

Fabrice Bernard-Granger

Jérémy Oullier

Stage d'informatique
pour la maîtrise de physique recherche
de l'université Joseph Fourier, Grenoble.

Modélisation d'un problème de bulles magnétiques

Stage effectué au laboratoire Louis Néel du CNRS de Grenoble
sous la tutelle de Monsieur Jean-Christophe Toussaint
enseignant-chercheur de l'école d'ingénieur INPG de Grenoble.

Stage effectué pendant l'année universitaire 1999-2000

Sommaire

1. Introduction.....	3
2. Approche informatique générale.....	3
2.1 Effet de l'ordinateur.....	3
2.2 Effet de l'intégration d'une équation.....	4
2.3 Principe de résolution d'équations différentielles.....	4
2.4 Méthodes de résolution.....	4
2.4.1 Pour des équations différentielles.....	4
2.4.2 Pour l'intégration de polynômes.....	6
3. Approche physique du problème, mise en équation.....	6
3.1 Domaine physique considéré.....	7
3.1.1 Origine du magnétisme.....	7
3.1.2 Ferromagnétisme.....	7
3.2 Modélisation.....	7
3.2.1 Positionnement du problème, en toutes généralités.....	7
3.2.2 Mouvement de la paroi.....	8
3.2.2.1 Approximation.....	8
3.2.2.2 Phénomène physique.....	9
3.2.2.3 Mise en équations.....	9
3.2.3 Evolution de l'énergie libre.....	11
3.2.3.1 Pourquoi l'énergie libre.....	11
3.2.3.2 Mise en équation.....	11
4. Résolution de l'équation dans l'espace des coordonnées.....	12
5. Résolution de l'équation dans l'espace temps.....	12
5.1 premier exemple de résolution.....	13
5.2 Dernier exemple de résolution.....	13
6. Conclusion.....	14
 Annexe.....	 15
1 Programme.....	16
2 Méthodes d'intégration.....	27
3 Méthode de Gauss.....	40
4 Résultat d'une simulation.....	43

1. Introduction

Dans le cadre du magistère de physique recherche de Grenoble, nous avons effectué un stage de modélisation numérique dans le domaine des bulles magnétiques. Pour ce faire nous avons travaillé au laboratoire Louis Néel sous la direction de Mr J.C Toussaint, responsable de la simulation numérique. Ce laboratoire, sous l'égide du CNRS, est issu de l'élan novateur apporté par Mr Néel dans le domaine du magnétisme sur Grenoble.

Le but de nos travaux a été de mettre au point un code numérique permettant de simuler l'évolution d'une paroi immatérielle séparant deux domaines d'aimantation opposés, avec une approche ampérienne et non coulombienne, dans le système d'unité MKSA.

Dans une première partie, il sera présenté les problèmes de résolution ayant attiré à une grande partie de modélisation numérique pour l'appliquer à notre sujet. Toute cette approche nécessite quelques explications concernant la modélisation physique que l'on abordera ensuite. Et enfin, on expliquera la résolution numérique avec ces défauts et ces contraintes.

Dans la conclusion nous ferons un bilan de ce stage tant d'un point de vue physique que personnel, ainsi que les développements technologiques possible de ce phénomène.

2 Aspect informatique général

Le but le plus courant, dans de telle recherche, est de résoudre un problème physique numériquement, problème qui est explicité par des équations différentielles. C'est-à-dire qu'on va les résoudre en faisant faire un grand nombre de calcul par un ordinateur. Il y a donc deux effets qui s'imposent, et c'est à nous de poser le problème de façon à ce que ces inconvénients ne nous fausse pas nos résultats. Ces effets sont les suivants :

- effet de l'ordinateur, lié aux calculs.
- effet de l'intégration d'une équation.

2.1 Effet de l'ordinateur

Ce problème est incurable. Quand un ordinateur traite des nombres, il est obligé de les tronquer à partir d'un certain nombre de chiffres après la virgule. Du fait

qu'il n'a pas une mémoire infinie. Donc, plus on fait des calculs plus on incrémente l'erreur de troncature. Pour affaiblir cette erreur, il faut minimiser le nombre de calcul que fait le programme. On ne s'occupera pas de ce problème dans notre traitement vu qu'il ne nous a pas gêné en premier lieu. C'est-à-dire que cette erreur commise était beaucoup plus petite que celle qu'on commettait dans l'approximation de notre intégration.

2.2 Effet de l'intégration d'une équation

Notre programme va donc intégrer une équation aux dérivées partielles qui dépend de certaines variables (temps, espace). Cette intégration va pouvoir être conduite de diverse façon dont l'ordre de l'approximation en dépendra, c'est à nous de choisir la plus efficace pour notre programme. Celle avec laquelle le nombre de calcul fait par l'ordinateur est convenable (quoique ce problème a été mis de côté dans notre étude), et celle qui nous apportera la meilleure approximation dans l'intégration des variables. En d'autres termes, la divergence des solutions obtenues au bout d'un certain temps d'intégration (qui est grand dans notre résolution) est due à l'erreur commise par la méthode d'intégration.

2.3 Principe de résolution d'équations différentielles

Les étapes à suivre sont les suivantes :

- séparer les variables d'espaces et de temps
- intégrer l'équation par rapport aux seules variables d'espaces
- intégrer l'équation par rapport à la variable temporelle.

Les intégrations se font par des méthodes adéquates.

2.4 Méthodes de résolution

2.4.1 Pour des équations différentielles

Pratiquement toutes les intégrations utilisent le développement de Taylor d'une fonction d'une certaine variable en un point et un pas donné de la variable. Soit le développement de Taylor d'une fonction scalaire f :

$$f(a + da) = f(a) + \frac{\partial f(a)}{\partial a} da + \frac{1}{2} \frac{\partial^2 f(a)}{\partial a^2} da^2 + \dots$$

$$f(a - da) = f(a) - \frac{\partial f(a)}{\partial a} da + \frac{1}{2} \frac{\partial^2 f(a)}{\partial a^2} da^2 + \dots$$

Où 'a' est la variable et 'da' son pas.

Ce qui nous donne en première approximation des relations sur les dérivées partielles :

$$\frac{\partial f(a)}{\partial a} = \frac{f(a + da) - f(a - da)}{2 da}$$

$$\frac{\partial^2 f(a)}{\partial a^2} = \frac{f(a + da) + f(a - da) - 2 f(a)}{da^2}$$

On remarque que cela nous conduit à une discrétisation de notre espace et de notre fonction. L'exemple présenté nous montre la discrétisation des opérateurs considérés en un point avec deux premiers voisins. On pourra élargir cette définition à des opérateurs et des fonctions vectorielles et scalaires à plusieurs dimensions. Ainsi on pourra résoudre de nombreuses équations comportant des opérateurs tels que le gradient, la divergence, le rotationnel ... si le contexte le permet. Dans notre cas nous garderons cette définition en mémoire. En effectuant une discrétisation, il ne faut pas oublier les problèmes liés aux bords. Or, dans notre problème, l'objet discrétisé est fermé, nous avons donc des conditions aux limites périodiques, en comparaison. On verra que dans pratiquement toute discrétisation de l'espace (coordonnées ou temps) un problème surgit. En reprenant les dernières équations exposées, on peut faire encore une approximation, soit :

$$\frac{\partial f(a)}{\partial a} = \frac{f(a + da) - f(a)}{da}$$

$$\frac{\partial f(a)}{\partial a} = \frac{f(a) - f(a - da)}{da}$$

La première équation sera utilisée pour la résolution de l'opérateur dans l'espace des temps et la seconde pour l'espace des coordonnées.

Diverses méthodes d'intégration en temps existent :

- Euler (développement de Taylor au premier ordre)

$$f(t + dt) = f(t) + dt * \frac{\partial f(t)}{\partial t}$$

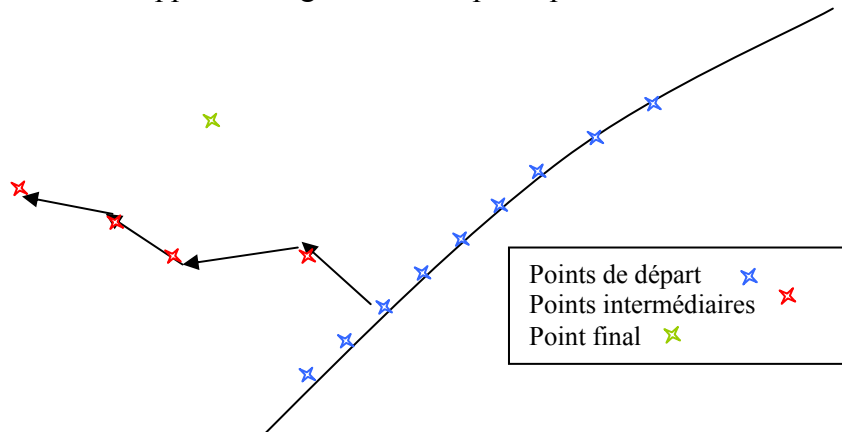
Avec : f(t) la valeur de la fonction au temps t.

f(t+dt) la valeur de la fonction au temps t+dt.

'dt' le pas de temps choisi pour la méthode et la précision.

- Runge & Kutta (méthodes des points intermédiaires)

Cette méthode est une approximation de la fonction en un pas supérieur par des calculs de points intermédiaires, le nombre de point intermédiaire définissent l'ordre de la méthode (ordre équivalent a celui du développement de Taylor). Les points intermédiaires sont calculés par la méthode d'Euler, ce qui nécessite par exemple le calcul des positions $f(t)$ et des forces $\frac{\partial f(t)}{\partial t}$ pour chaque points. Le résultats au pas de temps supérieure sera une combinaison linéaire associée aux points intermédiaires. Cette méthode est appelée Runge & Kutta à pas séparés. Voici le schéma explicatif :



L'inconvénient de cette méthode, c'est quelle nécessite énormément de calcul (calcul de la fonction à intégrer en chaque point intermédiaire). Pour remédier à cet inconvénient, il existe une même méthode, Runge & Kutta à pas liés (méthode qui ne sera pas utilisée dans ce stage).

