REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

Mémoire

Présentée à

L'Université des Sciences et Technologies Houari Boumediene-Alger-

Département des Mathématiques

Pour obtenir le grade de

MAGISTER

Option : Analyse et approximation des équations aux dérivées partielles

Par
Massoun Youssouf

Thème :

La méthode d'analyse homotopique Des équations différentielles non linéaires

Soutenue le 06/10/2015 devant la commission d'examen

Membres du jury :

M. Mohamed Said Moulay	**Professeur, USTHB- Alger**	**Président**
M. Abdelhamid benmezai	**Professeur, USTHB-Alger**	**Examinateur**
M. Rachid Benzine	**Professeur, Université d' Annaba**	**Examinateur**
M. Mohamed Elamine Talbi	**M.C.B , Université de Blida**	**invité**
M. Djilali Benayat	**Professeur, ENS-Kouba**	**Directeur de mémoir**

بسم الله الرحمن الرحيم

رب صل على روح الوجود
الآية الكبرى مرآة النهض
والآل والأصحاب بدور الكمال
والأئمة الكرام خواص البشر

A mes Pères

Sidi Mohamed et *Sidi Abdellatif*

Remerciements :

Je remercie mon directeur de thèse, Monsieur **Djilali Benayat**, qui m'a transmis sa motivation, nécessaire pour mener à bien ce travail, pour son soutien sans faille, sa disponibilité, (sa porte toujours ouverte me permettant de le déranger a tout instant), pour toute l'aide et les conseils qu'il a su m'apporter pendant la préparation de cette thèse. Grace à lui, ce fut un plaisir de faire des mathématiques et je lui en exprime toute ma reconnaissance.

Je remercie sincèrement le professeur **Mohamed Said Moulay** pour l'honneur qu'il me fait en présidant ce jury. Je remercie vivement le professeur ***Abdelhamid Benmezai*** le professeur ***Rachid Benzine*** , ***Mohamed El-Amine Talbi***. pour leur acceptation à participer au jury dévaluation de cette thèse .

Mes sincères salutaion au responsable du laboratoire de recherche (EDP) en l'occurrence monsieur **Kessab Amour** sans oublier les autres professeurs.

Grand merci à mon Père, ma Mère, mes frères, ma sœur. Hayat, Alaa , Mohamed Rayane , Mohamed El-Amine et Abdellatif.

Je ne pourrais clôturer ces remerciements sans me retourner vers mon ainé Dr **Nourredinne Belkaid**, Dr **Abderrahmane Belkaid**, Dr **Abdessallame Belkaid** et *Abderrezak Said* et toute l'équipe (…) et (*Mohamed, Lahcen* et *El-Hadj, Youcef Belkaid ,Dr.Ougaida, R.chérifi, Hamrat, Bouzid , Makhlouf, Tighza ,…*) qui ont tout fait pour que cette thèse voit le jour.

Vous tous qui m'avez toujours soutenu et poussé en avant, cette thèse, je suis fier de vous l'offrir. Merci.

Table des matières

Chapitre 1

Introduction

Les équations non linéaires, de tout type, sont beaucoup plus difficiles à résoudre analytiquement que les équations linéaires. Les deux principaux critères de satisfaction d'une méthode analytique sont :

1. La méthode peut **toujours** donner une approximation analytique de la solution, de "**manière efficace**".
2. La méthode garantit que l'approximation est "**suffisamment correcte**" pour toutes les valeurs des paramètres physiques de l'équation.

Citons, parmi les principales méthodes analytiques de résolutions des EDP et ODE :

1. Les techniques de perturbation

   Elles sont très largement utilisées, surtout en technologie et en sciences appliquées. Elles sont basées sur des développements en séries entières en des (petits ou grands) paramètres de perturbations apparaissant dans l'équation ou bien dans les conditions initiales ou aux bords. Le point faible des techniques de perturbation est que la majorité des équations non-linéaires n'ont pas de paramètres de perturbation.
2. La méthode du petit paramètre artificiel de Lyapounov [2]
3. La méthode de la δ-expansion [2]
4. La méthode de décomposition de Adomian [2]

En général, ces méthodes satisfont bien le premier critère, mais pas le second car elles n'assurent pas la convergence des séries obtenues.

La méthode d'homotopie de Shijun Liao satisfait aux deux critères et est particulièrement adaptée aux problèmes hautement non-linéaires. Très succinctement, elle consiste

à plonger l'équation E dans une famille d'équations $(E_q)_{0\leq q\leq 1}$ où l'on a $E_1 = E$ et E_0 est une solution approchée de départ. La solution $\Phi(\tau, q)$ de l'équation E_q est développée en série entière en le paramètre q; pour $q = 1$, $\Phi(\tau, 1)$ doit être une solution de $E_1 = E$, c'est à dire de l'équation initiale.

Le plan de ce travail est le suivant :

1. Résumé de la méthode HAM
2. Détails de la méthode HAM
   (a) Propriétés de la dérivation homotopique
   (b) Les équations de déformation d'ordre supérieur
   (c) Exemples
   (d) Théorèmes de convergence
   (e) Expression de la solution
      i. Choix de l'approximation initiale
      ii. Choix de l'opérateur linéaire auxiliaire
3. Programme Mathematica BVPh2.0 et exemples pratiques
4. Applications de HAM à plusieurs types d'équations

Chapitre 2

Résumé de HAM

La notion d'homotopie est synonyme de "déformation continue" ; c'est un concept purement topologique. Soient f et g deux applications continues entre les espaces topologiques X et Y; elles sont dites homotopes s'il existe une application continue $F : X \times [0,1] \longrightarrow Y$ telle que $F(x,0) = f(x)$ et $F(x,1) = g(x), \forall x \in X$. On peut remplacer l'intervalle $[0,1]$ par un intervalle $[a,b]$, $a \neq b$, car ils sont tous homéomorphes entre eux. On regarde généralement F comme la famille continue de déformations $(f_t = F(.,t) : X \longrightarrow Y)_{0 \leq t \leq 1}$, qui démarre en $f_0 = f$ et aboutit en $f_1 = g$.

Soit

$$N(u(\tau)) = 0 \tag{2.1}$$

une EDP non-linéaire où $u(\tau)$ est la fonction inconnue de la variable τ et N un opérateur différentiel non-linéaire. La première étape de HAM est de construire l'équation de déformation d'ordre 0 :

$$(1-q) L[\Phi(\tau,q) - u_0(\tau)] = cqH(\tau) N[\Phi(\tau,q)] \tag{2.2}$$

où $q \in [0,1]$ est le paramètre de déformation appelé paramètre de plongement, $c \neq 0$ est un paramètre auxiliaire qui permettra d'avoir un contrôle sur la convergence des séries, L est un opérateur linéaire vérifiant $L(u) = 0 \Longrightarrow u = 0$, $H(\tau)$ est une fonction auxiliaire qui ne s'annule pas, et enfin, $u_0(\tau)$ est une solution approchée de l'EDP. En pratique, on a $\tau = (x,t)$ où x est une variable d'espace et t le temps. L'auteur de cette méthode l'a qualifié d'homotopie car, pour le paramètre q variant de 0 à 1, on a toute une famille d'équations, indexée par q, démarrant , pour $q = 0$, par l'équation $L[\Phi(\tau,0) - u_0(\tau)] = 0$

ayant pour solution $\Phi(\tau,0)=u_0(\tau)$, et aboutissant, pour $q=1$, en l'équation initiale $N[\Phi(\tau,1)]=0$. HAM utilise le dévelopement de $\Phi(\tau,q)$ en série entière en le paramètre d'homotopie q :

$$\Phi(\tau,q)=u_0(\tau)+\sum_{m=1}^{\infty}u_m(\tau)\,q^m \tag{2.3}$$

où $u_m(\tau)=\left[\frac{1}{m!}\frac{\partial^m\Phi(\tau,q)}{\partial q^m}\right]_{q=0}$. Si le choix de L, de $u_0(\tau)$, de c et de la fonction auxiliaire sont convenables, alors la série (2.3) est convergente en $q=1$, donnant $u(\tau)=u_0(\tau)+\sum_{m=1}^{\infty}u_m(\tau)$ qui doit être une solution de l'équation initiale (2.1).

Remark 2.0.1 *1. En toute rigueur, qualifier cette méthode "d'homotopie" est un abus de langage car, dans le présent contexte, on ne dit pas ce que signifie "la continuité" de la famille*

$((1-q)\,L[\Phi(\tau,q)-u_0(\tau)]=cqH(\tau)\,N[\Phi(\tau,q)])_{q\in[0,1]}$.

2. Si l'équation initiale a plusieurs solutions, HAM possède la propriété de pouvoir les détecter et donner leur expression analytique [Abbasbandy et al.[5]].

Chapitre 3

Détails de la méthode HAM

3.1 Propriétés de la dérivation homotopique :

La derivée d'homotopie est utilisé dans le cadre de HAM.

Ici, nous donnons d'abord une définition rigoureuse de la dérivée d'homotopie puis prouvons certaines de ses propriétés en génèral. Grace à ces propriétés, il est facile de déduire les équations de déformation d'ordre supérieur correspondantes de toute équation de déformation d'ordre zéro donnée .[1]

Notations :

Dans tout ce qui suit, nous utiliserons les notations suivantes :

L un opérateur linéaire auxiliaire qui a la propriété $L(0) = 0$ et indépendante du paramètre d'homotopie $q \in [0, 1]$.

