resolution a ces temps particuliers
# et on n'aura pas a se soucier des pas de temps, car le solveur
  fera le travail qu'on donne des pas de temps
# fixes ou variables.
```

```
#

# # Sert à  imposer des temps lors d'un calcul à  pas de temps
variable.
# booleen TempsImpose : sert à  activer l'imposition de
temps.
# scalaire NbTempsImpose : définit les temps que l'on veut
imposer.
  Prends les premiers de la liste.

booleen TempsImpose f
scalaire NbTempsImpose 0
scalaire TempsImpose_0 86400.
scalaire TempsImpose_1 259200.
scalaire TempsImpose_2 1209600.

# # Données pour le calcul avec pas de temps variable. # N.B: Si on
veut faire un calcul avec un pas de temps fixe, on a
  seulement besoin d'indiquer le temps initial (T_0 définit plus haut),
# d'inscrire le pas de temps (PasDeTemps)
# et le nombre de pas de temps (NbPasDetemps un peu plus bas) que
  l'on veut faire.
# Il ne faut pas oublier de mettre en commentaire la ligne
définissant le pas de temps variable. #

# Pas de temps (valeur initiale si variable): PasDeTempsInit
# Pas de temps minimal:                       PasDeTempsMin
# Pas de temps maximal (pas le limite = 0):   PasDeTempsMax

# ceci donne des pas de temps de 3 heures
scalaire PasDeTemps    10800
scalaire PasDeTempsMin 0.1
scalaire PasDeTempsMax 500000

# Nombre maximal de pas de temps: NbPasDeTemps # fréquence du calcul
en elasticite
  (relativement au problème en M: 0 == pas de calcul): freqCalcul
# fréquence d'impression des champs (pas d'impression = 0):
freqImpression


scalaire NbPasDeTemps    600
scalaire freqCalcul      1
scalaire freqImpression  1
```

```
#
# valeur specifique au solveurInstNlinPETSc et solveurLinPETSc
#

# tolerance pour le calcul des pas de temps
scalaire TolerancePasVariable 5e-3

# Tolerance accrue pour le solveur non linéaire # Tolerance pour la
convergence des solveurs scalaire ToleranceNlin 1e-9 scalaire
ToleranceLin 1e-12

# Tolerance accrue pour le solveur non linéaire
#initialement, a -9
et j'essaie a -8 scalaire ToleranceLinMec 1e-11 scalaire
ToleranceLinMecR 5e-40 scalaire Divergence 1e+10

# nombre maximum d'iterations pour les solveur
scalaire MaxItNlin 200
scalaire MaxItLin 6000
scalaire MaxItLinMec 6000


# # est-ce que l'on aura a résoudre un problème avec des mouvements
de  corps rigide? # a modifier si on sait que l'on aura des
mouvements de corps rigide. # # ATTENTION : il faudra alors modifier
dans le code suivant la direction CONNUE des mouvements de corps
rigide

booleen CorpsRigideGCP f

# # on veut la trace de la solution en certains points # NbEval
nombre de points d'evaluations # Coord_i le point i pour i=0 a
NbEval-1 # # en mettant NbEval à  zéro on ne fait plus les
evaluations


scalaire NbEval 6
#ajouter les autres coins
vectoriel3d Coord_0 [0, 0, 0]
vectoriel3d Coord_1 [0, 0, 0.009]
vectoriel3d Coord_2 [0, -0.1, 0]
vectoriel3d Coord_3 [0.05, -0.02, 0.009]
vectoriel3d Coord_4 [0.05, -0.07, 0.007]
```

```
vectoriel3d Coord_5 [0.07, -0.02, 0.009]


#-----------------------------------------------------------------

# On choisit le type de solveur qu'on utilise

##  Solveur PETSc  ##
# Pour plus d'information sur les solveurs, voir le lien
  "format du fichier champs" dans la documentation MEF++

solveurlinpetsc      typesolveur      "SolveurLin" gradient_conjugue
# solveurlinpetsc      typesolveur      "SolveurLin" direct
solveurlinpetsc      typeprecond      "SolveurLin" sor
solveurlinpetsc      configprecondsor "SolveurLin" [v]
solveurlinpetsc      suiviconvergence "SolveurLin" defaut
solveurlinpetsc      tolerance        "SolveurLin" [ToleranceLin,ToleranceLin,
                                                    Divergence,MaxItLin]

solveurinstnlinpetsc typesolveur                "SolveurInst"
solveurinstnlinpetsc solveurlin                 "SolveurInst" SolveurLin
solveurinstnlinpetsc parametresnlin             "SolveurInst" [ToleranceNlin,
                                                               MaxItNlin,v]
solveurinstnlinpetsc parametresinst "SolveurInst" [PasDeTemps,
                                    NbPasDeTemps, T_0, eulerimplicit,v]
solveurinstnlinpetsc reassemblagesafaire        "SolveurInst" [1,1]