Un document explicatif est en annexe pour plus de précision.

2.4.2 Pour l'intégration de polynôme

Intégrer un polynôme de manière numérique suppose qu'on connaît sa valeur en tout point dans l'espace des coordonnées continues. La méthode de Gauss permet d'éliminer cette contrainte et permet ainsi d'intégrer un polynôme en ne connaissant qu'un certain nombre de point (ce qui est utile pourvu que l'on ait un espace discret). Cette méthode considère que le résultat de l'intégration du polynôme s'écrit par combinaison linéaire de ce polynôme en différent point.

Un document explicatif est en annexe pour plus de précision.

3 Aspect physique

Comme dans tous travaux de simulation numérique il faut bien cerner le problème physique afin de le modéliser de la manière la plus juste et la plus proche de

la réalité, en ne tenant compte que des phénomènes prépondérants. Ce qui nous amènera à faire des approximations, nécessaires. Ces modifications de la réalité seront le plus possible explicites.

3.1 Domaine physique considéré

3.1.1 Origine du magnétisme

Les premières observations d'attraction / répulsion magnétique ont été observées en Chine plusieurs siècles avant JC, il furent par la suite utilisés pour l'orientation marine (sensibilité de la magnétite au champ magnétique terrestre). Ce n'est que bien plus tard qu'une utilisation industrielle s'est développée, après la compréhension théorique (Faraday, Ampère) et la mise en équations (Maxwell).

3.1.2 Ferromagnétisme

Consécutivement à l'application d'un champ magnétique extérieur la matière a une certaine réponse qui dépend des caractéristiques du matériau considéré, un des comportements magnétiques est le ferromagnétisme, celui-ci va gouverner les phénomènes étudiés par la suite. Une phase ferromagnétique est une phase ordonnée du point de vue magnétique c'est à dire qu'en dessous d'une certaine température (T_c : température de Curie), les sites magnétiques présentent un moment non nul ce qui se traduit macroscopiquement par une aimantation. Cette phase fait apparaître dans le solide différents domaines (d'aimantation up et down) afin de minimiser l'énergie du système (pour annuler la contribution du champ démagnétisant), équilibre du système. On se retrouve donc avec une structure en domaine dont on va étudier l'évolution, c'est la recherche de l'équilibre.

3.2 Modélisation

3.2.1 Positionnement du problème, en toutes généralités

La configuration de départ est décrite sur le schéma suivant :

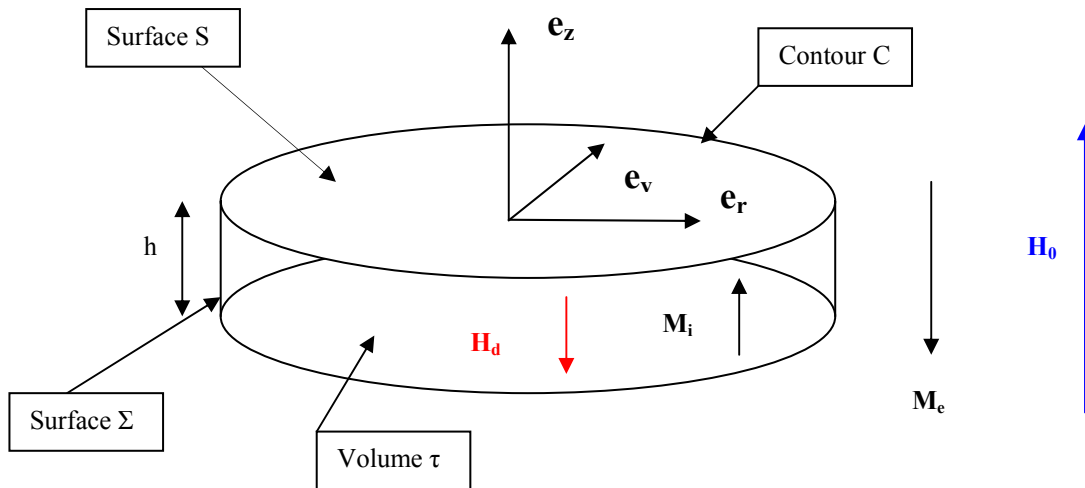


Figure 1

Paramètres fixés :

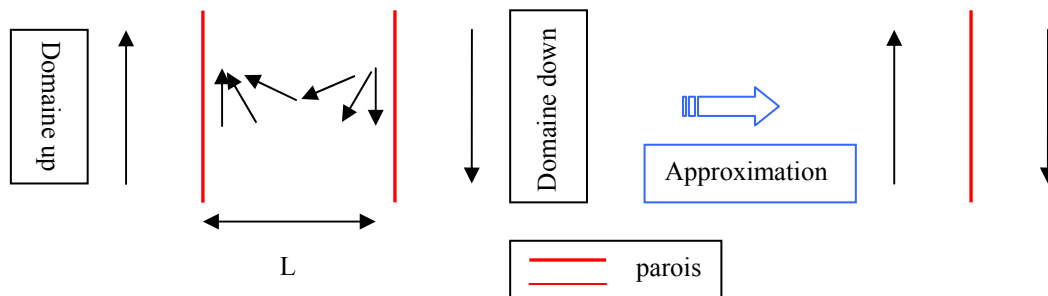
- une forme donnée de domaine, choix pour le départ (recherche de l'évolution d'une telle structure).
- un champ extérieur ainsi que les aimantations données (paramètre du problème).
- notons que ce domaine s'inscrit dans un solide de plus grande dimension.

3.2.2 Mouvement de la paroi

On regarde l'évolution de la paroi (i.e. du pourtour C) au cours du temps. En fait on verra plus loin que l'on ne considère que le mouvement de C car on moyenne les grandeurs sur la hauteur 'h'.

3.2.2.1 Approximation

On considère une paroi de forme cylindrique, invariant suivant la direction z, confère la figure 1. On va négliger l'épaisseur de la paroi par rapport à la grande taille des domaines (confère la figure ci-après). En fait, on ne s'occupe pas de l'énergie contenue dans les parois, même si elle peut rentrer en compte, faiblement.



Avec ces approximations, on se ramène donc à un problème plan. On considère aussi que le domaine extérieur est infini, sa taille n'entre pas en compte dans le problème car on s'intéresse uniquement aux interactions sur la paroi. Voici une figure représentative de notre problème en pensant qu'il y a des forces appliquées sur la paroi, et ces forces que l'on va essayer d'équilibrer par le biais de l'informatique. Notons aussi que l'épaisseur de la paroi 'L' n'est pas considérée ce qui compliquerait énormément le problème (point de Brillouin).

3.2.2.2 Phénomène physique

Sur la paroi fictive on peut considérer qu'elle subie deux contraintes, une d'origine magnétique et une de tension de surface (toujours moyennée sur la hauteur 'h', donc se ramenant à une tension linéique). Comme on veut regarder l'évolution de cette paroi il faut lui donner une loi d'évolution temporelle qui ne peut être la loi de la dynamique de Newton puisque que celle-ci n'a pas de masse, on la modélise par une relaxation proportionnelle à la vitesse d'évolution.

Si l'on rentre plus en détail dans l'expression de ces différentes grandeurs il faut que l'on caractérise le champ magnétique global qui règne dans l'élément de solide considéré. A première vue nous n'avions pensé qu'au champ externe H_0 , mais il faut tenir compte des interactions entre dipôles magnétiques (microscopique) regroupées dans une grandeur que l'on appelle le champ démagnétisant H_d .

Pour le calcul de H_d on utilise la loi de Biot et Savart gouvernée par des densités courants (uniquement surfacique ' Σ ' dans notre cas, car en volume l'aimantation est constante donc rotationnel de \mathbf{M} est nul).

3.2.2.3 Mise en équation

Nous utiliserons pour tous le problème les coordonnées cylindriques comme définies sur la figure 1.

- Equation d'évolution spatial : (équation tirée de certains calculs dont on passera les détails).

$$\delta f = 0 \Rightarrow \gamma \vec{n} - 2\mu_0 MH \partial_\theta (\vec{OM} \wedge \vec{u}_y) = \vec{0}$$

avec

$$\vec{n} = -\partial_\theta \left(\frac{\partial_\theta \vec{OM}}{|\partial_\theta \vec{OM}|} \right) = -\partial_\theta \vec{t}$$

Avec γ qui est le paramètre de la tension linéique

$\vec{OM}(\theta)$ la position de l'élément considéré

\vec{M} l'aimantation

\vec{H} le champ global tel que $\vec{H} = \vec{H}_0 + \vec{H}_{d \text{ ou } m}$