N un opérateur non linéaire.

$u_0(x, t)$ la valeur initiale de l'équation original $N(u) = 0$.

c le paramètre de contrôle de la convergence indépendant de q.

$H(x, t)$ une fonction auxiliaire indépendant de q.

x représente un vecteur de variable spatiale.

t est la variable indépendante temporelle.

Définition 3.1.1 *Soit ϕ une fonction du paramètre d'homotopie q, alors :*

$$D_m(\phi) = \frac{1}{m!}\left.\frac{d^m\phi}{dq^m}\right|_{q=0} \tag{3.1}$$

est appelé la dérivée d'homotopie d'ordre supérieur de ϕ , où $m \geq 0$ est un entier, et D_m est appelé l'opérateur de dérivée d'homotopie d'ordre m.

Théorème 3.1.1 *Pour deux séries d'homotopie de Maclaurin arbitraires*

$$\phi = \sum_{k=0}^{+\infty} u_k q^k \qquad\qquad \psi = \sum_{k=0}^{+\infty} w_k q^k,$$

où ϕ et ψ sont analytiques en $q \in [0; a]$, on a :

$$\begin{aligned}
D_m(\phi) &= u_m & (3.2)\\
D_m(q^k\phi) &= D_{m-k}(\phi) = \begin{cases} u_{m-k} & si \quad 0 \leq k \leq m, \\ 0 & si\ non\ . \end{cases} & (3.3)\\
D_m(\phi\psi) &= \sum_{k=0}^{m} D_k(\phi) D_{m-k}(\psi) = \sum_{k=0}^{m} u_k w_{m-k} & (3.4)\\
&= \sum_{k=0}^{m} D_{m-k}(\phi) D_k(\psi) = \sum_{k=0}^{m} u_{m-k} w_k & (3.5)\\
D_m(\phi^{n+1}) &= \sum_{k=0}^{m} D_k(\phi) D_{m-k}(\phi^n) = \sum_{k=0}^{m} D_{m-k}(\phi) D_k(\phi^n) & (3.6)
\end{aligned}$$

où $m \geq 0, n \geq 1$ et $0 \leq k \leq m$ sont des entiers.

Preuve.

- *D'après le théorème de Taylor (Fitzpatrick, 1996), le coefficient unique u_m de la série de Maclaurin $\phi = \sum_{k=0}^{+\infty} u_k q^k$ par rapport au paramètre d'homotopie q est donné par :*

$$u_m = \frac{1}{m!}\left.\frac{\partial^m\phi}{\partial q^m}\right|_{q=0},$$

  ce qui donne (3,2) à l'aide de la définition (3,1) de $D_m(\phi)$.

- *On a :*

$$q^k\phi = q^k \sum_{j=0}^{+\infty} u_j q^j = \sum_{j=0}^{+\infty} u_j q^{j+k} = \sum_{m=k}^{+\infty} u_{m-k} q^m,$$

ce qui donne à l'aide de (3,2) :

$$D_m\left(q^k\phi\right) = u_{m-k} = D_{m-k}\left(\phi\right), \qquad si\ 0 \leq k \leq m,$$

et

$$D_m\left(q^k\phi\right) = 0 \qquad si\ k > m.$$

– *D'après la règle de Leibniz pour les dérivées de produits, on obtient :*

$$\frac{\partial^m \phi\psi}{\partial q^m} = \sum_{k=0}^{m} \frac{m!}{k!(m-k)!} \frac{\partial^k \phi}{\partial q^k} \frac{\partial^{m-k} \psi}{\partial q^{m-k}} = \sum_{k=0}^{m} \frac{m!}{k!(m-k)!} \frac{\partial^k \psi}{\partial q^k} \frac{\partial^{m-k} \phi}{\partial q^{m-k}},$$

ce qui donne selon (3,1) et (3,2) :

$$\begin{aligned} D_m\left(\phi\psi\right) &= \frac{1}{m!} \frac{\partial^m \phi\psi}{\partial q^m}\bigg|_{q=0} = \sum_{k=0}^{m} \left(\frac{1}{k!} \frac{\partial^k \phi}{\partial q^k}\bigg|_{q=0}\right) \left[\frac{1}{(m-k)!} \frac{\partial^{m-k} \psi}{\partial q^{m-k}}\bigg|_{q=0}\right] \\ &= \sum_{k=0}^{m} D_k\left(\phi\right) D_{m-k}\left(\psi\right) = \sum_{k=0}^{m} u_k w_{m-k}. \end{aligned}$$

De même :

$$D_m\left(\phi\psi\right) = \sum_{k=0}^{m} D_k\left(\psi\right) D_{m-k}\left(\phi\right) = \sum_{k=0}^{m} u_{m-k} w_k.$$

– *Posons $\psi = \phi^n$ .D'après (3,5), on a :*

$$D_m\left(\phi^{n+1}\right) = D_m\left(\phi\phi^n\right) = \sum_{k=0}^{m} D_k\left(\phi\right) D_{m-k}\left(\phi^n\right).$$

De même :

$$D_m\left(\phi^{n+1}\right) = D_m\left(\phi\phi^n\right) = \sum_{k=0}^{m} D_k\left(\phi^n\right) D_{m-k}\left(\phi\right).$$

■

Théorème 3.1.2 *Si $\phi = \sum_{k=0}^{+\infty} u_k q^k$ et $\psi = \sum_{k=0}^{+\infty} w_k q^k$ sont deux séries d'homotopie de Maclaurin , où ϕ et ψ sont analytiques pour $q \in [0,1]$, f et g sont indépendantes du paramètre d'homotopie $q \in [0,1]$.*
Alors :

$$D_m(f\phi + \psi g) = fD_m(\phi) + gD_m(\psi) = fu_m + gw_m \tag{3.7}$$

Preuve. *Comme l'opérateur D_m défini par (3,1) est linéaire, et f et g sont indépendantes de q, on obtient :*

$$D_m\left(f\phi+\psi g\right)=fD_m\left(\phi\right)+gD_m\left(\psi\right).$$

D'après (3,2), on a : $D_m\left(\phi\right)=u_m$ et $D_m\left(\psi\right)=w_m$, donc

$$D_m\left(f\phi+\psi g\right)=fD_m\left(\phi\right)+gD_m\left(\psi\right)=fu_m+gu_m.$$

■

Théorème 3.1.3 *Pour deux séries d'homotopie de Maclaurin :*

$$\phi=\sum_{k=0}^{+\infty}u_kq^k \qquad\qquad \psi=\sum_{k=0}^{+\infty}w_kq^k$$

où ϕ et ψ sont analytiques pour tout $q\in[0,a]$.
Alors :

$$D_m[L(\phi)]=L[D_m(\phi)]=L(u_m), \tag{3.8}$$

et

$$D_m[\psi L(\phi)]=\sum_{n=0}^{m}D_{m-n}(\psi)L[D_n(\phi)]=\sum_{n=0}^{m}w_{m-n}L(u_n) \tag{3.9}$$

où $m\geq 0$ est un entiers.

Preuve.
Puisque L est linéaire et indépendant de q, en utilisant le théorème 3,1, on obtient :

$$L\left(\phi\right)=L\left(\sum_{k=0}^{+\infty}u_kq^k\right)=\sum_{k=0}^{+\infty}L\left(u_k\right)q^k.$$

En utilisant la formule (3,2), on a : $D_m\left[L\left(\phi\right)\right]=L\left(u_m\right).$
D'un autre côté, selon (3,2), on obtient : $L\left[D_m\left(\phi\right)\right]=L\left(u_m\right).$
Ainsi :

$$D_m\left[L\left(\phi\right)\right]=L\left[D_m\left(\phi\right)\right]=L\left(u_m\right).$$

Ensuite, d'après le théorème 1, on a :

$$\begin{aligned} D_m\left[\psi L\left(\phi\right)\right] &= \sum_{n=0}^{m} D_{m-n}\left(\psi\right) D_n\left[L\left(\phi\right)\right] \\ &= \sum_{n=0}^{m} D_{m-n}\left(\psi\right) L\left[D_n\left(\phi\right)\right] = \sum_{n=0}^{m} w_{m-n} L\left(u_n\right). \end{aligned}$$

■

Théorème 3.1.4 *Soient les séries d'homotopie de Maclaurin :*

$$\phi = \sum_{i=0}^{+\infty} u_i q^i \qquad\qquad \psi = \sum_{j=0}^{+\infty} w_j q^j$$

où ϕ et ψ sont analytiques pour tout $q \in [0, a]$, si $\phi = \psi$ pour tout $q \in [0, a]$. Alors :

$$u_m = w_m \qquad et \qquad D_m(\phi) = D_m(\psi)$$

pour tout entier $m \geq 0$ et réel $a > 0$.

Preuve. *Puisque $\phi = \psi$, on a :*

$$\sum_{k=0}^{+\infty} \left(u_k - w_k\right) q^k = 0.$$

L'expression ci-dessus est vérifiée $\forall q \in [0, a]$, ssi :

$$u_m = w_m, \qquad m \geq 0,$$

ce qui donne, vu (3,2),

$$D_m\left(\phi\right) = D_m\left(\psi\right).$$

■

Théorème 3.1.5 *Soient f et g deux fonctions réelles C^∞, et*

$$\phi = \sum_{i=0}^{+\infty} u_i q^i \qquad et \qquad \psi = \sum_{j=0}^{+\infty} w_j q^j$$

deux séries d'homotopie de Maclaurin.