# # Propriétes des matrices permettant d'optimiser la resolution
instationnaire # Ce sont des optimisations. Le code marchera sans
ces options
  (plus long cependant)
#
# PM0   la matrice masse depend du temps via C_MRho
# PM1   la matrice masse depend de la solution recherchee  via C_MRho
# PK0   la matrice rigidite depend du temps
# PK1   la matrice rigidite depend de la solution recherchee
# PK2   la matrice rigidite depend des CL
        (la matrice sera modifiee par les CL: robin <=> neumannonlineaire)
#
# solveurinstnlinpetsc parametres_matrice_masse    "SolveurInst" [PM0,PM1]
# solveurinstnlinpetsc parametres_matrice_rigidite "SolveurInst" [PK0,PK1,PK2]
#
solveurinstnlinpetsc parametres_matrice_masse    "SolveurInst" [f,f]
solveurinstnlinpetsc parametres_matrice_rigidite "SolveurInst" [f,v,v]
```

```
#
# on choisit un pas de temps variable calculÃ© par le solveur
# (commenter si on en veut pas, mais attention a NbPasDeTemps)
#
# solveurinstnlinpetsc pasdetempsvariable "SolveurInst"
  [PasDeTempsMin,PasDeTempsMax,T_F,TolerancePasVariable]


solveurlinpetsc      typesolveur      "SolveurMec" gradient_conjugue
# solveurlinpetsc      typesolveur      "SolveurMec" direct
solveurlinpetsc      typeprecond      "SolveurMec" sor
solveurlinpetsc      configprecondsor "SolveurMec" [v,0.7]
solveurlinpetsc      suiviconvergence "SolveurMec" defaut
solveurlinpetsc      tolerance        "SolveurMec"
[ToleranceLinMecR,ToleranceLinMec,Divergence,MaxItLinMec]

#
# FIN DES CONSTANTES, DEBUT DES CHAMPS OU DES CONSTANTES DEPENDANT D'AUTRES
  CONSTANTES
#

#-------------------------------------------------------------------------
# Définition du champ de transformation géométrique utilisé
  pour le calcul:

ChampGeoLin    "ChampGeo" ""

#-------------------------------------------------------------------------
# Definition de constantes

scalaire "zero" 0

#_________________________________________________________________________
# Partie diffusion non-stationnaire

# Humidité initiale. MP est utilisé en cas d'hystéresse, représente
l'humidité au pas précédent.

# ici scallin est utilisé pour la résolution linéaire. On utilise s

# calquad pour la résolution quadratique

 scallin "M" 6.3
```

```
 scallin "MP" 6.3

# scalhierarchique "M" 6.3
# scalhierarchique "MP" 6.3

# scalquad "M" 6.3
# scalquad "MP" 6.3

# Si on utilise DilEr* alors on indique au programme que BetaM tiendra compte
  de l'hystérese:
#

booleen Hysterese v

#-------------------------------------------------------------------
# Partie deformation
# déformation initiale

# ici v3dlin est utilisé pour la résolution linéaire. On utilise
v3dquad pour # la résolution quadratique

# v3dquad "U" [0,0,0]
# v3dhierarchique  "U" [0,0,0]
 v3dlin  "U" [0,0,0]

#-------------------------------------------------------------------------

# Champ nécessaire pour la CL
# MInfini varie dans le temps

# Ceci donne l'humidité ambiante


scalaire "MInfini" 5
#-------------------------------------------------------------------
# Definition des types de bois


# Definition d'un materiau "érable"
{ Erable_1

    scalaire    "db"        597
    scalaire    "DL"        1.3e-08
#   scalaire    "DR"        f(M)=(-6.34*M+40.1)*1.0e-10
```

```
    scalaire    "DR"        1.8e-11
#   scalaire    "DT"        f(M)=(-6.34*M+40.1)*1.0e-10
    scalaire    "DT"        1.8e-11
    scalaire    "KL"        f(DL,db)=DL*db/100
    scalaire    "KR"        f(DR,db)=DR*db/100
    scalaire    "KT"        f(DT,db)=DT*db/100
#   scalaire    "dKR"       -6.34*1.0e-10
#   scalaire    "dKT"       -6.34*1.0e-10
    scalaire    "dKR"       0
    scalaire    "dKT"       0
    scalaire    "dKL"       0
    scalaire    "EL"        13.810
    scalaire    "ER"        1.311
    scalaire    "ET"        0.678
    scalaire    "GLR"       1.013
    scalaire    "GLT"       0.753
    scalaire    "GRT"       0.255
    scalaire    "VTL"       0.025
    scalaire    "VTR"       0.43
    scalaire    "VLR"       0.46
    scalaire    "VLT"       0.50
    scalaire    "AL"        0.00014715
    scalaire    "AR"        0.0021382
    scalaire    "AT"        0.0033495
    scalaire    "BL"        0.0001842
    scalaire    "BR"        0.0019053
    scalaire    "BT"        0.0027579
    scalaire    "H_M"       3.2e-4
    scalaire    "C_MRho"    f(db)=db*1e-2
    empilementto2   "K_M"       [KL,KT,KR,zero,zero,zero]
    empilementto2   "dK_MdM"    [dKL,dKT,dKR,zero,zero,zero]
    to4elasticite   "Eijkl"     [EL,ET,ER,VLT,VTR,VLR,GLT,GRT,GLR]