\vec{n} vecteur normal à la paroi

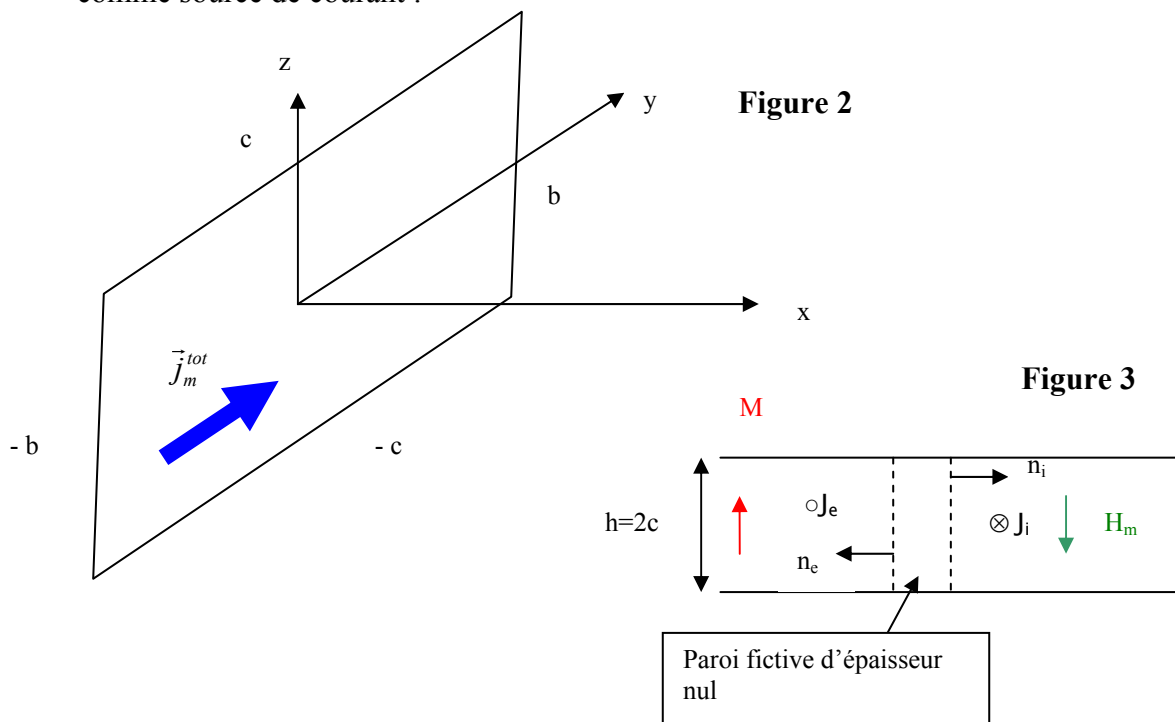
\vec{t} vecteur tangentielle à la paroi

et

$$\vec{H}_d = \frac{1}{4\pi} \iint_{\Sigma} d\Sigma \frac{\vec{j}_m^s \wedge \vec{r}}{r^3} \quad \text{où} \quad \vec{j}_m^s = \vec{M} \wedge \vec{n} \quad \text{densité de courant}$$

Afin de tenir compte des deux contributions des aimantations (interne et externe) le courant total de surface sera multiplié par deux.

Pour le calcul du champ démagnétisant on va utiliser une méthode de Gauss mais il faut au préalable retravailler l'intégrant, c'est à dire considérer un petit élément plan comme source de courant :



$$\langle \vec{H}_m \rangle = \frac{1}{2c} \int_{-c}^c dz \vec{H}_m(x, z) \quad \text{moyenné sur la hauteur}$$

il reste à intégrer sur y ce que l'on fait avec Gauss sur la fonction :

$$f(y', x, y) = \frac{-x(x^2 + (y - y')^2 + 4c^2)^{1/2}}{c(x^2 + (y - y')^2)} + \frac{x}{c(x^2 + (y - y')^2)^{1/2}}$$

- Equation d'évolution spatial :

$$\vec{f} = \alpha \partial_t \vec{OM}$$

avec α le paramètre d'évolution temporelle.

3.2.3 Evolution de l'énergie libre

3.2.3.1 Pourquoi l'énergie libre

Ce paramètre représente la seule grandeur qui nous permet de juger de la fiabilité de notre modélisation, en effet notre système doit évoluer de telle manière à minimiser son énergie.

Comme à l'intérieur du domaine l'aimantation est constante, l'énergie libre est dépendante des courants surfaciques à l'interface entre domaines (figure 3). Dans notre étude, nous ne considérons seulement que 2 domaines 'up' et 'down', pour l'un des deux domaines la surface est rejetée à l'infini donc sa contribution à l'énergie est nulle.

Dans le calcul de l'énergie il faut compter la contribution des deux champs H_0 et H_m mais comme H_0 est constant on ne traite que le cas du démagnétisant (l'énergie est définie à une constante près). Cette énergie est en fait due à l'interaction dipôle-dipôle entre tous les moments dipolaires du volume considéré.

3.2.3.2 Mise en équation

À une constante près on a pour le terme démagnétisant :

$$E = -\mu_0 \frac{1}{2} \int_{\tau} \vec{B} \vec{M} d\tau \quad \text{avec } \vec{B} = \vec{\nabla} \wedge \vec{A}$$

Le demi vient du fait que c'est une interaction dipolaire et qu'il ne faut pas compter deux fois la même énergie.

En retravaillant un peu l'expression via le théorème de la divergence on obtient :

$$E = \frac{M}{2} \oint_{\Sigma} \vec{t} \cdot \vec{A} d\Sigma = \frac{M}{2} h \int_0^{2\pi} \frac{\partial O\vec{M}}{\partial \theta} \cdot \vec{A} d\theta$$

Notre étude s'est arrêtée à ce stade sans que nous puissions implémenter le calcul de cette énergie. Pour le faire il faut exprimer le potentiel vecteur comme suit puis calculer l'intégrale avec la méthode de Gauss comme pour le champ démagnétisant la seule chose qui change est la fonction $f(x,y,z)$.

4 Résolution de l'équation dans l'espace des coordonnées

Soit l'équation régissant le problème dans l'espace des coordonnées :

$$\beta \vec{n} - B \frac{\partial O\vec{M} \wedge \vec{u}_z}{\partial \theta} = \vec{f} \quad \vec{n} = - \frac{\frac{\partial O\vec{M}}{\partial \theta}}{\left| \frac{\partial O\vec{M}}{\partial \theta} \right|} = - \frac{\partial \vec{t}}{\partial \theta}$$

Avec β une constante du problème. \vec{f} est la force appliquée à la paroi. C'est une équation dynamique, à l'équilibre la force est nulle.

Pour trouver le vecteur normal, on a appliquée simplement la méthode d'Euler avec l'équation définie en première partie. Voir la partie *calcul du vecteur normal* dans le programme mis en annexe.

Pour le calcul B qui est explicité dans la partie physique, on a utilisé la méthode de Gauss. Cette partie a été faite par Mr Toussaint par manque de temps dans notre stage (un papier explicatif est en annexe sur cette méthode). Sinon, voir la partie *calcul de B par la méthode de Gauss* dans le programme.

Ainsi, on a fait la première partie du problème, intégrer l'équation par rapport aux seules variables d'espaces. Maintenant, il ne nous reste plus qu'à regarder l'intégration de l'équation en temps.

5 Résolution de l'équation dans l'espace temps

Dans cet objectif, il faut résoudre l'équation suivante :

$$\vec{f} = \alpha \frac{\partial O\vec{M}}{\partial t}$$

Avec α un paramètre du système, et la force qui est calculable par ce que l'on a vu précédemment.

5.1 Premier exemple de résolution

Dans ce cas, c'est la méthode d'Euler qui a été mise à contribution. Soit donc, par cette approximation :

$$\vec{f} = \alpha \frac{O\vec{M}(t + dt) - O\vec{M}(t)}{dt}$$

Avec dt le pas de temps. Ce qui nous donne :

$$O\vec{M}(t + dt) = O\vec{M}(t) + \frac{dt}{\alpha} \vec{f}$$

Ainsi avec l'expression de la force, on pourra faire évoluer notre en déterminant la position de la paroi au temps t+dt, après il faudra recalculer la force avec ces nouvelles positions, et ainsi chercher le minimum en énergie.

On a pris pour exemple, une forme de cercle pour la paroi. On peut déjà faire une remarque qui n'est liée à la méthode d'Euler. Le cercle est un objet totalement symétrique par rapport à θ , on peut donc supposer que ce dernier n'évoluera pas avec le temps, mais plutôt que ce sera r. Mais, pourquoi pas que dans l'évolution le cercle tourne sur lui-même, c'est à vérifier. Par réflexion, le fait que le cercle tourne ne pourra en aucun cas faire évoluer l'énergie. On suppose aussi que le cercle est la forme la plus pour une telle paroi, forme à obtenir en fait d'évolution.

On remarque bien que le cercle tourne, mais on n'a pas de solutions stables. Et surtout, on met peu de temps pour que le système diverge. C'est-à-dire que le cercle s'élargi jusqu'à l'infinie en changeant de forme.

Donc cela signifie que cette première méthode n'est pas la bonne.

5.1 Dernier exemple de résolution

La méthode utilisée est celle de Runge & Kutta à pas séparés à l'ordre 4. C'est-à-dire qu'on a 4 points intermédiaires à calculer par la méthode d'Euler pour déterminer la position au temps dt supérieur qui est une combinaison linéaire de ces 4 points. Voir la partie rk4 dans le programme mis en annexe.

Avec cette méthode on n'a plus eut de problème lié à l'intégration. On a aussi commencé à faire une simulation pour une forme circulaire de la paroi. Aucun problème ne surgit. Il fallait attendre au moins 3 heures après avoir trouver le point stable pour que le système diverge à cause de la méthode d'intégration. Puis on est passé à une forme ellipsoïdale pour la paroi. Pour de faible excentricité, on retrouvait

bien le point stable. Mais pour des fortes, la paroi prenait une forme de huit sans points de superposition. On observait aussi un regroupement de point, problème lié à la discrétisation. (voir résultat d'une simulation en annexe)

Nous en somme arrivé là, à la fin de notre stage.

6 Conclusion

Nous ne sommes pas arrivés au terme de notre modélisation mais on peut d'ores et déjà dire que la méthode numérique utilisée pour discrétiser la courbe est mal adaptée car il apparaît des divergences dans certaines grandeurs lorsqu'il y a 'accumulation' de points à un endroit donné du domaine. Pour pallier à ce problème il faudrait utiliser une routine qui redistribue à chaque évolution les points de discrétisation. Ce qui pourrait améliorer nos résultats pour d'autres formes de la paroi. Enfin, il serait intéressant d'implanter le calcul complet de l'énergie dans le programme pour voir si elle tend bien vers un minimum.