Si $f(\phi) = g(\psi)$ pour tout $q \in [0, a]$ alors :

$$D_m[f(\phi)] = D_m[g(\psi)], \qquad \forall m \geq 0, \ \forall a > 0.$$

Preuve.

Posons :

$$\Phi = f(\phi), \quad \Psi = g(\psi)$$

Selon le théorème 4, on a :

$$D_m(\Phi) = D_m(\Psi)$$

ce qui donne :

$$D_m[f(\phi)] = D_m[g(\psi)].$$

■

Théorème 3.1.6 *Pour une série d'homotopie de Maclaurin arbitraire $\phi = \sum_{k=0}^{+\infty} u_k q^k$;on a :*

$$D_m(\phi^2) = \sum_{n=0}^{m} u_{m-n} u_n, \tag{3.10}$$

$$D_m(\phi^3) = \sum_{n=0}^{m} u_{m-n} \sum_{k=0}^{n} u_{n-k} u_k, \tag{3.11}$$

$$D_m(\phi^4) = \sum_{n=0}^{m} u_{m-n} \sum_{k=0}^{n} u_{n-k} \sum_{j=0}^{k} u_{k-j} u_j, \tag{3.12}$$

$$D_m(\phi^5) = \sum_{n=0}^{m} u_{m-n} \sum_{k=0}^{n} u_{n-k} \sum_{j=0}^{k} u_{k-j} \sum_{i=0}^{j} u_{j-i} u_i, \tag{3.13}$$

$$D_m(\phi^\sigma) = \sum_{r_1=0}^{m} u_{m-r_1} \sum_{r_2=0}^{r_1} u_{r_1-r_2} \sum_{r_3=0}^{r_2} u_{r_2-r_3} \cdots \sum_{r_{\sigma-1}=0}^{r_{\sigma-2}} u_{r_{\sigma-2}-r_{\sigma-1}} u_{r_{\sigma-1}}, \tag{3.14}$$

où $m \geq 0$ et $\sigma \geq 2$ sont des entiers positifs.

Preuve.

– *Selon (3.2) et (3.4), on a :*

$$D_m\left(\phi^2\right) = \sum_{n=0}^{m} D_{m-n}(\phi) D_n(\phi) = \sum_{n=0}^{m} u_{m-n} u_n.$$

– *et :*

$$D_n\left(\phi^2\right) = \sum_{k=0}^{n} u_{n-k}u_k,$$

de même :

$$D_m\left(\phi^3\right) = D_m\left(\phi\phi^2\right) = \sum_{n=0}^{m} D_{m-n}\left(\phi\right) D_n\left(\phi^2\right) = \sum_{n=0}^{m} u_{m-n} \sum_{k=0}^{n} u_{n-k}u_k.$$

– *D'après (3.11) on a :*

$$D_n\left(\phi^3\right) = \sum_{k=0}^{n} u_{n-k} \sum_{j=0}^{k} u_{k-j}u_j,$$

et selon (3.4) on a :

$$D_m\left(\phi^4\right) = \sum_{n=0}^{m} D_{m-n}\left(\phi\right) D_n\left(\phi^3\right) = \sum_{n=0}^{m} u_{m-n} \sum_{k=0}^{n} u_{n-k} \sum_{j=0}^{k} u_{k-j}u_j.$$

– *Selon (3.12) on a :*

$$D_n\left(\phi^4\right) = \sum_{k=0}^{n} u_{n-k} \sum_{j=0}^{k} u_{k-j} \sum_{i=0}^{j} u_{j-i}u_i,$$

et selon (3.2) on a :

$$D_m\left(\phi^5\right) = \sum_{n=0}^{m} D_{m-n}\left(\phi\right) D_n\left(\phi^5\right) = \sum_{n=0}^{m} u_{m-n} \sum_{k=0}^{n} u_{n-k} \sum_{j=0}^{k} u_{k-j} \sum_{i=0}^{j} u_{j-i}u_i$$

etc....

■

Théorème 3.1.7 *Pour toute série d'homotopie de Maclaurin :*

$$\phi = \sum_{k=0}^{+\infty} u_k q^k$$

on a :

$$\begin{aligned} D_0(e^{\alpha\phi}) &= e^{\alpha u_0}, \\ D_m(e^{\alpha\phi}) &= \alpha \sum_{k=0}^{m-n} \left(1-\tfrac{k}{m}\right) D_{m-k}(\phi) D_k(e^{\alpha\phi}) = \alpha \sum_{k=0}^{m-n} \left(1-\tfrac{k}{m}\right) u_{m-k} D_k(e^{\alpha\phi}). \end{aligned}$$

où m est un entier positif, et $\alpha \neq 0$ est indépendant du paramètre d'homotopie q.

Preuve.

– *D'après la définition (3.1) de l'opérateur D_m , on obtient évidemment :*

$$D_0\left(e^{\alpha u}\right) = e^{\alpha u_0},$$

En outre puisque α est indépendant de q on a :

$$\frac{\partial e^{\alpha u}}{\partial q} = \alpha e^{\alpha u} \frac{\partial u}{\partial q},$$

Ainsi, selon la règle de ***Leibniz*** *pour les dérivées de produits, on a :*

$$\begin{aligned} \tfrac{1}{m!}\tfrac{\partial^m e^{\alpha u}}{\partial q^m} &= \tfrac{1}{m!}\tfrac{\partial^{m-1}}{\partial q^{m-1}}\left(\alpha e^{\alpha u}\tfrac{\partial u}{\partial q}\right) = \tfrac{\alpha}{m}\sum_{k=0}^{m-1} \tfrac{1}{k!(m-1-k)!}\tfrac{\partial^k e^{\alpha u}}{\partial q^k}\tfrac{\partial^{m-k}\phi}{\partial q^{m-k}} \\ &= \tfrac{\alpha}{m}\sum_{k=0}^{m-1} \tfrac{m-k}{m}\left(\tfrac{1}{k!}\tfrac{\partial^k e^{\alpha u}}{\partial q^k}\right)\left[\tfrac{1}{(m-k)!}\tfrac{\partial^{m-k}\phi}{\partial q^{m-k}}\right] \end{aligned}$$

On pose $q = 0$ dans l'expression ci-dessus et en utilisant la définition (3.1) et la propriété (3.2) on obtient :

$$\begin{aligned} D_m\left(e^{\alpha\phi}\right) &= \alpha \sum_{k=0}^{m-1} \left(1-\tfrac{k}{m}\right) D_{m-k}\left(\phi\right) D_k\left(e^{\alpha\phi}\right) \\ &= \alpha \sum_{k=0}^{m-1} \left(1-\tfrac{k}{m}\right) u_{m-k} D_k\left(e^{\alpha\phi}\right), \end{aligned}$$

où $m \geq 1$.

■

Théorème 3.1.8 *Pour une série d'homotopie de Maclaurin arbitraire :*

$$\phi = \sum_{k=0}^{+\infty} u_k q^k$$

on obtient les formules de récurrence :

$$\begin{aligned}
D_0(\sin\phi) &= \sin u_0 \\
D_0(\cos\phi) &= \cos u_0 \\
D_m(\sin\phi) &= \sum_{k=0}^{m-n}\left(1-\tfrac{k}{m}\right) D_{m-k}(\phi)D_k(\cos\phi) \\
&= \sum_{k=0}^{m-n}\left(1-\tfrac{k}{m}\right) u_{m-k}D_k(\cos\phi) \\
D_m(\cos\phi) &= \sum_{k=0}^{m-n}\left(1-\tfrac{k}{m}\right) D_{m-k}(\phi)D_k(\sin\phi) \\
&= \sum_{k=0}^{m-n}\left(1-\tfrac{k}{m}\right) u_{m-k}(\phi)D_k(\sin\phi)
\end{aligned}$$

où $m \geq 1$ est un entier.

Preuve.