    # Pour la dilatation qui est BetaM
    scalaire "DilErX" f(M,MP,BL,AL)=(M > MP)*BL+(M < MP)*AL
    scalaire "DilErY" f(M,MP,BT,AT)=(M > MP)*BT+(M < MP)*AT
    scalaire "DilErZ" f(M,MP,BR,AR)=(M > MP)*BR+(M < MP)*AR

    empilementto2 "BetaM" [DilErX,DilErY,DilErZ,zero,zero,zero]
}

# Definition d'un matériau "érable"
{ Erable_Perpendiculaire
```

```
    scalaire    "db"        597
    scalaire    "DL"        1.3e-08
#   scalaire    "DR"        f(M)=(-6.34*M+40.1)*1.0e-10
    scalaire    "DR"        1.8e-11
#   scalaire    "DT"        f(M)=(-6.34*M+40.1)*1.0e-10
    scalaire    "DT"        1.8e-11
    scalaire    "KL"        f(DL,db)=DL*db/100
    scalaire    "KR"        f(DR,db)=DR*db/100
    scalaire    "KT"        f(DT,db)=DT*db/100
#   scalaire    "dKR"       -6.34*1.0e-10
#   scalaire    "dKT"       -6.34*1.0e-10
    scalaire    "dKR"       0
    scalaire    "dKT"       0
    scalaire    "dKL"       0
    scalaire    "EL"        13.810
    scalaire    "ER"        1.311
    scalaire    "ET"        0.678
    scalaire    "GLR"       1.013
    scalaire    "GLT"       0.753
    scalaire    "GRT"       0.255
    scalaire    "VTL"       0.025
    scalaire    "VTR"       0.43
    scalaire    "VLR"       0.46
    scalaire    "VLT"       0.50
    scalaire    "AL"        0.00014715
    scalaire    "AR"        0.0021382
    scalaire    "AT"        0.0033495
    scalaire    "BL"        0.0001842
    scalaire    "BR"        0.0019053
    scalaire    "BT"        0.0027579
    scalaire    "H_M"       3.2e-4
    scalaire    "C_MRho"    f(db)=db*1e-2
    empilementto2   "K_M"       [KT,KL,KR,zero,zero,zero]
    empilementto2   "dK_MdM"    [dKT,dKL,dKR,zero,zero,zero]
    to4elasticite   "Eijkl"     [ET,EL,ER,VTL,VLR,VTR,GLT,GLR,GRT]

    # Pour la dilatation qui est BetaM
    scalaire "DilErX" f(M,MP,BT,AT)=(M > MP)*BT+(M < MP)*AT
    scalaire "DilErY" f(M,MP,BL,AL)=(M > MP)*BL+(M < MP)*AL
    scalaire "DilErZ" f(M,MP,BR,AR)=(M > MP)*BR+(M < MP)*AR

    empilementto2 "BetaM" [DilErX,DilErY,DilErZ,zero,zero,zero]
}
```

```
# Définition d'un matériau "colle"
{ Colle

    scalaire    "db"        1500
    scalaire    "DL"        1.0e-14
    scalaire    "DR"        1.0e-14
    scalaire    "DT"        1.0e-14
    scalaire    "KL"        f(DL,db)=DL*db/100
    scalaire    "KR"        f(DR,db)=DR*db/100
    scalaire    "KT"        f(DT,db)=DT*db/100
    scalaire    "dKL"       0
    scalaire    "dKR"       0
    scalaire    "dKT"       0
    scalaire    "EL"        10
    scalaire    "ER"        10
    scalaire    "ET"        10
    scalaire    "GLR"       3.703703704
    scalaire    "GLT"       3.703703704
    scalaire    "GRT"       3.703703704
    scalaire    "VTL"       0.35
    scalaire    "VTR"       0.35
    scalaire    "VLR"       0.35
    scalaire    "VLT"       0.35
    scalaire    "B"     0.019
    scalaire    "H_M"       3.2e-4
    scalaire    "C_MRho"    f(db)=db*1e-2
    empilementto2   "K_M"       [KL,KT,KR,zero,zero,zero]
    empilementto2   "dK_MdM"    [dKL,dKT,dKR,zero,zero,zero]
    to4elasticite   "Eijkl"     [ET,EL,ER,VTL,VLR,VTR,GLT,GLR,GRT]
    empilementto2   "BetaM"     [B,B,B,zero,zero,zero]
}