Du point de vue personnel, ce stage a bien complété le module d'informatique de la maîtrise, notamment sur les problèmes de divergence de solution uniquement dû à la méthode de révolution. De plus le fait de travailler au sein d'un laboratoire nous a permis de mieux connaître les rouages de la recherche française.

ANNEXE

Le programme

```
#include <iostream>
#include <fstream>
#include <complex>
#include <stdio.h>

#include <stdlib.h>
#include <math.h>

#define Npts 100
#define Niter 1000000
#define EPS 1.e-8
#define DEBUG 0

const double pi=acos(-1.);

struct point
{
double x;
double y;
};

struct vect
{
double x;
double y;
};

//-----0
complex <double> argth(complex <double> z)
{
complex <double> I(0., 1.);
return 0.5*log(1.+z+I*EPS)-0.5*log(1.-z+I*EPS);
}

//-----1
double norme(vect v)
{
return sqrt(v.x*v.x+v.y*v.y);
}

//-----2
void InitPosition(point *M)
{
for (int i=0; i<Npts; i++)
{
double theta=2.*pi/Npts*i;
M[i].x=80.*cos(theta)*(1.+0.2*cos(8.*theta));
M[i].y=80.*sin(theta)*(1.+0.2*cos(8.*theta));
}
}
```

```
}

```

Début partie : calcul du champ démagnétisant par la méthode de Gauss

```
//-----3
double dfbm(double x, double y, double yp, double b, double c)
{
double fct;
if (fabs(x)<EPS) return 0.;

fct=-x/c*sqrt(x*x+(y-b)*(y-b)+4.*c*c)/(x*x+(y-b)*(y-b));
fct+=x/c/sqrt(x*x+(y-b)*(y-b));

return -2.*fct/(4.*pi);
}

```

fonction lié à Gauss

```
//-----4
double fbm(double x, double y, double b, double c)
{
double fct;
if (fabs(x)<EPS) return 0.;
double w=0.5+1./6./sqrt(6./5.);
double ksi=sqrt((3.-2.*sqrt(6./5.))/7.);

fct=w*(dfbm(x, y, -b*ksi, b, c)+dfbm(x, y, b*ksi, b, c));

w=0.5-1./6./sqrt(6./5.);
ksi=sqrt((3.+2.*sqrt(6./5.))/7.);

fct+=w*(dfbm(x, y, -b*ksi, b, c)+dfbm(x, y, b*ksi, b, c));

return fct*b;
}

```

```
//-----5

```

```
void Calbm(double epais, point *M, vect *t, double *bm)
{
double c=epais/2.;
double dth=2.*pi/Npts;
vect deltaM;
vect Milieu;

for (int i=0; i<Npts; i++)
{
bm[i]=0.;

for (int j=0; j<Npts-1; j++)
{

```

```

deltaM.x=M[j+1].x-M[j].x;
deltaM.y=M[j+1].y-M[j].y;
Milieu.x=(M[j+1].x+M[j].x)/2.;
Milieu.y=(M[j+1].y+M[j].y)/2.;
double b=norme(deltaM)/2.;

double X=M[i].x-Milieu.x;
double Y=M[i].y-Milieu.y;
vect n;
n.x=-t[j].y;
n.y= t[j].x;
double y=X*t[j].x+Y*t[j].y;
double x=X*n.x +Y*n.y;
#if DEBUG
cout<<x<<"t"<<y<<"t"<<fbm(x, y, b, c)<<endl;
#endif
bm[i]+=fbm(x, y, b, c);
}

deltaM.x=M[0].x-M[Npts-1].x;
deltaM.y=M[0].y-M[Npts-1].y;
Milieu.x=(M[0].x+M[Npts-1].x)/2.;
Milieu.y=(M[0].y+M[Npts-1].y)/2.;
double b=norme(deltaM)/2.;

double X=M[i].x-Milieu.x;
double Y=M[i].y-Milieu.y;
vect n;
n.x=-t[Npts-1].y;
n.y= t[Npts-1].x;
double y=X*t[Npts-1].x+Y*t[Npts-1].y;
double x=X*n.x +Y*n.y;
#if DEBUG
cout<<x<<"t"<<y<<"t"<<fbm(x, y, b, c)<<endl;
#endif

bm[i]+=fbm(x, y, b, c);

}
#if DEBUG
for (int i=0; i<Npts; i++)
cout<<"#noeud "<<i<<"t"<<bm[i]<<endl;
exit(1);
#endif
}

```

Fin partie : calcul du champ démagnétisant par la méthode de Gauss

```

//-----6
void CaldMdth(point *M, vect *dMdth)

```

```

{
double dth=2.*pi/Npts;

dMdh[0].x=(M[1].x-M[Npts-1].x)/2./dth;
dMdh[0].y=(M[1].y-M[Npts-1].y)/2./dth;

for (int i=1; i<Npts-1; i++)
    {
    dMdh[i].x=(M[i+1].x-M[i-1].x)/2./dth;
    dMdh[i].y=(M[i+1].y-M[i-1].y)/2./dth;
    }
dMdh[Npts-1].x=(M[0].x-M[Npts-2].x)/2./dth;
dMdh[Npts-1].y=(M[0].y-M[Npts-2].y)/2./dth;
}

//-----7

void Caltangte(point *M, vect *t)
{
double dth=2.*pi/Npts;
double tnorme;

for (int i=0; i<Npts-1; i++)
    {
    t[i].x=(M[i+1].x-M[i].x)/dth;
    t[i].y=(M[i+1].y-M[i].y)/dth;
    tnorme=norme(t[i]);
    t[i].x/=tnorme;
    t[i].y/=tnorme;
    }

t[Npts-1].x=(M[0].x-M[Npts-1].x)/dth; //pour des pblemes de continuer
t[Npts-1].y=(M[0].y-M[Npts-1].y)/dth;
tnorme=norme(t[Npts-1]);
t[Npts-1].x/=tnorme;
t[Npts-1].y/=tnorme;
}

//-----8

void Calnormale(vect *t, vect* n)
{
double dth=2.*pi/Npts;
double nnorme;

n[0].x=-(t[0].x-t[Npts-1].x)/dth; // pour la continuer
n[0].y=-(t[0].y-t[Npts-1].y)/dth;

for (int i=1; i<Npts; i++)

```

```

    {
    n[i].x=-(t[i].x-t[i-1].x)/dth;
    n[i].y=-(t[i].y-t[i-1].y)/dth;
    }
}

//-----9
void Calforce(double p1,double b, double *bm, vect *dMdth, vect *n, vect *force)
{
for (int i=0; i<Npts; i++)          // p2 parametre physique
    {
    force[i].x=-p1*n[i].x+(b+bm[i])*dMdth[i].y; // b est le champ externe et bm le
    champ demagnetisant
    force[i].y=-p1*n[i].y-(b+bm[i])*dMdth[i].x;
    }
}

//-----10
void Evolution(double p2,double dt, vect *force, point *M)
{
for (int i=0; i<Npts; i++)          // p2 parametre physique
    {
    M[i].x+=dt*force[i].x/p2;
    M[i].y+=dt*force[i].y/p2;
    }
}

```

Début partie : rk4

```

//----- 10 bis
void evolution2(double p1,double p2,double b,double *bm,double dt,double ep,point
*M,vect *dMdt,vect *dMdtheta,vect
*Normale,vect *tangte, vect *force)
{

```

le tableau M represente les points initiaux dont on cherche sa valeur après l'intégration.

```

//declaration des pterus internes a la fonction
point *K0=new point [Npts];          points tampon
point *K1=new point [Npts];          déclaration des
point *K2=new point [Npts];          tableaux de points
point *K3=new point [Npts];          intermédiaires
point *K4=new point [Npts];

```

```

if ((!K4)||(!K0)||(!K1)||(!K2)||(!K3)) message d'erreur lié à l'allocation de mémoire
{
    cerr<<"Pb alloc"<<endl;
    exit(1);
}

```

calcul de la force par rapport au tableau M de points

```

CaldMdth(M, dMdtheta);
Caltangte(M, tangte);
Calnormale(tangte, Normale);
Calbm(ep, M, tangte, bm);
Calforce(p1,b, bm, dMdtheta, Normale, force);

```

Calcul du tableau des premiers points intermédiaires K1

```

for(int i=0;i<Npts;i++)
{
  K1[i].x=dt*force[i].x;  EULER
  K1[i].y=dt*force[i].y;
}
for(int i=0;i<Npts;i++) // pour tempotaire cf impossible de faire une somme sur les
structures
{
  K0[i].x=M[i].x+K1[i].x/2.;
  K0[i].y=M[i].y+K1[i].y/2.;
}

```

calcul de la force par rapport au tableau K0 des 1er points intermédiaires

```

CaldMdth(K0, dMdtheta);          //--recalcul de la force1
Caltangte(K0, tangte);
Calnormale(tangte, Normale);
Calbm(ep, K0, tangte, bm);
Calforce(p1,b, bm, dMdtheta, Normale, force);

```

Calcul du tableau des premiers points intermédiaires K2

```

for(int i=0;i<Npts;i++)
{
  K2[i].x=dt*force[i].x;  EULER
  K2[i].y=dt*force[i].y;
}
for(int i=0;i<Npts;i++)
{
  K0[i].x=M[i].x+K2[i].x/2.;
  K0[i].y=M[i].y+K2[i].y/2.;
}

```

calcul de la force par rapport au tableau K0 des 2eme points intermédiaires

```

CaldMdth(K0, dMdtheta);          //--recalcul de la force2
Caltangte(K0, tangte);
Calnormale(tangte, Normale);
Calbm(ep, K0, tangte, bm);
Calforce(p1,b, bm, dMdtheta, Normale, force);