Selon la définition (3.1), on a :

$$D_0\left(\sin\phi\right) = \sin u_0 \qquad\qquad D_0\left(\cos\phi\right) = \cos u_0,$$

*En utilisant la formule d'**Euler** et le théorème 2, on obtient :*

$$D_m\left(\sin\phi\right) = D_m\left(\frac{e^{i\phi}-e^{-i\phi}}{2i}\right) = \frac{1}{2i}\left[D_m\left(e^{i\phi}\right) - D_m\left(e^{-i\phi}\right)\right] \tag{3.15}$$

et

$$D_m\left(\cos\phi\right) = D_m\left(\frac{e^{i\phi}+e^{-i\phi}}{2}\right) = \frac{1}{2}\left[D_m\left(e^{i\phi}\right) + D_m\left(e^{-i\phi}\right)\right]. \tag{3.16}$$

D'après le théorème 7 et le théorème 2, on a :

$$\begin{aligned}
D_m\left(e^{i\phi}\right) &= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) D_k\left(e^{i\phi}\right) D_{m-k}\left(i\phi\right) \\
&= i\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) D_k\left(e^{i\phi}\right) D_{m-k}\left(\phi\right),
\end{aligned}$$

et de même :

$$\begin{aligned}
D_m\left(e^{-i\phi}\right) &= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) D_k\left(e^{-i\phi}\right) D_{m-k}\left(-i\phi\right) \\
&= -i\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) D_k\left(e^{-i\phi}\right) D_{m-k}\left(\phi\right),
\end{aligned}$$

*En remplaçant les deux expressions ci-dessus dans (3.15) et (3.16), puis en utilisant le théorème 2 et la formule d'**Euler**, on a :*

$$\begin{aligned} D_m(\sin\phi) &= \tfrac{1}{2}\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)D_{m-k}(\phi)\left[D_k\left(e^{i\phi}\right)+D_k\left(e^{-i\phi}\right)\right] \\ &= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)D_{m-k}(\phi)\,D_k\left(\tfrac{e^{i\phi}+e^{-i\phi}}{2}\right) \\ &= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)D_{m-k}(\phi)\,D_k(\cos\phi) \\ &= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u_{m-k}D_k(\cos\phi) \end{aligned}$$

de même :

$$\begin{aligned} D_m(\cos\phi) &= \tfrac{i}{2}\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)D_{m-k}(\phi)\left[D_k\left(e^{i\phi}\right)-D_k\left(e^{-i\phi}\right)\right] \\ &= -\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)D_{m-k}(\phi)\,D_k\left(\tfrac{e^{i\phi}-e^{-i\phi}}{2i}\right) \\ &= -\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)D_{m-k}(\phi)\,D_k(\sin\phi) \\ &= -\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u_{m-k}D_k(\sin\phi) \end{aligned}$$

■

Les résultats (3.6), (3.10) et (3.14) ont été prouvés par **Molabahrami** et **Khani** (2009)[34]. La plupart des autres ont été prouvés par Liao (2009a)[8],[1]. En utilisant ces théorèmes, on peut calculer les dérivées d'homotopie de toute fonction C^∞ donnée par série d'homotopie de **Maclaurin** . Par exemple, pour $\phi=\sum_{k=0}^{+\infty}u_kq^k$, on a :

$$D_m\left(3\phi^2+4e^{-5\phi}\sin\phi\right)=3\sum_{k=0}^{m}u_{m-k}u_k+4\sum_{k=0}^{m}D_k\left(e^{-5\phi}\right)D_{m-k}(\sin\phi),$$

où $D_k\left(e^{-5\phi}\right)$ et $D_{m-k}(\sin\phi)$ sont données par le théorème 7 et le théorème 8. **Molabahrami** et **Khani** (2009)[34] et **Liao** (2009a)[8], **Turkyilmazoglu** (2010) ont prouvé le théorème suivant :

Théorème 3.1.9 *Soit $\check{D}$ un opérateur défini par :*

$$\check{D}_m(\phi) = \frac{1}{m!}\frac{\partial^m \phi}{\partial q^m},$$

pour toute fonction $f \in C^\infty([a,b])$, et pour tout série d'homotopie :

$$\phi = \sum_{k=0}^{+\infty} u_k q^k$$

On a :

$$\check{D}_0\left[f(\phi)\right] = f(\phi), \tag{3.17}$$

$$\check{D}_m\left[f(\phi)\right] = \sum_{k=0}^{m-n}\left(1-\frac{k}{m}\right)\check{D}_{m-k}(\phi)\frac{\partial}{\partial\phi}\left\{\check{D}_k\left[f(\phi)\right]\right\} \tag{3.18}$$

et

$$D_m[f(\phi)] = \left\{\check{D}_m\left[f(\phi)\right]\right\}\Big|_{q=0} \tag{3.19}$$

Preuve.

Il est clair que $\check{D}_0\left[f(\phi)\right] = f(\phi)$.Si $m \geq 0$, et par la règle de **Leibniz** pour les dérivées de produits, on a :

$$\begin{aligned}
\check{D}_m\left[f\left(\phi\right)\right] &= \tfrac{1}{m!}\tfrac{\partial^m f(\phi)}{\partial q^m} = \tfrac{1}{m!}\tfrac{\partial^{m-1}}{\partial q^{m-1}}\left[\tfrac{\partial f(\phi)}{\partial\phi}\tfrac{\partial\phi}{\partial q}\right] \\
&= \tfrac{1}{m!}\sum_{k=0}^{m-1}\tfrac{(m-1)!}{k!(m-1-k)!}\tfrac{\partial^{m-1-k}}{\partial q^{m-1-k}}\left(\tfrac{\partial\phi}{\partial q}\right)\tfrac{\partial^k}{\partial q^k}\left[\tfrac{\partial f(\phi)}{\partial\phi}\right] \\
&= \sum_{k=0}^{m-1}\tfrac{(m-k)}{m}\left[\tfrac{1}{(m-1)!}\tfrac{\partial^{m-k}\phi}{\partial q^{m-k}}\right]\left\{\tfrac{1}{k!}\tfrac{\partial^k}{\partial q^k}\left[\tfrac{\partial f(\phi)}{\partial\phi}\right]\right\} \\
&= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)\check{D}_{m-k}\left(\phi\right)\check{D}_k\left[\tfrac{\partial f(\phi)}{\partial\phi}\right],
\end{aligned} \tag{3.20}$$

De la relation

$$\check{D}_k\left[\frac{\partial f\left(\phi\right)}{\partial\phi}\right] = \frac{\partial}{\partial\phi}\left\{\check{D}_k\left[f\left(\phi\right)\right]\right\},$$

on a :

$$\check{D}_m\left[f\left(\phi\right)\right] = \sum_{k=0}^{m-1}\left(1-\frac{k}{m}\right)\check{D}_{m-k}\left(\phi\right)\frac{\partial}{\partial\phi}\left\{\check{D}_k\left[f\left(\phi\right)\right]\right\}$$

pour $m \geq 1$.Ensuite, selon la définition de $\check{D}_m$ on a facilement :

$$D_m\left[f\left(\phi\right)\right] = \left\{\check{D}_m\left[f\left(\phi\right)\right]\right\}\Big|_{q=0}.$$

■

Théorème 3.1.10 *Pour une fonction C^∞ $f(u)$ et une série de Maclaurin d'homotopie $\phi = \sum_{k=0}^{+\infty} u_k q^k$, on a :*

$$D_0[f(\phi)] = f(u_0), \tag{3.21}$$

$$D_m[f(\phi)] = \sum_{k=0}^{m-1} \left(1 - \tfrac{k}{m}\right) u_{m-k} \tfrac{\partial\{D_k[f(\phi)]\}}{\partial u_0}, \tag{3.22}$$

et

$$D_m[f(\phi)] = \sum_{k=0}^{m-1} \left(1 - \tfrac{k}{m}\right) u_{m-k} D_k[f'(\phi)], \quad m \geq 1 \tag{3.23}$$

Preuve.

On peut vérifier facilement, de la définition (3.1), que la relation (3,27) est vraie.
Pour $q = 0$, on a :

$$\check{D}_{m-k}(\phi) = D_{m-k}(\phi) = u_{m-k},$$

D'ailleurs si $q = 0$, nous avons $\phi = u_0$ et donc $\frac{\partial}{\partial \phi} = \frac{\partial}{\partial u_0}$.Puis, selon le théorème 9, on a :

$$D_m[f(\phi)] = \sum_{k=0}^{m-1} \left(1 - \frac{k}{m}\right) u_{m-k} \frac{\partial\{D_k[f(\phi)]\}}{\partial u_0}.$$

Si $q = 0$ dans (3.20), nous obtenons :

$$D_m[f(\phi)] = \sum_{k=0}^{m-1} \left(1 - \frac{k}{m}\right) u_{m-k} D_k[f'(\phi)].$$

■

Notons que les théorèmes 7 et 8 sont des cas particuliers de (3.23) du théorème 10. A l'aide de la formule de récurrence (3.22) du théorème 10, il est facile d'obtenir les dérivées homotopiques d'ordre supérieur d'une fonction C^∞ arbitraire et une série homotopiques

de Maclaurin donnée $\phi = \sum_{k=0}^{+\infty} u_k q^k$. Par exemple si $f(\phi) = \phi^\alpha$ alors

$$\begin{aligned}
D_0(\phi^\alpha) &= u_0^\alpha, \\
D_1(\phi^\alpha) &= \alpha u_0^{\alpha-1} u_1, \\
D_2(\phi^\alpha) &= \tfrac{1}{2}\alpha(\alpha-1) u_0^{\alpha-2} u_1^2 + \alpha u_0^{\alpha-1} u_2, \\
D_3(\phi^\alpha) &= \tfrac{1}{6}\alpha(\alpha-1)(\alpha-2) u_0^{\alpha-3} u_1^3 + \alpha(\alpha-1) u_0^{\alpha-2} u_1 u_2 + \alpha u_0^{\alpha-1} u_3, \\
&\vdots
\end{aligned}$$

ou $\alpha \neq 0$ est indépendant du paramètre de d'homotopie q.
De même, il donne quand $f(\phi) = \alpha^\phi$, que :

$$\begin{aligned}
D_0\left(\alpha^\phi\right) &= \alpha^{u_0}, \\
D_1\left(\alpha^\phi\right) &= (\ln\alpha)\alpha^{u_0} u_1, \\
D_2\left(\alpha^\phi\right) &= \tfrac{1}{2}(\ln\alpha)^2 \alpha^{u_0} u_1^2 + (\ln\alpha)\alpha^{u_0} u_2, \\
D_3\left(\alpha^\phi\right) &= \tfrac{1}{6}(\ln\alpha)^3 \alpha^{u_0} u_1^3 + (\ln\alpha)^2 \alpha^{u_0} u_1 u_2 + (\ln\alpha)\alpha^{u_0} u_3, \\
&\vdots
\end{aligned}$$

où $\alpha > 0$ est indépendant du paramètre d'homotopie q.
Ces résultats sont exactement les mêmes que ceux donnés par le théorème 9.