# ------------------------------------------------------------------

#champssurgeom Piece1=Erable_1 Piece2=Erable_Perpendiculaire
#              Piece3=Erable_Perpendiculaire Colle_01=Colle


champssurgeom Piece1=Erable_Perpendiculaire Piece2=Erable_Perpendiculaire
              Piece3=Erable_1 Colle_01=Colle
```

6.2 Fichiers .CL

6.2.1 Conditions limites pour le problème hydrique

```
neumannnonlin scalaire "Faces_a_humidite" "M" fonction(H_M,M,MInfini) =
                                                        [-H_M*(M-MInfini)]
```

6.2.2 Conditions limites pour le porblème mécanique

```
dirichlet scalaire "Symetriey" "UX" "zero"
dirichlet scalaire "Symetriex" "UY" "zero"
dirichlet scalaire "Point_fixe" "UZ" "zero"
```

6.3 Adimensionnalisation

Condition limite pour l'hydrique

$$\frac{dbD}{100\%}\nabla M\cdot\overrightarrow{n}=h_M\left(M-M_\infty\right)$$

On pose :

$$\begin{array}{rclcrcl}
\widehat{x} & = & \frac{x}{L_0} & \Longrightarrow & x & = & \widehat{x}L_0\\
\widehat{t} & = & \frac{t}{t_0} & \Longrightarrow & t & = & \widehat{t}t_0\\
\widehat{db} & = & \frac{db}{db_0} & \Longrightarrow & db & = & \widehat{db}db_0\\
\widehat{m} & = & \frac{m}{m_0} & \Longrightarrow & m & = & \widehat{m}m_0\\
\widehat{D} & = & \frac{D}{D_0} & \Longrightarrow & D & = & \widehat{D}D_0\\
\widehat{f} & = & \frac{f}{f_0} & \Longrightarrow & f & = & \widehat{f}f_0
\end{array}$$

Modèle pour l'hydrique

$$\frac{db}{100\%}\frac{\partial M}{\partial t}-\nabla\left(\frac{Ddb}{100\%}\nabla M\right)=f$$

devient

$$\frac{\widehat{db}db_0}{100}\frac{\partial\widehat{M}M_0}{\partial\widehat{t}t_0}-\frac{\widehat{\nabla}}{L_0}\left(\frac{\widehat{D}D_0\widehat{db}db_0}{100\%}\frac{\widehat{\nabla}}{L_0}\widehat{M}M_0\right)=f\widehat{f}$$

$$\frac{db_0M_0}{t_0}\frac{\widehat{db}}{100}\frac{\partial\widehat{M}}{\partial\widehat{t}}-\frac{D_0db_0M_0}{L_0^2}\widehat{\nabla}\cdot\left(\frac{\widehat{D}\widehat{db}}{100}\widehat{\nabla}\widehat{M}\right)=f\widehat{f}$$

On multiplie par $\frac{L_0^2}{D_0db_0M_0}$

$$\frac{L_0^2}{t_0D_0}\frac{\widehat{db}}{100}\frac{\partial\widehat{M}}{\partial\widehat{t}}-\widehat{\nabla}\cdot\left(\frac{\widehat{D}\widehat{db}}{100}\widehat{\nabla}\widehat{M}\right)f\widehat{f}$$

Condition limite hydrique au bord
Notre modèle :

$$\frac{dbD}{100}\nabla M\cdot\overrightarrow{n}=h_m\left(M-M_\infty\right)$$

Devient

$$\begin{aligned}\frac{db_0\widehat{db}D_0\widehat{D}}{100}\frac{\widehat{\nabla}}{L_0}M_0M\overrightarrow{n} &= h_0\hat{h}\left(M_0\widehat{M}-\widehat{M_\infty}M_0\right)\\ \frac{db_0\widehat{db}D_0\widehat{D}}{100}\widehat{\nabla}L_0M_0\widehat{M}\overrightarrow{n} &= h_0\hat{h}M_0\left(\widehat{M}-\widehat{M_\infty}\right)\\ \frac{db_0D_0M_0}{L_0}\frac{\widehat{db}\widehat{D}}{100}\widehat{\nabla}\widehat{M}\cdot\overrightarrow{n} &= h_0M_0\hat{h}\left(\widehat{M}-\widehat{M_\infty}\right)\end{aligned}$$

On multiplie par :
$\frac{L_0}{db_0D_0M_0}$
On obtient :

$$\frac{\widehat{db}\widehat{D}}{100}\widehat{\nabla}\widehat{M}\overrightarrow{n}=\frac{L_0h_0}{db_0D_0}\hat{h}\left(\widehat{M}-\widehat{M_\infty}\right)$$

On pose ensuite :
$D_0=\frac{L_0^2}{t_0}$
$f_0=\frac{db_0D_0M_0}{L_0^2}$
$h_0=\frac{D_0db_0}{L_0}$
et on a :

$$\frac{\widehat{db}}{100}\frac{\partial\widehat{M}}{\partial t}-\widehat{\nabla}\cdot\left(\frac{\widehat{db}\widehat{D}}{100}\widehat{\nabla}\widehat{M}=\hat{f}\right)$$

Pour l'armoire, les longueurs caractéristiques sont :
$L_0=0,1$ m
$t_0=3600\times 24$ secondes
$db_0=597\left(\frac{\text{kg}}{\text{m}^3}\right)$
$D_0=\frac{10^{-2}}{3600\times 24}\frac{\text{m}^2}{\text{s}}$
$h_0=\frac{597\times 1{,}6\times 10^{-7}}{10^{-1}}\left(\frac{\text{kg}}{\text{ms}\%}\right)$

Pour la partie mécanique, puisque le terme source est 0, alors, il n'est pas nécessaire de normaliser, car l'équation l'est déjà.

6.4 Fichier .champs adimensionnalisé