```

Calcul du tableau des premiers points intermédiaires K3

```

for(int i=0;i<Npts;i++)
{
  K3[i].x=dt*force[i].x;  EULER

```

```

    K3[i].y=dt*force[i].y;
  }
for(int i=0;i<Npts;i++)
{
  K0[i].x=M[i].x+K3[i].x;
  K0[i].y=M[i].y+K3[i].y;
}

```

calcul de la force par rapport au tableau K0 des 3eme points intermédiaires

```

CaldMdh(K0, dMdtheta);      //--recalcul de la force3
Caltangte(K0, tangte);
Calnormale(tangte, Normale);
Calbm(ep, K0, tangte, bm);
Calforce(p1,b, bm, dMdtheta, Normale, force);

```

Calcul du tableau des premiers points intermédiaires K4

```

for(int i=0;i<Npts;i++)
{
  K4[i].x=dt*force[i].x;   EULER
  K4[i].y=dt*force[i].y;
}

```

Calcul de M après l'intégration

```

for(int i=0;i<Npts;i++)
{
  M[i].x=M[i].x+(1./6.)*(K1[i].x+2.*K2[i].x+2.*K3[i].x+K4[i].x);
  M[i].y=M[i].y+(1./6.)*(K1[i].y+2.*K2[i].y+2.*K3[i].y+K4[i].y);
}
}

```

Fin partie : rk4

```

//-----11
void Sauvecfg(point *M, double *bm, vect *force)
{
  ofstream cible("position1", ios::out);
  for (int i=0; i<Npts; i++)
  {
    cible<<M[i].x<<"\t"<<M[i].y<<"\t"<<bm[i]
      <<"\t"<<force[i].x<<"\t"<<force[i].y<<endl;
  }
}

```

```

//-----12
double CalEnergSurf(point *M, vect *dMdh)
{
  double dth=2.*pi/Npts;
  double E=0.;
  vect pm;

  for (int i=0; i<Npts; i++)

```

```

    {
        E+=norme(dMdth[i]);
    }
return E*dth;
}

//-----13
double CalEnergVol(point *M, vect *dMdth)
{
double dth=2.*pi/Npts;
double E=0.;
vect pm;

for (int i=0; i<Npts; i++)
    {
        E+=-0.5*fabs(M[i].x*dMdth[i].y-M[i].y*dMdth[i].x);
    }
return E*dth;
}

//-----PROGRAMME
main()
{
cout<<" valeur de gamma /(2*mu0*M) : "<<endl; // gamma intensite de la force de
tension
double p1;
cin>>p1;

//cout<<" valeur de alpha /(2*mu0*M) : "<<endl; // alpha parametre d'evolution
temporel
double p2=1.0;
//cin>>p2;

cout<<"valeur du induction b"<<endl;
double b;
cin>>b;

cout<<"epaisseur de la couche"<<endl;
double ep;
cin>>ep;

cout<<"valeur de dt"<<endl;
double dt;
cin>>dt;

point *M=new point [Npts];
if (!M)
    {
        cerr<<"Pb alloc"<<endl;
        exit(1);
    }
}

```

```
}

vect *dMdt=new vect [Npts];
if (!dMdt)
{
    cerr<<"Pb alloc"<<endl;
    exit(1);
}

double *bm=new double [Npts];
if (!bm)
{
    cerr<<"Pb alloc"<<endl;
    exit(1);
}

vect *force=new vect [Npts];
if (!force)
{
    cerr<<"Pb alloc"<<endl;
    exit(1);
}

vect *dMdtheta=new vect [Npts];
if (!dMdtheta)
{
    cerr<<"Pb alloc"<<endl;
    exit(1);
}

vect *tangte=new vect [Npts];
if (!tangte)
{
    cerr<<"Pb alloc"<<endl;
    exit(1);
}

vect *Normale=new vect [Npts];
if (!Normale)
{
    cerr<<"Pb alloc"<<endl;
    exit(1);
}

InitPosition(M);
ofstream cible("evolforce1", ios::out); // creation du fichier evol

int n=0;
for (;) // equivalent a for ( int n=0 ; n++) la boucle ne s'arrete pas
```

```
{
  evolution2 (p1,p2,b,bm,dt,ep,M,dMdt,dMtheta,Normale,tangte,force);
  if (!(n%20)) // sauvegarde toute les 20 boucles
  {
    Sauvecfg(M, bm, force); // le fichier est out

    cible<<n<<"\t"<<force[0].x<<"\t"<<force[0].y<<"\t"<<bm[0]<<endl;
    // le fichier est evolforce
  }
  n++;
}
}
```

Méthode d'intégration

EQUATIONS DIFFERENTIELLES

1- INTRODUCTION

De nombreuses méthodes de résolution d'équations aux dérivées partielles sont fondées sur le principe suivant :

- Séparer les variables d'espace et de temps,
- Intégrer l'équation par rapport aux seules variables d'espace par une méthode de différences finies, d'éléments finis ou d'intégrales de frontières,
- Intégrer alors l'équation différentielle obtenue en fonction de la seule variable temporelle.

Avant d'aborder les méthodes de résolution d'équations aux dérivées partielles nous allons donc traiter les équations différentielles.

La plupart des méthodes de discrétisation que nous verrons conduisent à un système différentiel sous la forme matricielle :

$$[B] \left\{ \frac{\partial Y}{\partial t} \right\} = [A] \{Y\} + \{C\} \quad (1)$$

où $[A]$ et $[B]$ sont des matrices dont les termes peuvent dépendre de l'inconnue.

Nous étudierons donc plus en détail les méthodes les plus classiques appliquées à l'équation de la forme (1).

2 - QUELQUES DEFINITIONS

2.1 - Equivalence entre équation et système :

Rappel :

Une équation différentielle de degré n peut s'exprimer sous la forme d'un système de n équations différentielles d'ordre 1 :

$$g\left(y, \frac{\partial y}{\partial t}, \frac{\partial^2 y}{\partial t^2}, \dots, \frac{\partial^n y}{\partial t^n}, t\right) = 0 \tag{2}$$

est équivalent à :

$$\left\{ \begin{array}{l} Y_1 = \frac{\partial y}{\partial t} \\ Y_2 = \frac{\partial Y_1}{\partial t} = \frac{\partial^2 y}{\partial t^2} \\ \vdots \\ Y_{n-1} = \frac{\partial Y_{n-2}}{\partial t} = \frac{\partial^{n-1} y}{\partial t^{n-1}} \\ g\left(y, Y_1, Y_2, \dots, \frac{\partial Y_{n-1}}{\partial t}, t\right) = 0 \end{array} \right. \tag{3}$$

dont les conditions initiales seront :

pour $t = t_0$, (y, Y_1, \dots, Y_{n-1}) sont connues.

En conséquence, dans ce qui suit, nous ne traiterons que le cas des systèmes différentiels d'ordre 1.

2.2 - Forme explicite, forme implicite :

Soit le système différentiel $\vec{g}\left(\frac{\partial \vec{Y}}{\partial t}, \vec{Y}, t\right) = \vec{0}$.

On dit qu'on peut le mettre sous forme explicite (ou canonique) si on peut l'écrire sous la forme :

$$\frac{\partial \vec{Y}}{\partial t} = \vec{f}(\vec{Y}, t) \tag{4}$$

Exemple : le système

$$[B] \frac{\partial \vec{Y}}{\partial t} - [A] \vec{Y} - \vec{C} = \vec{0} \tag{5}$$

peut être explicité seulement si $[B]^{-1}$ existe.

2.3 - Taille d'un système :

Nous appellerons système de faible taille un système comportant moins d'une cinquantaine d'équations, de taille moyenne les systèmes comportant entre 50 et 500 équations, les grands systèmes pouvant atteindre 100 000 équations différentielles à résoudre simultanément.

3 - METHODES EXPLICITES A PAS SEPARES - RUNGE ET KUTTA

Ce sont les plus utilisées dans les cas des systèmes de faible taille, elles peuvent être appliquées aux systèmes de taille moyenne.

3.1 - Principe général :

On va construire la fonction $\vec{Y}(t)$ point par point en choisissant un pas h supposé constant. Soit n le dernier point calculé : connaissant la valeur de \vec{Y}_n on va calculer la valeur \vec{Y}_{n+1} au point $n+1$. Pour cela on définit des points intermédiaires afin de pouvoir approcher les dérivées qui interviennent dans les développements en série si on veut obtenir une précision élevée.

3.2 - Première approche :

On choisit pour formule donnant \vec{Y}_{n+1} le développement en série de Taylor sans modification.

La plus simple de ces méthodes est bien sûr la formule d'Euler :

$$\vec{Y}_{n+1} = \vec{Y}_n + h \vec{f}(t_n, \vec{Y}_n)$$

dont la précision n'est que du 1^{er} ordre. L'erreur est alors en $\frac{h^2}{2} \vec{f}''(\xi)$.

3.2.1 - Stabilité de la méthode :

A chaque pas de calcul la valeur obtenue comporte une erreur provenant d'une part de la **troncature** dans la série (erreur de méthode) et d'autre part des calculs imprécis dus à l'ordinateur (erreur **d'arrondi**).

Nous allons étudier la propagation de ces erreurs, d'un pas à l'autre sur une équation de la forme :

$$\frac{\partial \vec{Y}}{\partial t} = [A] \vec{Y} + \vec{C} \quad (6)$$

Pour laquelle on supposera $[A]$ et \vec{C} indépendants de \vec{Y} et t .

Soit $\vec{\varepsilon}_n$ l'erreur commise au pas n , on pose :

$$\vec{Y}_n = \vec{Y}_n^* + \vec{\varepsilon}_n \quad (7)$$

où \vec{Y}_n^* est la valeur supposée exacte.