En général, au moyen de la formule de récurrence (3.22) du théorème 10, nous avons pour toute fonction C^∞ donnée $f(\phi)$ [1] :

$$\begin{aligned}
D_0[f(\phi)] &= f(u_0), \\
D_1[f(\phi)] &= f'(u_0) u_1, \\
D_2[f(\phi)] &= \tfrac{1}{2} f''(u_0) u_1^2 + f'(u_0) u_2, \\
D_3[f(\phi)] &= \tfrac{1}{6} f'''(u_0) u_1^3 + f''(u_0) u_1 u_2 + f'(u_0) u_3, \\
D_4[f(\phi)] &= \tfrac{1}{24} f^{(4)}(u_0) u_1^4 + \tfrac{1}{2} f'''(u_0) u_1^2 u_2 + f''(u_0)\left(u_1 u_3 + \tfrac{u_2^2}{2}\right) + f'(u_0) u_4, \\
&\vdots
\end{aligned}$$

Dans la pratique, au moyen de système de calcul formel comme Mathematica et Maple il est facile d'obtenir $D_m[f(\phi)]$ pour m assez grand .
Par exemple, en utilisant les commandes Mathematica suivantes :[1]

```
GetD[0] := f[u[0]] ;
GetD[m_] := Sum[(1-k/m)*u[m-k]*D[GetD[k],u[0]],{k,0,m-1}] ;
```

On peut obtenir $D_m[f(\phi)]$ pour toute fonction C^∞ donnée $f(\phi)$ d'une série d'homotopie

de Maclaurin $\phi = \sum_{k=0}^{+\infty} u_k q^k$. Donc, pour tout fonction C^∞ $f(\phi)$ donné, où ϕ est une série d'homotopie de Maclaurin, on peut toujours obtenir sa dérivée d'homotopie d'ordre m : $D_m[f(\phi)]$ au moyen des théorèmes ci-dessus.
En outre, les théorèmes suivants peuvent être démontrés de manière similaire.[1]

Théorème 3.1.11 *Pour une fonction C^∞ $f(u,w)$ et pour les deux séries de Maclaurin d'homotopie*

$$\phi = \sum_{k=0}^{+\infty} u_k q^k \qquad\qquad \psi = \sum_{k=0}^{+\infty} w_k q^k$$

on a

$$D_0[f(\phi,\psi)] = f(u_0,w_0), \tag{3.24}$$

$$\begin{aligned} D_m[f(\phi,\psi)] = & \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) u_{m-k}\tfrac{\partial\{D_k[f(\phi,\psi)]\}}{\partial u_0} \\ & +\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) w_{m-k}\tfrac{\partial\{D_k[f(\phi,\psi)]\}}{\partial w_0}, \end{aligned} \tag{3.25}$$

et

$$\begin{aligned} D_m[f(\phi,\psi)] = & \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) u_{m-k} D_k\left[\tfrac{\partial f(\phi,\psi)}{\partial\phi}\right] \\ & +\sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right) w_{m-k} D_k\left[\tfrac{\partial f(\phi,\psi)}{\partial\psi}\right] \end{aligned} \tag{3.26}$$

pour $m \geq 1$.

Théorème 3.1.12 *Pour une fonction C^∞ $f(u,u',u'')$ et la série de Maclaurin d'homotopie*

$$\phi = \sum_{k=0}^{+\infty} u_k q^k$$

Avec la définition :

$$\phi' = \sum_{k=0}^{+\infty} u'_k q^k \qquad\qquad \phi'' = \sum_{k=0}^{+\infty} u''_k q^k$$

On a :

$$D_0[f(\phi,\phi',\phi'')] = f(u_0,u'_0,u''_0) \tag{3.27}$$

$$
\begin{aligned}
D_m\left[f\left(\phi,\phi',\phi''\right)\right] &= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u_{m-k}\tfrac{\partial\{D_k[f(\phi,\phi',\phi'')]\}}{\partial u_0} \\
&+ \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u'_{m-k}\tfrac{\partial\{D_k[f(\phi,\phi',\phi'')]\}}{\partial u'_0} \\
&+ \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u''_{m-k}\tfrac{\partial\{D_k[f(\phi,\phi',\phi'')]\}}{\partial u''_0}
\end{aligned} \tag{3.28}
$$

et

$$
\begin{aligned}
D_m\left[f\left(\phi,\phi',\phi''\right)\right] &= \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u_{m-k}D_k\left[\tfrac{\partial f(\phi,\phi',\phi'')}{\partial\phi}\right] \\
&+ \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u'_{m-k}D_k\left[\tfrac{\partial f(\phi,\phi',\phi'')}{\partial\phi'}\right] \\
&+ \sum_{k=0}^{m-1}\left(1-\tfrac{k}{m}\right)u''_{m-k}D_k\left[\tfrac{\partial f(\phi,\phi',\phi'')}{\partial\phi''}\right]
\end{aligned} \tag{3.29}
$$

Pour $m \geq 1$.

Notons que les théorèmes ci-dessus ne sont pas valable pour les fonctions qui contiennent explicitement le paramètre d'homotopie, comme $f(\phi,\psi,q)$.
Dans ce cas, il est plus efficace de développer directement la série de Maclaurin d'homotopie puis utiliser le théorème suivant [1] :

Théorème 3.1.13 *Soit :*

$$
\phi = \sum_{i=0}^{+\infty} u_i q^i \qquad et \qquad \psi = \sum_{j=0}^{+\infty} w_j q^j
$$

sont deux séries de Maclaurin d'homotopie qui sont analytiques pour $q \in [0;a]$ *soit :*

$$
N\left(\phi,\psi,q\right) = \sum_{k=0}^{+\infty}\delta_k q^k
$$

la série de Maclaurin d'homotopie de $N(\phi,\psi,q)$,
Alors :

$$
D_k\left[N\left(\phi,\psi,q\right)\right] = \delta_k = \left[\frac{1}{k!}\frac{d^k}{dq^k}N\left(\sum_{m=0}^{+\infty}u_m q^m, \sum_{n=0}^{+\infty}w_n q^n, q\right)\right]\Bigg|_{q=0} \tag{3.30}
$$

Preuve.
La formule (3.30) vient directement de la définition (3.1). À l'aide du théorème 13, nous pouvons obtenir la dérivée d'homotopie de tout opérateur non linéaire donné.

Par exemple, soit $\phi = \sum_{n=0}^{+\infty} u_n q^n$ *une série d'homotopie de Maclaurin. A l'aide de système de calcul formel comme Mathematica, il est facile d'obtenir la série d'homotopie de Maclaurin de* $\sin(q\phi)/q$, *i.e :*

$$\begin{aligned}\frac{\sin(q\phi)}{q} \quad \sim \quad & u_0 + u_1 q + \left(u_2 - \tfrac{1}{6}u_0^3\right) q^2 + \left(u_3 - \tfrac{1}{2}u_0^2 u_1\right) q^3 \\ & + \left(u_4 - \tfrac{1}{2}u_0^2 u_2 - \tfrac{1}{2}u_0 u_1^2 + \tfrac{1}{120}u_0^5\right) q^4 + ...\end{aligned}$$

Donc on a :

$$D_0\left[\frac{\sin(q\phi)}{q}\right] = u_0, \;\; D_1\left[\frac{\sin(q\phi)}{q}\right] = u_1, \quad D_2\left[\frac{\sin(q\phi)}{q}\right] = u_2 - \frac{1}{6}u_0^3 \tag{3.31}$$

et ainsi de suite. Pour plus de détails, on peut voir ***Liao*** *(2009b)[8]. En utilisant les théorèmes mentionnés ci-dessus, on peut dériver des formules explicites de dérivées d'homotopie de nombreuses fonctions complexes.*
Par exemple, l'autre façon pour obtenir $D_m[\sin(q\phi)/q]$ *est donnée par le théorème suivant.* ■

Théorème 3.1.14 *Pour une série d'homotopie de Maclaurin :*

$$\phi = \sum_{k=0}^{+\infty} u_k q^k,$$

où ϕ *est analytique pour* $q \in [0; a]$ *,alors :*

$$D_m\left[\frac{\sin(q\phi)}{q}\right] = D_{m+1}\left[\sin(q\phi)\right], \qquad m \geq 0,$$

où

$$\begin{aligned}
D_0\left[\sin(q\phi)\right] &= 0, \\
D_0\left[\cos(q\phi)\right] &= 1, \\
D_n\left[\sin(q\phi)\right] &= \sum_{k=0}^{n-1}\left(1 - \tfrac{k}{n}\right) D_{n-1-k}(\phi) D_k\left[\cos(q\phi)\right] \\
&= \sum_{k=0}^{n-1}\left(1 - \tfrac{k}{n}\right) u_{n-1-k} D_k\left[\cos(q\phi)\right] \\
D_n\left[\cos(q\phi)\right] &= -\sum_{k=0}^{n-1}\left(1 - \tfrac{k}{n}\right) D_{n-1-k}(\phi) D_k\left[\sin(q\phi)\right] \\
&= -\sum_{k=0}^{n-1}\left(1 - \tfrac{k}{n}\right) u_{n-1-k} D_k\left[\sin(q\phi)\right]
\end{aligned}$$

pour $n \geq 1$.