```
# Ceci est le fichier .champs qui sera utilise pour résoudre le
problème du panneau d'amoire # Ce fichier est adimensionnalisÃ© #
```

```
L'hydrique a été adimensionnalisée # La mécanique l'était déjà.



#==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION
# #      temps     --> seconde #      dimension --> mètre # pression
--> Gpa #
#==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION==ATTENTION


#-----------------------------------------------------------------
# Nom des fichiers de sorties
  chainecar nomsortie resultats1
#-----------------------------------------------------------------

# Par défaut, le problème résout un problème d'humidité # on choisit
ensuite les autres types de problèmes que l'on veut résoudre # Pour
le panneau d'armoire, on veut aussi résoudre la Mécanique

booleen Thermique f
booleen Mecanique v

#-----------------------------------------------------------------

# On choisit de quelle manière on veut résoudre le problème

# constantes


# Temps de depart:                    T_0
# Temps final:                        T_F

scalaire T_0 0
scalaire T_F 3542400

# # Si les données ont des discontinuités en temps on va forcer une
  resolution a ces temps particuliers
# et on n'aura pas a se soucier des pas de temps, car le solveur
  fera le travail qu'on donne des pas de temps fixes ou variables.

# # Sert \'a  imposer des temps lors d'un calcul \'a  pas de temps
variable.
# booleen TempsImpose : sert \'a  activer l'imposition de
```

```
temps.
# scalaire NbTempsImpose : d\'efinit les temps que l'on veut
imposer. Prends les premiers de la liste. #

booleen TempsImpose f
scalaire NbTempsImpose 0
scalaire TempsImpose_0 86400.
scalaire TempsImpose_1 259200.
scalaire TempsImpose_2 1209600.
scalaire TempsImpose_3 2332800.

# # Données pour le calcul avec pas de temps variable. # N.B: Si on
veut faire un calcul avec un pas de temps fixe, on a seulement
  besoin d'indiquer le temps initial (T_0 d\'efinit plus haut),
# d'inscrire le pas de temps (PasDeTemps)
# et le nombre de pas de temps (NbPasDetemps un peu plus bas) que l'on
  veut faire.
# Il ne faut pas oublier de mettre en commentaire la ligne
d\'efinissant
  le pas de temps variable.
#

# Pas de temps (valeur initiale si variable): PasDeTempsInit
# Pas de temps minimal:                       PasDeTempsMin
# Pas de temps maximal (pas le limite = 0):   PasDeTempsMax

scalaire PasDeTemps    0.125
scalaire PasDeTempsMin 0.1
scalaire PasDeTempsMax 500000

# Nombre maximal de pas de temps:                         NbPasDeTemps
# frequence du calcul en elasticite (relativement au probleme en M:
  0 == pas de calcul): freqCalcul
# frequence d'impression des champs (pas d'impression = 0):freqImpression

scalaire NbPasDeTemps    600
scalaire freqCalcul      1
scalaire freqImpression  1

#
# valeur specifique au solveurInstNlinPETSc et solveurLinPETSc
#

# tolerance pour le calcul des pas de temps
```

```
scalaire TolerancePasVariable 5e-3

# Tolerance accrue pourle solveur lineaire
# Tolerance pour la convergence des solveurs
scalaire ToleranceNlin 1e-9
scalaire ToleranceLin 5e-11