3.2.2 - Cas de la méthode d'Euler :

Le schéma numérique s'écrit :

$$\vec{Y}_{n+1} = \vec{Y}_n + h \vec{f}(t_n, \vec{Y}_n) \quad (8)$$

donc :

$$\vec{Y}_{n+1} = \vec{Y}_n + h ([A] \vec{Y}_n + \vec{C}) \quad (9)$$

ce qui conduit à :

$$\vec{Y}_{n+1} = \vec{Y}_n^* + h ([A] \vec{Y}_n^* + \vec{C}) + \vec{\varepsilon}_n + h [A] \vec{\varepsilon}_n \quad (10)$$

d'où par définition :

$$\vec{\varepsilon}_{n+1} = \vec{\varepsilon}_n + h [A] \vec{\varepsilon}_n \quad (11)$$

en notant $[I]$ la matrice identité

$$\vec{\varepsilon}_{n+1} = \{[I] + h [A]\} \vec{\varepsilon}_n \quad (12)$$

et donc par récurrence :

$$\vec{\varepsilon}_n = \{[I] + h [A]\}^n \vec{\varepsilon}_0 \quad (13)$$

nous devons donc analyser les valeurs de :

$$\{[I] + h [A]\}^n \text{ quand } n \rightarrow \infty.$$

Soit $[\alpha]$ la matrice diagonale correspondant à : $[A] = [P^{-1}] [\alpha] [P]$ où $[P]$ et $[P^{-1}]$ sont les matrices de changement de base diagonalisantes de $[A]$.

Alors $[P^{-1}] \{[I] + h [\alpha]\} [P] = [I] + h [A]$. Les valeurs propres de $[I] + h [A]$ sont donc sous la forme $1 + h \lambda_i$ où λ_i sont les valeurs propres de $[A]$. Donc,

$$\{[I] + h [A]\}^n = [P^{-1}] \{[I] + h [\alpha]\}^n [P]$$

Soit λ_m la plus grande valeur propre de $[A]$ (rayon spectral de la matrice). On obtient alors la condition suffisante :

$$|1 + h \lambda_m| < 1 \Rightarrow \{[I] + h [A]\}^n \xrightarrow{n \rightarrow \infty} 0$$

d'où $\vec{\varepsilon}_n \rightarrow \vec{0}$ quand $n \rightarrow \infty$.

Remarque :

Le pas h étant toujours positif on obtient comme condition :

$$\lambda_i < 0 \quad \forall i$$

et

$$-1 - h \lambda_m < 1$$

$$\text{d'où} \quad -2 < h \lambda_m < 0$$

$h |\lambda_m| < R = 2$, rayon de stabilité de la méthode,

$$\text{d'où } h < \frac{2}{|\lambda_m|}$$

La méthode sera donc stable pour un pas défini par la taille de la plus grande des valeurs propres de [A] en valeur absolue.

Exemple :

Sur l'équation $y' = -y$ comparer les approximations données pour les différentes valeurs de h : 0.5, 1, 2, 3.

3.3 - Deuxième approche :

On veut obtenir une précision d'ordre 2. On a $\frac{df}{dt} = \frac{\partial f}{\partial Y} \frac{\partial Y}{\partial t} + \frac{\partial f}{\partial t}$ d'où :

$$Y_{n+1} = Y_n + h f(Y_n, t_n) + \frac{h^2}{2} (f'_Y f + f'_t) + \dots$$

où l'on peut remarquer que le terme $(f'_Y f + f'_t)$ est déjà assez compliqué et va conduire à des formules pénibles à programmer. On imagine à l'ordre 4 !!!
(Rem : les notations vectorielles sont ici oubliées de façon à alléger les expressions)

Idée → approchons les dérivées d'ordre 2 et plus par des combinaisons linéaires d'accroissements.

3.3.1 - Exemple à l'ordre 2 :

$$Y_{n+1} = Y_n + \alpha_1 \Delta Y_{n1} + \alpha_2 \Delta Y_{n2}$$

avec ΔY_{n1} et ΔY_{n2} deux accroissements de Y_n calculés en deux points différents de l'intervalle $[t_n, t_{n+1}]$ à l'aide de la formule du 1^{er} ordre seulement.

Principe : On va chercher la position de ces points et les valeurs des coefficients α_1, α_2 pour que le résultat obtenu coïncide avec la série de Taylor jusqu'à l'ordre 2.

Exemple sur tangente améliorée :

$$Y_{n+1} = Y_n + h f(Y_n, t_n) + \frac{h^2}{2} (f'_Y f + f'_t)$$

$$Y_{n+1} = Y_n + h f(Y_n, t_n) + \frac{h^2}{2} f'_Y(Y_n, t_n) f(Y_n, t_n) + \frac{h^2}{2} f'_t(Y_n, t_n)$$

On prend un point en t_n et l'autre en $t_n + \alpha_0 h$.

$$Y_{n+1} = Y_n + \alpha_1 h f(Y_n, t_n) + \alpha_2 h f\left(Y_n + \alpha_0 h f(Y_n, t_n), t_n + \alpha_0 h\right)$$

$$f\left(Y_n + \alpha_0 h f(Y_n, t_n), t_n + \alpha_0 h\right) = f(Y_n, t_n) + \alpha_0 h f(Y_n, t_n) f'_Y(Y_n, t_n)$$

$$+ \alpha_0 h f'_t(Y_n, t_n)$$

d'où en développant :

$$Y_{n+1} = Y_n + \alpha_1 h f(Y_n, t_n)$$

$$+ \alpha_2 h f(Y_n, t_n)$$

$$+ \alpha_2 h \alpha_0 h f(Y_n, t_n) f'_Y(Y_n, t_n)$$

$$+ \alpha_2 h \alpha_0 h f'_t(Y_n, t_n)$$

On identifie avec Taylor, ce qui conduit à

$$\begin{cases} \alpha_1 + \alpha_2 = 1 \\ \alpha_2 \alpha_0 = \frac{1}{2} \end{cases}$$

Il y a une infinité de solutions à ce système. On choisit celle qui minimise le nombre de calculs et les erreurs (α_i du même ordre et de même signe). Par exemple :

$$\begin{cases} \alpha_1 = 0 \\ \alpha_2 = 1 \\ \alpha_0 = \frac{1}{2} \end{cases}$$

Ce qui se traduit par le schéma suivant :

$$Y_{n+1} = Y_n + h f\left(Y_n + \frac{h}{2} f(Y_n, t_n), t_n + \frac{h}{2}\right)$$

3.3.2 - Stabilité de la méthode d'ordre 2 (RK₂₂) :

On reprend comme à l'ordre 1, on obtient :

$$\bar{\varepsilon}_{n+1} = \left\{ [I] + h[A] + \frac{h^2}{2} [A]^2 \right\} \bar{\varepsilon}_n$$

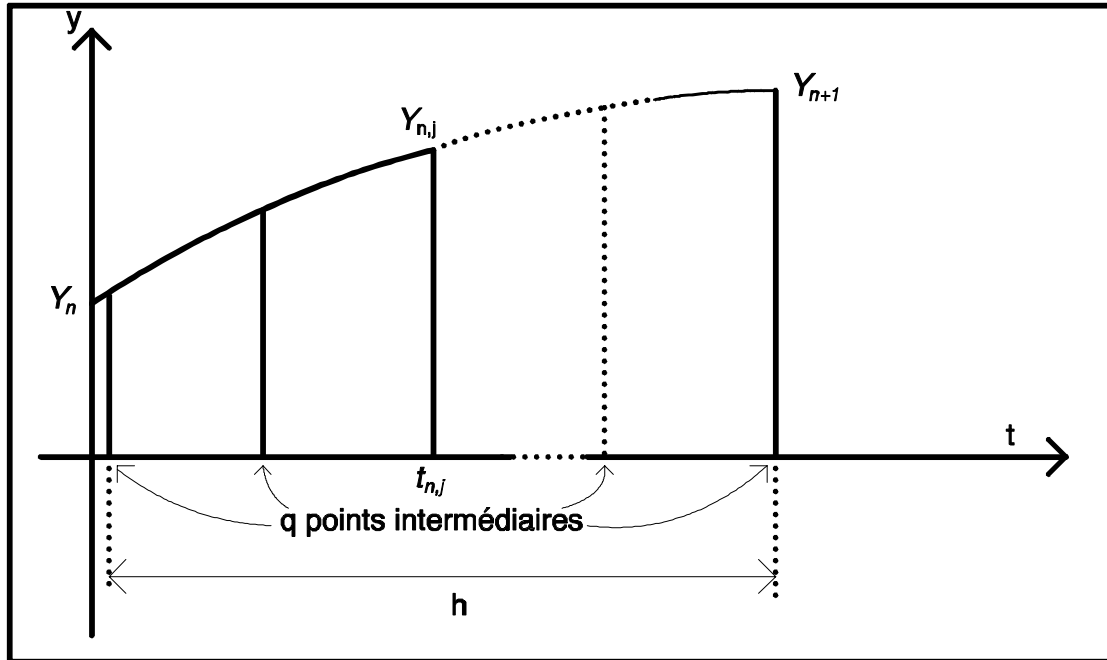
d'où avec $z = h\lambda_i$

$$\bar{\varepsilon}_{n+1} = \left(1 + z + \frac{z^2}{2} \right)^n \bar{\varepsilon}_0$$

d'où le rayon $R = 2$ comme pour la tangente, car :

$$1 + z + \frac{z^2}{2} < 1 \Rightarrow z \left(1 + \frac{z}{2}\right) < 0 \Rightarrow -2 < z < 0.$$

3.4 - Méthodes d'ordres supérieures :



$$\bar{Y}_{n+1} = \bar{Y}_n + h \sum_{j=1}^q b_j \bar{f}(t_{n,j}, \bar{Y}_{n,j}) \quad (14)$$

La valeur finale de l'accroissement est une combinaison linéaire des accroissements calculés en chaque point intermédiaire.