Preuve.

Selon la définition de (3.1), on a :

$$D_0 \left[\sin\left(q\phi\right)\right] = \sin 0 = 0, \quad D_0 \left[\cos\left(q\phi\right)\right] = \cos 0 = 1,$$

Selon la définition (3.1), on a :

$$\sin\left(q\phi\right) = \sum_{k=1}^{+\infty} D_k \left[\sin\left(q\phi\right)\right] q^k,$$

ce qui donne

$$\frac{\sin\left(q\phi\right)}{q} = \sum_{k=1}^{+\infty} D_k \left[\sin\left(q\phi\right)\right] q^{k-1} = \sum_{m=0}^{+\infty} D_{m+1} \left[\sin\left(q\phi\right)\right] q^m$$

de sorte que

$$D_m \left[\frac{\sin\left(q\phi\right)}{q}\right] = D_{m+1} \left[\sin\left(q\phi\right)\right].$$

D'après le théorème 8,on a :

$$D_n \left[\sin\left(q\phi\right)\right] = \sum_{k=0}^{n-1} \left(1 - \frac{k}{n}\right) D_{n-k}\left(q\phi\right) D_k \left[\cos\left(q\phi\right)\right],$$

ce qui donne d'après le théorème 1 que

$$\begin{aligned} D_n \left[\sin\left(q\phi\right)\right] &= \sum_{k=0}^{n-1} \left(1 - \tfrac{k}{n}\right) D_{n-1-k}\left(\phi\right) D_k \left[\cos\left(q\phi\right)\right] \\ &= \sum_{k=0}^{n-1} \left(1 - \tfrac{k}{n}\right) u_{n-1-k}\left(\phi\right) D_k \left[\cos\left(q\phi\right)\right] \end{aligned}$$

pour $n \geq 1$, *De même :*

$$D_n \left[\cos\left(q\phi\right)\right] = -\sum_{k=0}^{n-1} \left(1 - \frac{k}{n}\right) u_{n-1-k} D_k \left[\sin\left(q\phi\right)\right], \qquad n \geq 1.$$

Ceci termine la preuve. ■

En utilisant le théorème (14), nous obtenons exactement les mêmes résultats que

(3.31). En utilisant Mathematica, il est facile d'obtenir $D_n\left[\sin\left(q\phi\right)/q\right]$ ou $0 \leq n \leq 4$, par les commandes suivantes [1] :

```
Dsin[0] = 0 ;
Dcos[0] =1 ;
Dsin[n_] := Sum[(1 - k/n)*u[n-1-k]*Dcos[k],{k,0,n-1}] ;
Dcos[n_] := -Sum[(1 - k/n)*u[n-1-k]*Dsin[k],{k,0,n-1}] ;
For[k = 0, k<=4, k=k+1, Dsin[k+1] // Print]
```

3.2 Les équations de déformation d'ordre supérieur :

Dans cette section, nous donnons les équations de déformation d'ordre supérieur pour les différents types d'équations de déformation d'ordre zéro.[1]

Lemme 3.2.1 *Soit :*

$$\phi = \sum_{m=0}^{+\infty} u_m q^m$$

une série de Maclaurin d'homotopie, où $q \in [0,1]$ est le paramètre d'homotopie, on a :

$$D_m\left[\left(1-q\right)L\left(\phi - u_0\right)\right] = L\left(u_m - \chi_m u_{m-1}\right)$$

où

$$\chi_m = \begin{cases} 0 & si \;\; m \leq 1, \\ 1 & si \;\; m > 1. \end{cases} \tag{3.32}$$

Preuve. *Puisque L est un opérateur linéaire indépendant de q, alors :*

$$\left(1-q\right)L\left(\phi - u_0\right) = L\left(\phi - q\phi + qu_0 - u_0\right)$$

D'après le théorème 3, le théorème 2 et l'état (3.2) du théorème 1, on a :

$$\begin{aligned} D_m\left[\left(1-q\right)L\left(\phi - u_0\right)\right] &= D_m\left[L\left(\phi - q\phi + u_0 q - u_0\right)\right] \\ &= L\left[D_m\left(\phi - q\phi + u_0 q - u_0\right)\right] \\ &= L\left[D_m\left(\phi\right) - D_m\left(q\phi\right) + u_0 D_m\left(q\right)\right] \\ &= L\left[u_m - u_{m-1} + u_0 D_m\left(q\right)\right], \end{aligned}$$

qui est égal à $L\left(u_m\right)$ si $m = 1$, et $L\left(u_m - u_{m-1}\right)$ si $m > 1$, respectivement.

Ainsi selon la définition (3.32) de χ_m on obtient :

$$D_m\left[(1-q)\,L\left(\phi-u_0\right)\right]=L\left(u_m-\chi_m u_{m-1}\right).$$

■

Théorème 3.2.1 *Si*

$$\phi=\sum_{m=0}^{+\infty}u_m(x,t)q^m$$

est la série d'homotopie de Maclaurin de l'équation de déformation d'ordre zéro

$$(1-q)L(\phi-u_0)=cqH(x,t)N(\phi) \tag{3.33}$$

alors l'équation de déformation d'ordre m correspondante est :

$$L[u_m(x,t)-\chi_m u_{m-1}(x,t)]=cH(x,t)D_{m-1}\left[N\left(\phi\right)\right] \tag{3.34}$$

où D_{m-1} est définie par (3.1) et χ_m est définie par (3.32).

Preuve. *D'aprè le théorème 5, on a :*

$$D_m\left[(1-q)\,L\left(\phi-u_0\right)\right]=D_m\left[qcH\left(x,t\right)N\left(\phi\right)\right]$$

Selon le lemme 1, on obtient :

$$D_m\left[(1-q)\,L\left(\phi-u_0\right)\right]=L\left(u_m-\chi_m u_{m-1}\right)$$

Selon le théorème 2 et (3,3), il détient :

$$D_m\left[qcH\left(x,t\right)N\left(\phi\right)\right]=cH(x,t)D_{m-1}\left[N\left(\phi\right)\right]$$

Ainsi, on a l'équation de déformation d'ordre m :

$$L\left(u_m-\chi_m u_{m-1}\right)=cH(x,t)D_{m-1}\left[N\left(\phi\right)\right]$$

■

La liberté dont nous disposons sur le choix de l'équation de déformation d'ordre 0 nous permet d'introduire dans sa définition un nombre arbitraire de paramètres constants

de contrôle de la convergence $c_0, c_1, c_2, ...$

Théorème 3.2.2 *Si la série*

$$\sum_{k=0}^{+\infty} c_k q^k = c_0 + c_1 q + c_2 q^2 + ...$$

converge en $q = 1$, *où* c_0, c_1,... *sont des paramètres constants de contrôle la convergence avec* $c_0 \neq 0$, *et* $\sum_{k=0}^{+\infty} c_k \neq 0$, *et si*

$$\phi = \sum_{m=0}^{+\infty} u_m(x,t) q^m$$

est la série d'homotopie de Maclaurin de l'équation de déformation d'ordre zéro

$$(1-q)L(\phi - u_0) = qH(x,t) \left(\sum_{k=0}^{+\infty} c_k q^k \right) N(\phi) \tag{3.35}$$

alors l'équation de déformation d'ordre m *correspondant est :*

$$L[u_m(x,t) - \chi_m u_{m-1}(x,t)] = H(x,t) \sum_{k=1}^{m} c_{k-1} D_{m-k} \left[N(\phi) \right] \tag{3.36}$$

où D_{m-k} *est définie par (3.1) et* χ_m *est définie par (3.32).*

Preuve. On pose :

$$\psi = \sum_{k=0}^{+\infty} c_k q^{k+1},$$

On a d'après (3.2) :

$$D_0(\psi) = 0, \quad D_k(\psi) = c_{k-1},$$

Pour $k \geq 1$, selon la théorème 5, on a :

$$D_m \left[(1-q) L(\phi - u_0) \right] = D_m \left[H(x,t) \psi N(\phi) \right].$$

Selon lemme 1 on obtient :

$$D_m\left[(1-q)\,L\left(\phi-u_0\right)\right]=L\left(u_m-\chi_m u_{m-1}\right).$$

Puis, en utilisant le théorème 2 et (3.5) du théorème 1, on a :

$$\begin{aligned} D_m\left[H\left(x,t\right)\psi N\left(\phi\right)\right] &= H\left(x,t\right)\sum_{k=0}^{m} D_k\left(\psi\right) D_{m-k}\left[N\left(\phi\right)\right] \\ &= H\left(x,t\right)\sum_{k=1}^{m} D_k\left(\psi\right) D_{m-k}\left[N\left(\phi\right)\right] \\ &= H\left(x,t\right)\sum_{k=1}^{m} c_{k-1} D_{m-k}\left[N\left(\phi\right)\right]. \end{aligned}$$

Ainsi, l'équation de déformation d'ordre supérieur correspondant est :

$$L\left(u_m-\chi_m u_{m-1}\right)=H\left(x,t\right)\sum_{k=1}^{m} c_{k-1} D_{m-k}\left[N\left(\phi\right)\right].$$

■

L'équation de déformation d'ordre zéro (3.33) est un cas particulier de l'équation de déformation d'ordre zéro (3,35) pour le cas où $c_k=0$ avec $k\geq 1$.
De plus, les paramètres de contrôle de convergence $c_0, c_1, c_2...$ peuvent être généralisés en des paramètres non-constants, et ainsi, en combinant la fonction auxiliaire $H(x,t)$ avec le paramètre de contrôle de convergence c_k nous avons une équation de déformation d'ordre zéro plus générale .