# Tolerance accure pour le solveur lineaire
#initialement, a -9 et j'essaie a -8
scalaire ToleranceLinMec 5e-10
scalaire ToleranceLinMecR 5e-40
scalaire Divergence 1e+10

# nombre maximum d'iterations pour les solveur
scalaire MaxItNlin 200
scalaire MaxItLin 5000
scalaire MaxItLinMec 5000


# # est-ce que l'on aura a résoudre un problème avec des mouvements
de  corps rigide? # a modifier si on sait que l'on aura des
mouvements de corps rigide. # # ATTENTION : il faudra alors modifier
dans le code suivant la direction
  CONNUE des mouvements de corps rigide

booleen CorpsRigideGCP f

# # on veut la trace de la solution en certains points # NbEval
nombre de points d'evaluations # Coord_i le point i pour i=0 a
NbEval-1
# en mettant NbEval Ã  zéro on ne fait plus les evaluations


scalaire NbEval 6
#ajouter les autres coins
vectoriel3d Coord_0 [0, 0, 0]
vectoriel3d Coord_1 [0, 0, 0.009]
vectoriel3d Coord_2 [0, -0.1, 0]
vectoriel3d Coord_3 [0.05, -0.02, 0.009]
vectoriel3d Coord_4 [0.05, -0.07, 0.007]
vectoriel3d Coord_5 [0.07, -0.02, 0.009]

#-----------------------------------------------------------------
```

```
# On choisit le type de solveur qu'on utilise

##  Solveur PETSc  ##
# Pour plus d'information sur les solveurs,
voir le lien "format du fichier champs" dans la documentation MEF++

#solveurlinpetsc      typesolveur      "SolveurLin" gradient_conjugue
solveurlinpetsc      typesolveur      "SolveurLin" direct
solveurlinpetsc      typeprecond      "SolveurLin" sor
solveurlinpetsc      configprecondsor "SolveurLin" [v]
solveurlinpetsc      suiviconvergence "SolveurLin" defaut
solveurlinpetsc      tolerance        "SolveurLin" [ToleranceLin,ToleranceLin,
                                                    Divergence,MaxItLin]

solveurinstnlinpetsc typesolveur              "SolveurInst"
solveurinstnlinpetsc solveurlin                "SolveurInst"
SolveurLin solveurinstnlinpetsc parametresnlin "SolveurInst"
                                               [ToleranceNlin,MaxItNlin,v]
solveurinstnlinpetsc parametresinst            "SolveurInst" [PasDeTemps,
                                    NbPasDeTemps,T_0, eulerimplicit,v]
solveurinstnlinpetsc reassemblagesafaire       "SolveurInst" [1,1]

# # Propriétés des matrices permettant d'optimiser la resolution
instationnaire # Ce sont des optimisations. Le code marchera sans
ces options (plus long cependant) # # PM0   la matrice masse depend
du temps via C_MRho # PM1   la matrice masse depend de la solution
recherchée  via C_MRho # PK0   la matrice rigidité depend du temps #
PK1   la matrice rigidité depend de la solution recherchée # PK2 la
matrice rigidité depend des CL (la matrice sera modifiée par les CL:
                                          robin <=> neumannonlineaire)
#
# solveurinstnlinpetsc parametres_matrice_masse    "SolveurInst" [PM0,PM1]
# solveurinstnlinpetsc parametres_matrice_rigidite "SolveurInst" [PK0,PK1,PK2]
#
solveurinstnlinpetsc parametres_matrice_masse    "SolveurInst" [f,f]
solveurinstnlinpetsc parametres_matrice_rigidite "SolveurInst" [f,v,v]

# # on choisit un pas de temps variable calculé par le solveur #
(commenter si on en veut pas, mais attention a NbPasDeTemps) # #
solveurinstnlinpetsc pasdetempsvariable "SolveurInst"
     [PasDeTempsMin,PasDeTempsMax,T_F,TolerancePasVariable]


#solveurlinpetsc      typesolveur      "SolveurMec" gradient_conjugue
```

```
solveurlinpetsc      typesolveur      "SolveurMec" direct
solveurlinpetsc      typeprecond      "SolveurMec" sor
solveurlinpetsc      configprecondsor "SolveurMec" [v,0.7]
solveurlinpetsc      suiviconvergence "SolveurMec" defaut
solveurlinpetsc      tolerance        "SolveurMec"
                [ToleranceLinMecR,ToleranceLinMec,Divergence,MaxItLinMec]

#
# FIN DES CONSTANTES, DEBUT DES CHAMPS OU DES CONSTANTES DEPENDANT D'AUTRES
  CONSTANTES
#

#-------------------------------------------------------------- #
Définition du champ de transformation géométrique utilisé pour le
calcul:

ChampGeoLin    "ChampGeo" ""