De même les valeurs des ordonnées $\bar{Y}_{n,j}$ aux points intermédiaires sont calculées par une combinaison linéaire de même type :

$$\bar{Y}_{n,j} = \bar{Y}_{t_n + \xi_j h} = \bar{Y}_n + h \sum_{k=1}^l a_{jk} \bar{f}(t_n + \xi_k h, \bar{Y}_{n,k})$$

Le rang de la méthode est le nombre de points intermédiaires pris en compte.

Pour déterminer la valeur numérique des coefficients des combinaisons linéaires, on établit un développement en série de Taylor de la fonction et on identifie terme à terme ce développement et la formule d'approximation (14).

L'ordre de la méthode est l'ordre atteint dans la série de Taylor au cours de l'identification.

Les systèmes obtenus sont souvent surabondants et on est en présence d'infinités de solutions, ce qui permet de faire intervenir d'autres critères dans le choix des coefficients :

- * Minimisation du nombre d'opérations à réaliser (coût du calcul)
- * Optimisation de la stabilité de la méthode (nous reviendrons sur ce point au paragraphe suivant)
- * Minimisation des erreurs d'arrondi dans les calculs (coefficients tous de même signe et si possible de grandeurs relatives les plus proches possibles).

Classiquement c'est l'algorithme d'ordre 4 et de rang 4 qui est le plus souvent choisi car il offre un bon compromis entre tous ces critères ; on obtient alors :

$$\begin{aligned}
 \vec{K}_1 &= h \vec{f}(t_n, \vec{Y}_n) & \text{avec} & \quad \xi_1 = 0 \\
 \vec{K}_2 &= h \vec{f}\left(t_n + \frac{h}{2}, \vec{Y}_n + \frac{\vec{K}_1}{2}\right) & \quad \quad \quad & \xi_2 = \frac{1}{2} \\
 \vec{K}_3 &= h \vec{f}\left(t_n + \frac{h}{2}, \vec{Y}_n + \frac{\vec{K}_2}{2}\right) & \quad \quad \quad & \xi_3 = \frac{1}{2} \\
 \vec{K}_4 &= h \vec{f}(t_n + h, \vec{Y}_n + \vec{K}_3) & \quad \quad \quad & \xi_4 = 1 \\
 \vec{Y}_{n+1} &= \vec{Y}_n + \frac{1}{6} \vec{K}_1 + \frac{2}{6} \vec{K}_2 + \frac{2}{6} \vec{K}_3 + \frac{1}{6} \vec{K}_4
 \end{aligned} \tag{15}$$

3.5 - Stabilité de la méthode de Runge Kutta d'ordre 4 (RK44)

Nous pouvons conduire une étude similaire en reprenant le schéma numérique défini aux équations (15) appliqué au système :

$$\frac{\partial \vec{Y}}{\partial t} = [A] \vec{Y} + \vec{C}$$

en posant $\vec{Y}_n = \vec{Y}_n^* + \vec{\varepsilon}_n$ et en reportant dans les équations (15) nous obtenons :

$$\vec{\varepsilon}_{n+1} = \left\{ [I] + h[A] + \frac{h^2}{2} [A]^2 + \frac{h^3}{3!} [A]^3 + \frac{h^4}{4!} [A]^4 \right\} \vec{\varepsilon}_n$$

si λ_i est une valeur propre de $[A]$ alors :

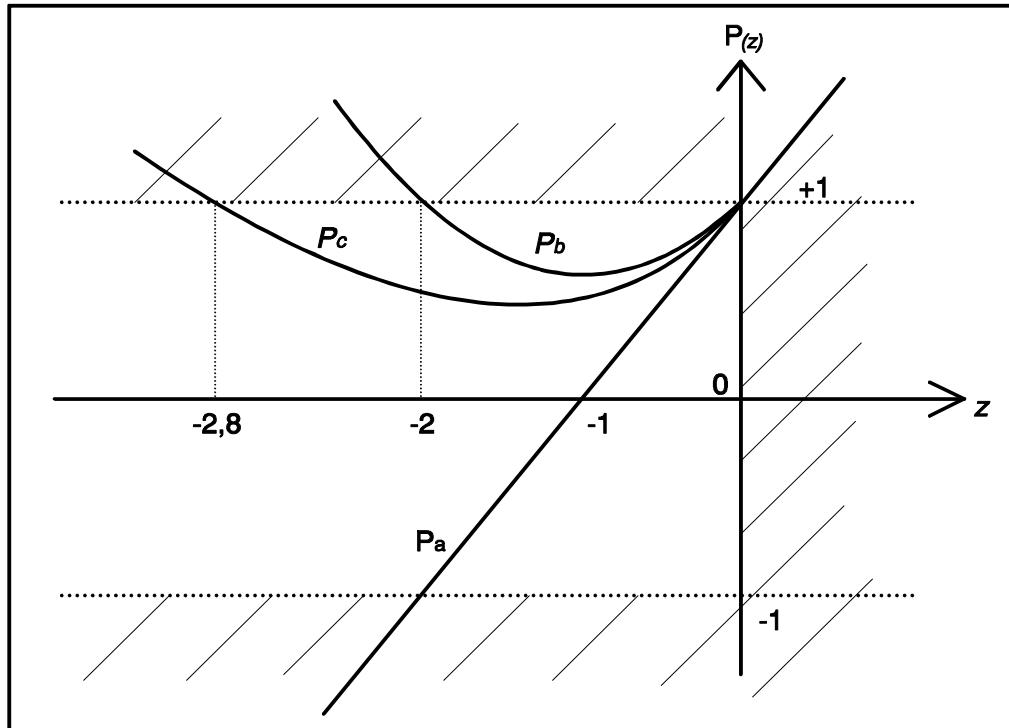
$1 + h\lambda_i + \frac{h^2}{2} \lambda_i^2 + \frac{h^3}{3!} \lambda_i^3 + \frac{h^4}{4!} \lambda_i^4$ est une valeur propre de la matrice de propagation des erreurs.

La condition de stabilité va donc s'exprimer :

$$|P(z)| < 1 \quad \text{avec} \quad P(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{3!} + \frac{z^4}{4!}$$

où $z = h\lambda_i$

Nous obtenons graphiquement :



Sur ce graphique nous avons porté les polynômes

$P_a = 1 + z$ (cas de la méthode d'Euler)

$P_b = 1 + z + \frac{z^2}{2}$ (cas de la méthode de la tangente améliorée)

$P_c = 1 + z + \frac{z^2}{2} + \frac{z^3}{3!} + \frac{z^4}{4!}$ (cas de la méthode de Runge Kutta d'ordre 4).

Définition :

On appelle rayon de stabilité d'une méthode la valeur ρ , borne inférieure de l'intervalle $[-\rho, 0]$ dans lequel le polynôme $P(z)$ demeure entre -1 et +1.

En conséquence dès que l'on connaît ρ le rayon de stabilité d'une méthode et λ la plus grande valeur propre du problème traité, on en déduit la condition de stabilité sur le pas :

$$h < \frac{\rho}{\lambda}$$

Résumé :

Les rayons de stabilité ρ des méthodes usuelles sont :

2	méthode d'Euler
2	méthode de la tangente améliorée
2,8	méthode de Runge Kutta

Remarque :

La détermination des valeurs propres d'un problème différentiel est souvent très délicate voire impossible. Des méthodes permettent de déterminer automatiquement le pas optimal.

4 - LES METHODES A PAS LIES :

Les méthodes de RUNGE - KUTTA d'ordre 3 ou 4 donnent de très bons résultats numériques mais présentent l'inconvénient de nécessiter beaucoup de calculs. Par exemple, avec la méthode d'ordre 4, il faut 4 évaluations de f pour obtenir Y_{n+1} à partir de Y_n . On dit que ces méthodes sont à pas séparés car le calcul de Y_{n+1} se fait seulement en utilisant Y_n . Dès que l'on veut avoir des méthodes à pas séparés d'ordre élevé on est obligé de faire beaucoup d'évaluations de f . Pour remédier à cet inconvénient, on utilise des méthodes où le calcul de Y_{n+1} fera intervenir non seulement Y_n mais aussi Y_{n-1}, \dots, Y_{n-k} (k fixé) : ce sont les méthodes à pas liés.

On peut les décrire sous la forme générale :

$$Y_{n+1} = \alpha_1 Y_n + \alpha_2 Y_{n-1} + \dots + \alpha_r Y_{n-r+1} \\ + \beta_0 f(Y_{n+1}, t_{n+1}) + \beta_1 f(Y_n, t_n) + \beta_2 f(Y_{n-1}, t_{n-1}) \\ + \dots + \beta_r f(Y_{n-r+1}, t_{n-r+1})$$

Si $\beta_0 = 0$ ou si f est linéaire : **formules explicites**

Si $\beta_0 \neq 0$ et si f est non linéaire : **formules implicites**

4.1 - Principe pour trouver les coefficients :

On exprime que la relation générale doit être vérifiée si Y est un polynôme de degré fixé.

On exprime cette relation sur la base canonique : $1, t, t^2, \dots$

d'où le système linéaire :

$$\begin{cases} 1 = \alpha_1 + \alpha_2 + \dots + \alpha_r \\ t = (t - \Delta t) \alpha_1 + (t - 2\Delta t) \alpha_2 + \dots + \beta_0 + \beta_1 + \dots \\ \vdots \\ t^s = (t - \Delta t)^s \alpha_1 + (t - 2\Delta t)^s \alpha_2 + \dots + \beta_0 t^{s-1} s + \beta_r (t - r\Delta t) t^{s-1} s \\ \vdots \end{cases}$$

qui fournit les valeurs des $\alpha_1 \dots \alpha_r$ cherchés.