Théorème 3.2.3 *Si la série*

$$\sum_{k=0}^{+\infty}\beta_k(x,t)q^k$$

converge pour $q=1$, *vers une fonction non nulle, où* $\beta_0\left(x,t\right),\beta_1\left(x,t\right),...,$ *sont appelées fonctions de contrôle de convergence et de plus, si*

$$\phi=\sum_{k=0}^{+\infty} u_m\left(x,t\right)q^k$$

est la série d'homotopie de Maclaurin de l'équation de déformation d'ordre zéro,

$$(1-q)L(\phi-u_0)=q\left(\sum_{k=0}^{+\infty}\beta_k(x,t)q^k\right)N(\phi) \tag{3.37}$$

alors l'équation de déformation d'ordre m correspondant est :

$$L[u_m(x,t) - \chi_m u_{m-1}(x,t)] = \sum_{k=0}^{+\infty} \beta_{k-1}(x,t) D_{m-k}\left[N\left(\phi\right)\right] \tag{3.38}$$

où D_{m-k} est définie par (3.1) et χ_m est définie par (3.32).

Preuve.

On pose

$$\psi = \sum_{k=0}^{+\infty} \beta_k\left(x,t\right) q^{k+1},$$

et d'après (3,2) on a :

$$D_0\left(\psi\right) = 0, \quad D_k\left(\psi\right) = \beta_{k-1}\left(x,t\right)$$

pour $k \geq 1$, et d'après le théorème 5 on a :

$$D_m\left[\left(1-q\right) L\left(\phi - u_0\right)\right] = D_m\left[\psi N\left(\phi\right)\right].$$

Selon lemme 1 on a :

$$D_m\left[\left(1-q\right) L\left(\phi - u_0\right)\right] = L\left(u_m - \chi_m u_{m-1}\right).$$

Ensuite, en utilisant le théorème 2 et (3.5) du théorème 1, nous avons :

$$\begin{aligned} D_m\left[\psi N\left(\phi\right)\right] &= \sum_{k=0}^{m} D_k\left(\psi\right) D_{m-k}\left[N\left(\phi\right)\right] \\ &= \sum_{k=1}^{m} D_k\left(\psi\right) D_{m-k}\left[N\left(\phi\right)\right] \\ &= \sum_{k=0}^{m} \beta_{k-1}\left(x,t\right) D_{m-k}\left[N\left(\phi\right)\right] \end{aligned}$$

Ainsi, l'équation de déformation d'ordre supérieur correspondante est :

$$L\left[u_m\left(x,t\right) - \chi_m u_{m-1}\left(x,t\right)\right] = \sum_{k=1}^{m} \beta_{k-1}\left(x,t\right) D_{m-k}\left[N\left(\phi\right)\right].$$

■

Notez que l'équation de déformation d'ordre zéro (3,35) est un cas particulier de l'équation de déformation d'ordre zéro (3,37) en cas de $\beta_k\left(x,t\right) = c_k H\left(x,t\right)$ pour $k \geq 0$,

Donc le concept de fonction de contrôle de la convergence est plus général : il contient non seulement les paramètres de contrôle de la convergence constante mais aussi la fonction auxiliaire $H(x,t)$.
Ceci indique l'essence (le rôle) de la fonction auxiliaire $H(x,t)$.

Définition 3.2.1 *Une fonction f analytique du paramètre d'homotopie $q \in [0,1]$ est appellée une* ***fonction de déformation****, si $f = 0$ quand $q = 0$, $f = 1$ lorsque $q = 1$, et sa série de Maclaurin*

$$f = \sum_{k=0}^{+\infty} \alpha_k q^k$$

converge absolument pour $q = 1$,

$$\sum_{k=0}^{+\infty} \alpha_k = 1$$

où α_k peut être une constante ou une fonction des variable spatio-temporelle indépendante.

Lemme 3.2.2 *Soit :*

$$\phi = \sum_{m=0}^{+\infty} u_m(x,t) q^m$$

une série de Maclaurin d'homotopie, où $q \in [0,1]$ est le paramètre d'homotopie.
Si $\alpha(q)$ est une fonction de déformation, c'est à dire $\alpha(0) = 0$, $\alpha(1) = 1$ et sa série de Maclaurin est :

$$\alpha(q) = \sum_{k=1}^{+\infty} \alpha_k q^k$$

existe et converge absolument pour $q = 1$, avec α_k est constante, alors :

$$D_m \left\{ [1 - \alpha(q)] L \left(\phi - u_0 \right) \right\} = L \left(u_m - \sum_{n=1}^{m-1} \alpha_n u_{m-n} \right)$$

Preuve. *On pose :*

$$\Phi = \phi - u_0 = \sum_{k=1}^{+\infty} u_k q^k, \qquad \Psi = \alpha \left(q \right) = \sum_{k=1}^{+\infty} \alpha_k q^k.$$

Selon (3.2) du théorème 1, on a :

$$D_0(\Phi) = D_0(\Psi) = 0, \quad D_n(\Psi) = u_n, \quad D_n(\Psi) = \alpha_n, \quad n \geq 1.$$

Ensuite, en utilisant les théorèmes 1- 3, on obtient :

$$\begin{aligned} D_m\{[1-\alpha(q)]L(\phi-u_0)\} &= D_m[(1-\Psi)L(\Phi)] = D_m[L(\Phi) - \Psi L(\Phi)] \\ &= D_m[L(\Phi)] - \sum_{n=0}^{m} D_n(\Psi) D_{m-n}[L(\Phi)] \\ &= L[D_m(\Phi)] - \sum_{n=0}^{m} D_n(\Psi) L[D_{m-n}(\Phi)] \\ &= L[D_m(\Phi)] - \sum_{n=1}^{m-1} D_n(\Psi) L[D_{m-n}(\Phi)] \\ &= L(u_m) - \sum_{n=1}^{m-1} \alpha_n L(u_{m-n}) \end{aligned}$$

ce qui donne, puisque α_k est constante, que

$$D_m\{[1-\alpha(q)]L(\phi-u_0)\} = L\left(u_m - \sum_{n=1}^{m-1} \alpha_n u_{m-n}\right)$$

■

Théorème 3.2.4 *Soit $\alpha(q)$ une fonction de déformation, i.e. $\alpha(0) = 0$, $\alpha(1) = 1$ et sa série de Maclaurin*

$$\alpha(q) = \sum_{k=1}^{+\infty} \alpha_k q^k$$

existe et converge absolument si $q = 1$ où k est constante, si la série :

$$\sum_{k=0}^{+\infty} \beta_k(x,t) q^k$$

converge pour $q = 1$ à une fonction non nulle, où $\beta_0(x,t), \beta_1(x,t), ...,$ sont appelées des fonctions de contrôle de la convergence, si

$$\phi = \sum_{m=0}^{+\infty} u_m(x,t) q^m$$

est la série d'homotopie de Maclaurin de l'équation de déformation d'ordre zéro,

$$[1-\alpha(q)]L(\phi-u_0)=q\left(\sum_{k=0}^{+\infty}\beta_k(x,t)q^k\right)N(\phi) \tag{3.39}$$

Alors l'équation de déformation d'ordre m correspondant est :

$$L[u_m(x,t)-\sum_{n=1}^{m-1}\alpha_n u_{m-n}(x,t)]=\sum_{k=1}^{+\infty}\beta_{k-1}(x,t)D_{m-k}\left[N\left(\phi\right)\right] \tag{3.40}$$

où D_{m-k} est définie par (3.1) .