#-------------------------------------------------------------- #
Definition de constantes

scalaire "zero" 0

# Definition de constantes pour l'adimenditionnalisation

# La piece est de dimension 10cm
scalaire    "L_o"   0.1

scalaire    "T_o"   3600*24
scalaire    "M_o"   1
scalaire    "db_o"  597
scalaire    "D_o"   f(L_o,T_o)=(L_o)^2/T_o
scalaire    "H_M_o" f(D_o,db_o,L_o)=D_o*db_o/L_o

#_____________________________________________________________ #
Partie diffusion non-stationnaire

# Humidité initiale. MP est utilisée en cas d'hystéresse, représente
l'humidité au pas précédent. # ici scallin est utilisé pour la
résolution linéaire. On utilise scalquad pour
  la résolution quadratique

# scallin "M" 6.3
# scallin "MP" 6.3
```

```
# scalhierarchique "M" 6.3
# scalhierarchique "MP" 6.3

 scalquad "M" 6.3
 scalquad "MP" 6.3

# Si on utilise DilEr* alors on indique au programme que BetaM

tiendra compte de l'hysterese: #

booleen Hysterese v

#------------------------------------------------------------- #
Partie deformation # déformation initiale

# ici v3dlin est utilisé pour la résolution linéaire. On utilise
v3dquad pour  la résolution quadratique

 v3dquad "U" [0,0,0]
# v3dhierarchique  "U" [0,0,0]
# v3dlin  "U" [0,0,0]

#--------------------------------------------------------------------

# Champ nécessaire pour la CL # Ceci donne l'humidité ambiante


scalaire "MInfini" 5

#--------------------------------------------------------------------
# Definition des types de bois


# Definition d'un matériau "érable"
{ Erable_1

    scalaire    "db"        f(db_o)=597/db_o
    scalaire    "DL"        f(D_o)=1.3e-08/D_o
#   scalaire    "DR"        f(M,M_o,D_o)=(-6.34*M*M_o+40.1)*1.0e-10/D_o
    scalaire    "DR"        f(D_o)=1.8e-11/D_o
#   scalaire    "DT"        f(M,M_o,D_o)=(-6.34*M*M_o+40.1)*1.0e-10/D_o
    scalaire    "DT"        f(D_o)=1.8e-11/D_o
    scalaire    "KL"        f(DL,db)=DL*db/100
```

```
    scalaire    "KR"        f(DR,db)=DR*db/100
    scalaire    "KT"        f(DT,db)=DT*db/100
#   scalaire    "dKR"       f(M_o,D_o)=-6.34*M_o*1.0e-10/D_o
#   scalaire    "dKT"       f(M_o,D_o)=-6.34*M_o*1.0e-10/D_o
    scalaire    "dKR"       0
    scalaire    "dKT"       0
    scalaire    "dKL"       0
    scalaire    "EL"        13.810
    scalaire    "ER"        1.311
    scalaire    "ET"        0.678
    scalaire    "GLR"       1.013
    scalaire    "GLT"       0.753
    scalaire    "GRT"       0.255
    scalaire    "VTL"       0.025
    scalaire    "VTR"       0.43
    scalaire    "VLR"       0.46
    scalaire    "VLT"       0.50
    scalaire    "AL"        0.00014715
    scalaire    "AR"        0.0021382
    scalaire    "AT"        0.0033495
    scalaire    "BL"        0.0001842
    scalaire    "BR"        0.0019053
    scalaire    "BT"        0.0027579
    scalaire    "H_M"       f(H_M_o)=3.2e-4/H_M_o
    scalaire    "C_MRho"    f(db)=db*1e-2
    empilementto2   "K_M"       [KL,KT,KR,zero,zero,zero]
    empilementto2   "dK_MdM"    [dKL,dKT,dKR,zero,zero,zero]
    to4elasticite   "Eijkl"     [EL,ET,ER,VLT,VTR,VLR,GLT,GRT,GLR]

    # Pour la dilatation qui est BetaM
    scalaire "DilErX" f(M,MP,BL,AL)=(M > MP)*BL+(M < MP)*AL
    scalaire "DilErY" f(M,MP,BT,AT)=(M > MP)*BT+(M < MP)*AT
    scalaire "DilErZ" f(M,MP,BR,AR)=(M > MP)*BR+(M < MP)*AR

    empilementto2 "BetaM" [DilErX,DilErY,DilErZ,zero,zero,zero]
}

# Definition d'un matériau "érable"
{ Erable_Perpendiculaire

    scalaire    "db"        f(db_o)=597/db_o
    scalaire    "DL"        f(D_o)=1.3e-08/D_o
#   scalaire    "DR"        f(M,M_o,D_o)=(-6.34*M*M_o+40.1)*1.0e-10/D_o
    scalaire    "DR"        f(D_o)=1.8e-11/D_o
```