Exemples d'applications :

On se limite à des ordres relativement faibles : $s \approx 1$ à 4

Nous allons noter : $f_i = f(t_i, Y_i)$

On a ainsi par exemple : avec $\alpha_1 = 1$, $\beta_0 = 0$, $\beta_1 = 3h/2$, $\beta_2 = -h/2$

la formule : $Y_{n+1} = Y_n + h(3f_n - f_{n-1})/2$

On voit que pour l'utiliser il faut calculer f en (t_{n-1}, Y_{n-1}) et en (t_n, Y_n) . On ne peut donc se servir de la relation précédente que pour $n = 1, 2, \dots$; mais il faut connaître Y_0 (on connaît : condition initiale) et Y_1 approximation de la solution en t_1 . Il faut donc auparavant utiliser une méthode à pas séparés pour calculer Y_1 .

Les méthodes à pas liés ne peuvent pas démarrer toutes seules : il faut au départ utiliser une méthode à pas séparés.

On démontre que la méthode à pas liés précédente est d'ordre 2 à condition que Y_1 ait été calculé par une méthode à pas séparés également d'ordre 2.

Autre exemple, ordre 4, ($n = 3, 4 \dots$) :

$$Y_{n+1} = Y_n + h(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3})/24$$

Les méthodes à pas liés fonctionnent souvent avec un pas constant h .

4.2 - Prédiction - Correction :

A chaque pas deux méthodes d'intégration sont utilisées successivement :

- * Une méthode explicite qui fournira une bonne valeur de départ, qui prédira Y_{n+1} (on l'appelle le prédicteur).
- * Une méthode implicite avec laquelle nous ferons seulement une itération de la méthode des approximations successives, on corrigera la première valeur de Y_{n+1} (On l'appellera le correcteur).

L'ordre du correcteur est d'une unité supérieur à celui du prédicteur si bien que la différence entre les deux résultats permet de contrôler l'erreur sans avoir à utiliser une autre méthode entièrement indépendante et donc d'en déduire le pas optimal.

Exemple de prédiction correction :

$$Y_{n+1}^* = Y_n + h (23f_n - 16f_{n-1} + 5f_{n-2}) / 12 \quad \text{prédicteur}$$

$$Y_{n+1} = Y_n + h (9f(t_{n+1}, Y_{n+1}^*) + 19f_n - 5f_{n-1} + f_{n-2}) / 24 \quad \text{correcteur}$$

Le prédicteur est d'ordre 3 et le correcteur d'ordre 4. On utilisera la méthode de RUNGE-KUTTA d'ordre 4 pour le démarrage.

Pour ces méthodes il se pose aussi le problème de la stabilité que l'on peut étudier avec les mêmes outils théoriques que ceux mis en oeuvre pour les méthodes à pas séparés.

BIBLIOGRAPHIE

- Hacques G., *Algorithmique numérique*, cours Dunod Université.
- Acton F.S., *Numerical Methods that Work*, New-York, Harper & Row (1970), ch. 5.
- Gear W.C., *Numerical Initial Value Problems in Ordinary Differential Equations*, Englewood Cliffs, N.J. : Prentice Hall (1971).
- Shampine L.F. and Gordon M.K., *Computer Solution of Ordinary Differential Equations: the Initial Value Problem*, San Francisco, W. H. Freeman (1975).
- Stoer J. and Bulirsh R., *Introduction to Numerical Analysis*, New-York, Springer Verlag (1980), ch. 7.
- Rice J.R., *Numerical Methods, Software and Analysis*, New-York, McGraw-Hill (1983), §9.2.
- Press W., Flannery B., Teukolsky S. and Vetterling W., *Numerical Recipes : the Art of Scientific Computing*, Cambridge University Press (1986).

Méthode de Gauss

Intégration numérique à une dimension par la méthode de Gauss-Legendre

La méthode de Gauss est une technique d'intégration numérique très utilisée dans laquelle les r coefficients w_i et les abscisses ξ_i sont déterminés de manière à intégrer exactement des polynômes d'ordre $m \leq 2r - 1$.

Remplaçons l'intégrale d'une fonction polynomiale $y(\xi)$ par une combinaison linéaire de ses valeurs aux points d'intégration ξ_i :

$$\int_{-1}^1 y(\xi) d\xi = \sum_{i=1}^r w_i \cdot y(\xi_i) \quad (1)$$

Déterminons les $2r$ coefficients de manière à ce que (1) soit vérifiée exactement pour le polynôme suivant:

$$y(\xi) = \sum_{i=1}^{2r} a_i \cdot \xi^{i-1} \quad (2)$$

Portons cette expression dans (1):

$$\sum_{i=1}^{2r} a_i \cdot \int_{-1}^1 \xi^{i-1} \cdot d\xi = \sum_{i=1}^{2r} a_i \cdot \sum_{j=1}^r w_j \cdot \xi_j^{i-1} \quad (3)$$

Pour que (3) soit identiquement vérifiée pour tout a_i , il faut:

$$\int_{-1}^1 \xi^\alpha \cdot d\xi = \frac{2}{\alpha + 1} = \sum_{i=1}^r w_i \cdot \xi_i^\alpha \quad \text{pour } \alpha \text{ pair} \quad (4)$$

et

$$\int_{-1}^1 \xi^\alpha \cdot d\xi = 0 = \sum_{i=1}^r w_i \cdot \xi_i^\alpha \quad \text{pour } \alpha \text{ impair} \quad (5)$$

soit

$$2 = \sum_{i=1}^r w_i \quad 0 = \sum_{i=1}^r w_i \cdot \xi_i \quad \frac{2}{3} = \sum_{i=1}^r w_i \cdot \xi_i^2 \quad \dots \quad 0 = \sum_{i=1}^r w_i \cdot \xi_i^{2r-1} \quad (6)$$

Ce système de $2r$ équations est linéaire en w_i et non linéaires en ξ_i ; il détermine les $2r$ paramètres de (1) respectant les conditions $w_i > 0$ et $-1 < \xi_i < 1$.

Exemple: calcul des coefficients de la méthode de Gauss à deux points.

Dans ce cas $r=2$, l'expression (1) devient: $\int_{-1}^1 y(\xi) d\xi = w_1 \cdot y(\xi_1) + w_2 \cdot y(\xi_2)$

Pour que cette approximation soit exacte pour un polynôme de degré $2r-1=3$, il faut que les relations (6) soient satisfaites:

$$\begin{cases} 2 = w_1 + w_2 & 0 = w_1 \cdot \xi_1 + w_2 \cdot \xi_2 \\ \frac{2}{3} = w_1 \cdot \xi_1^2 + w_2 \cdot \xi_2^2 & 0 = w_1 \cdot \xi_1^3 + w_2 \cdot \xi_2^3 \end{cases} \quad (7)$$

La solution de ce système est:

$$w_1 = w_2 = 1 \quad \text{et} \quad \xi_1 = -\xi_2 = \frac{1}{\sqrt{3}} \quad (8)$$

Les abscisses ξ_i , solutions de (4) et (5) sont aussi les racines du polynôme de Legendre d'ordre r , $P_r(\xi) = 0$, défini par la formule de récurrence:

$$P_0(\xi) = 1 \quad P_1(\xi) = \xi \quad \cdot \quad P_r(\xi) = \frac{2r-1}{r} \xi \cdot P_{r-1}(\xi) - \frac{r-1}{r} \cdot P_{r-2}(\xi) \quad (9)$$

Les poids d'intégration s'écrivent $w_i = \frac{2 \cdot (1 - \xi_i^2)}{[r \cdot P_{r-1}(\xi_i)]^2}$.

L'erreur d'intégration est de la forme $e = \frac{2^{2r+1} \cdot (r!)^4}{(2r+1) \cdot [(2r)!]^3} \cdot \frac{d^{2r}y}{d\xi^{2r}}$.

r	ξ_i	w_i	Erreur e	Intégration exacte des polynômes de degré m
1	0.	2.	$\frac{1}{6} \frac{d^2y}{d\xi^2}$	1
2	$\pm \frac{1}{\sqrt{3}}$	1.	$\approx 0.7 \cdot 10^{-2} \frac{d^4y}{d\xi^4}$	3
3	0. $\pm \sqrt{\frac{3}{5}}$	8./9. 5./9.	$\approx 0.6 \cdot 10^{-4} \frac{d^6y}{d\xi^6}$	5
4	$\pm \sqrt{\frac{3-2\sqrt{6/5}}{7}}$ $\pm \sqrt{\frac{3+2\sqrt{6/5}}{7}}$	$\frac{1}{2} + \frac{1}{6\sqrt{6/5}}$ $\frac{1}{2} - \frac{1}{6\sqrt{6/5}}$	$\approx 0.3 \cdot 10^{-6} \frac{d^8y}{d\xi^8}$	7
5	0. $\pm \frac{1}{3} \sqrt{5-4\sqrt{5/14}}$ $\pm \frac{1}{3} \sqrt{5+4\sqrt{5/14}}$	128./225. $\frac{161}{450} + \frac{13}{180\sqrt{5/14}}$ $\frac{161}{450} - \frac{13}{180\sqrt{5/14}}$	$\approx 0.8 \cdot 10^{-9} \frac{d^{10}y}{d\xi^{10}}$	9
6	$\pm 0.23861 \ 91861$ $\pm 0.66120 \ 93865$ $\pm 0.93246 \ 95142$	0.46791 39346 0.36076 15730 0.17132 44923	$\approx 1.5 \cdot 10^{-12} \frac{d^{12}y}{d\xi^{12}}$	11
7	0. $\pm 0.40584 \ 51514$ $\pm 0.74153 \ 11856$ $\pm 0.94910 \ 79123$	0.41795 91837 0.38183 00505 0.27970 53915 0.12948 49662	$\approx 2.1 \cdot 10^{-15} \frac{d^{14}y}{d\xi^{14}}$	13

Résultat d'une simulation