```
#   scalaire    "DT"        f(M,M_o,D_o)=(-6.34*M*M_o+40.1)*1.0e-10/D_o
    scalaire    "DT"        f(D_o)=1.8e-11/D_o
    scalaire    "KL"        f(DL,db)=DL*db/100
    scalaire    "KR"        f(DR,db)=DR*db/100
    scalaire    "KT"        f(DT,db)=DT*db/100
#   scalaire    "dKR"       f(M_o,D_o)=-6.34*M_o*1.0e-10/D_o
#   scalaire    "dKT"       f(M_o,D_o)=-6.34*M_o*1.0e-10/D_o
    scalaire    "dKR"       0
    scalaire    "dKT"       0
    scalaire    "dKL"       0
    scalaire    "EL"        13.810
    scalaire    "ER"        1.311
    scalaire    "ET"        0.678
    scalaire    "GLR"       1.013
    scalaire    "GLT"       0.753
    scalaire    "GRT"       0.255
    scalaire    "VTL"       0.025
    scalaire    "VTR"       0.43
    scalaire    "VLR"       0.46
    scalaire    "VLT"       0.50
    scalaire    "AL"        0.00014715
    scalaire    "AR"        0.0021382
    scalaire    "AT"        0.0033495
    scalaire    "BL"        0.0001842
    scalaire    "BR"        0.0019053
    scalaire    "BT"        0.0027579
    scalaire    "H_M"       f(H_M_o)=3.2e-4/H_M_o
    scalaire    "C_MRho"    f(db)=db*1e-2
    empilementto2   "K_M"       [KT,KL,KR,zero,zero,zero]
    empilementto2   "dK_MdM"    [dKT,dKL,dKR,zero,zero,zero]
    to4elasticite   "Eijkl"     [ET,EL,ER,VTL,VLR,VTR,GLT,GLR,GRT]

    # Pour la dilatation qui est BetaM
    scalaire "DilErX" f(M,MP,BT,AT)=(M > MP)*BT+(M < MP)*AT
    scalaire "DilErY" f(M,MP,BL,AL)=(M > MP)*BL+(M < MP)*AL
    scalaire "DilErZ" f(M,MP,BR,AR)=(M > MP)*BR+(M < MP)*AR

    empilementto2 "BetaM" [DilErX,DilErY,DilErZ,zero,zero,zero]
}

# Definition d'un matériau "colle"
{ Colle

    scalaire    "db"        f(db_o)=1500/db_o
```

```
    scalaire    "DL"        f(D_o)=1.0e-14/D_o
    scalaire    "DR"        f(D_o)=1.0e-14/D_o
    scalaire    "DT"        f(D_o)=1.0e-14/D_o
    scalaire    "KL"        f(DL,db)=DL*db/100
    scalaire    "KR"        f(DR,db)=DR*db/100
    scalaire    "KT"        f(DT,db)=DT*db/100
    scalaire    "dKL"       0
    scalaire    "dKR"       0
    scalaire    "dKT"       0
    scalaire    "EL"        10
    scalaire    "ER"        10
    scalaire    "ET"        10
    scalaire    "GLR"       3.703703704
    scalaire    "GLT"       3.703703704
    scalaire    "GRT"       3.703703704
    scalaire    "VTL"       0.35
    scalaire    "VTR"       0.35
    scalaire    "VLR"       0.35
    scalaire    "VLT"       0.35
    scalaire    "B"     0.019
    scalaire    "H_M"       f(H_M_o)=3.2e-4/H_M_o
    scalaire    "C_MRho"    f(db)=db*1e-2
    empilementto2   "K_M"       [KL,KT,KR,zero,zero,zero]
    empilementto2   "dK_MdM"    [dKL,dKT,dKR,zero,zero,zero]
    to4elasticite   "Eijkl"     [ET,EL,ER,VTL,VLR,VTR,GLT,GLR,GRT]
    empilementto2   "BetaM"     [B,B,B,zero,zero,zero]
}


#-----------------------------------------------------------------

#champssurgeom Piece1=Erable_1 Piece2=Erable_Perpendiculaire
#              Piece3=Erable_Perpendiculaire Colle_01=Colle
champssurgeom Piece1=Erable_Perpendiculaire Piece2=Erable_Perpendiculaire
              Piece3=Erable_1 Colle_01=Colle
```

6.5 Fichiers .CL adimensionnalisés

```
neumannnonlin scalaire "Faces_a_humidite" "M" fonction(H_M,M,MInfini) =
```

```
                                                    [-H_M*(M-MInfini)]

dirichlet scalaire "Symetriey" "UX" "zero"
dirichlet scalaire "Symetriex" "UY" "zero"
dirichlet scalaire "Point_fixe" "UZ" "zero"
```