Preuve. On pose :

$$\psi=\sum_{q=0}^{+\infty}\beta_k q^{k+1},$$

d'après (3.2), on a :

$$D_0\left(\psi\right)=0,\quad D_k\left(\psi\right)=\beta_{k-1}\left(x,t\right)$$

pour $k\geq 1$, selon le théorème 5, nous avons :

$$D_m\left\{\left[1-\alpha\left(q\right)\right]L\left(\phi-u_0\right)\right\}=D_m\left[\psi N\left(\phi\right)\right]$$

D'après le lemme 2, on a :

$$D_m\left\{\left[1-\alpha\left(q\right)\right]L\left(\phi-u_0\right)\right\}=L\left(u_m-\sum_{n=1}^{m-1}\alpha_n u_{m-n}\right).$$

Ensuite, en utilisant les théorèmes 1, 2 on obtient :

$$\begin{aligned}D_m\left[\psi N\left(\phi\right)\right] &= \sum_{k=0}^{m}D_k\left(\psi\right)D_{m-k}\left[N\left(\phi\right)\right]\\ &= \sum_{k=1}^{m}D_k\left(\psi\right)D_{m-k}\left[N\left(\phi\right)\right]\\ &= \sum_{k=1}^{m}\beta_{k-1}\left(x,t\right)D_{m-k}\left[N\left(\phi\right)\right]\end{aligned}$$

Donc l'équation de déformation d'ordre supérieur correspondante est :

$$L\left[u_m(x,t) - \sum_{n=1}^{m-1} \alpha_n u_{m-n}(x,t)\right] = \sum_{k=1}^{m} \beta_{k-1}(x,t)\, D_{m-k}\left[N(\phi)\right].$$

■

L'équation de déformation d'ordre zéro (3.37) et l'équation de déformation d'ordre supérieur correspondante (3.38) sont des cas particuliers de (3.39) et (3.40), pour le cas où $\alpha_1 = 1$, et $\alpha_k = 0$ et $k \geq 2$, respectivement.

Définition 3.2.2 *Un opérateur $A(\varphi, x, t, q)$ est appelée opérateur de déformation , si $A(\varphi, x, t, 0) = 0$ en $q = 0$ et $q = 1$, $A(\varphi, x, t, q)$, où $q \in [0, 1]$ est le paramètre d'homotopie, et où $\phi = \sum_{k=0}^{+\infty} u_k q^k$ est une série de Maclaurin d'homotopie.*

Théorème 3.2.5 *Soit $\alpha(q)$ représente une fonction de déformation, i.e. $\alpha(0) = 0$, $\alpha(1) = 1$ et sa série de Maclaurin*

$$\alpha(q) = \sum_{k=1}^{+\infty} \alpha_k q^k$$

existe et converge si $q = 1$, où α_k est constante. Soit A un opérateur de déformation i.e.

$$A(\varphi, x, t, 0) = A(\varphi, x, t, 1) = 0,$$

Si la série

$$\sum_{k=0}^{+\infty} \beta_k(x,t) q^k$$

converge pour $q = 1$ vers une fonction non nulle, où $\beta_0(x,t), \beta_1(x,t), \beta_2(x,t), ...$sont des fonctions appelées les fonctions de contrôle de la convergence, et si

$$\phi = \sum_{m=0}^{+\infty} u_m(x,t) q^m,$$

est la série d'homotopie de Maclaurin de l'équation de déformation d'ordre zéro :

$$[1-\alpha(q)]L(\phi-u_0)=q\left(\sum_{k=0}^{+\infty}\beta_k(x,t)q^k\right)N(\phi)+A(\varphi,x,t,q) \tag{3.41}$$

alors l'équation de déformation d'ordre m correspondante est :

$$L[u_m(x,t)-\sum_{n=1}^{m-1}\alpha_n u_{m-1}(x,t)]=\sum_{k=1}^{+\infty}\beta_{k-1}(x,t)D_{m-k}\left[N\left(\phi\right)\right]+D_mA(\varphi,x,t,q) \tag{3.42}$$

où D_{m-k} est définie par (3.1) .

Preuve. *On pose*

$$\psi=\sum_{k=0}^{+\infty}\beta_k\left(x,t\right)q^{k+1},$$

d'après (3.2), on a :

$$D_0\left(\psi\right)=0,\quad D_k\left(\psi\right)=\beta_{k-1}\left(x,t\right)$$

pour $k\geq 1$, selon le théorème 5, on a :

$$D_m\left\{\left[1-\alpha\left(q\right)\right]L\left(\phi-u_0\right)\right\}=D_m\left[\psi N\left(\phi\right)+A\left(\phi,x,t,q\right)\right].$$

Selon le lemme 2, on obtient :

$$D_m\left\{\left[1-\alpha\left(q\right)\right]L\left(\phi-u_0\right)\right\}=L\left(u_m-\sum_{n=1}^{m-1}\alpha_n u_{m-n}\right)$$

Ensuite, en utilisant les théorème 1,2 , on a :

$$\begin{aligned}D_m\left[\psi N\left(\phi\right)+A\left(\phi,x,t,q\right)\right] &= \sum_{k=0}^{m}D_k\left(\psi\right)D_{m-k}\left[N\left(\phi\right)\right]+D_m\left[A\left(\phi,x,t,q\right)\right]\\ &= \sum_{k=1}^{m}D_k\left(\psi\right)D_{m-k}\left[N\left(\phi\right)\right]+D_m\left[A\left(\phi,x,t,q\right)\right]\\ &= \sum_{k=0}^{m}\beta_{k-1}\left(x,t\right)D_{m-k}\left[N\left(\phi\right)\right]+D_m\left[A\left(\phi,x,t,q\right)\right]\end{aligned}$$

Donc, l'équation de déformation d'ordre supérieur correspondant est :

$$L\left(u_m-\sum_{n=1}^{m-1}\alpha_n u_{m-n}\right) = \sum_{k=1}^{m}\beta_{k-1}\left(x,t\right)D_{m-k}\left[N\left(\phi\right)\right]+D_m\left[A\left(\phi,x,t,q\right)\right]$$

■

Notez que l'équation de déformation d'ordre zéro (3.39) et son équation de déformation d'ordre supérieur (3.40) sont des cas particuliers de (3.41) et (3.42) en cas de

$$A\left(\phi, x, t, q\right) = 0$$

Bien que (3.41) et (3.42) soient générales, une équation de déformation d'ordre zéro plus générale encore peut être donnée par le théorème suivant.

Lemme 3.2.3 *Soit*

$$\phi = \sum_{m=0}^{+\infty} u_m q^m$$

la série de Maclaurin d'homotopie, si $\alpha(x,t,q)$ *est la fonction de déformation, i.e* $\alpha(x,t,0) = 0,\ \alpha(x,t,1) = 1$ *et sa série de Maclaurin*

$$\alpha(x,t,q) \sim \sum_{k=1}^{+\infty} \alpha_k(x,t) q^k$$

existe et converge pour $q = 1$ *.*
Alors :

$$D_m\left\{\left[1 - \alpha(x,t,q)\right] L(\phi - u_0)\right\} = L(u_m) - \sum_{n=1}^{m-1} \alpha_n(x,t) L(u_{m-n}).$$

Théorème 3.2.6 *Soit* $\alpha(x,t,q)$ *une fonction de déformation, i.e.* $\alpha(x,t,0) = 0$ *et* $\alpha(x,t,1) = 1$*, et sa série de Maclaurin*

$$\alpha(x,t,q) = \sum_{k=1}^{+\infty} \alpha_k(x,t) q^k$$

existe et converge pour $q = 1$*, où* $\alpha_k(x,t)$ *est dépendant de* x *et* t *,soit* B *un operateur dépend de* ϕ, x, t *et le paramètre d'homotopie* $q \in [0,1]$ *qui vérifie :*

$$\begin{aligned} B(\phi, x, t, 0) &= 0, \\ B(\phi, x, t, 1) &= \gamma(x,t) N(\phi) \end{aligned}$$

où $\gamma(x,t) \neq 0$ *,si*

$$\phi = \sum_{m=0}^{+\infty} u_m(x,t) q^m$$

est la série d'homotopie de Maclaurin de l'équation de déformation d'ordre zéro,

$$[1-\alpha(x,t,q)]L(\phi-u_0)=B(\phi,x,t,q) \tag{3.43}$$

Alors l'équation de déformation d'ordre m correspondante est :

$$L[u_m(x,t)]-\sum_{n=1}^{m-1}\alpha_n(x,t)L[u_{m-n}(x,t)]=D_m\left[B\left(\phi,x,t,q\right)\right] \tag{3.44}$$

où D_m est définie par (3.1) .

Preuve. Selon le théorème 5, on a :

$$D_m\left\{\left[1-\alpha\left(x,t,q\right)\right]L\left(\phi-u_0\right)\right\}=D_m\left[B\left(\phi,x,t,q\right)\right]$$

D'après lemme 3 on a :

$$D_m\left\{\left[1-\alpha\left(x,t,q\right)\right]L\left(\phi-u_0\right)\right\}=L\left(u_m\right)-\sum_{n=1}^{m-1}\alpha_n\left(x,t\right)L\left(u_{m-n}\right)$$

Ainsi, l'équation de déformation d'ordre m correspondant est :

$$L\left[u_m\left(x,t\right)\right]-\sum_{n=1}^{m-1}\alpha_n\left(x,t\right)L\left[u_{m-n}\left(x,t\right)\right]=D_m\left[B\left(\phi,x,t,q\right)\right].$$

■

Les équations de déformation d'ordre zéro (3.33), (3.35), (3.37), (3.39), (3.41) et (3.43) sont de plus et plus générales comme les équations de déformation d'ordre supérieur correspondant aux (3.34), (3.36), (3.38), (3.40), (3.42) et (3.44).
Parce que nous avons très grande liberté pour construire l'équation de déformation d'ordre zéro dans la définition générale.
Les équations de déformation d'ordre supérieur ont des propriétés communes :

1. Les équations de déformation d'ordre supérieur sont toujours linéaires par rapport à l'inconnu u_n,
2. Les conditions sur le côté gauche de l'équation de déformation d'ordre supérieur

sont assez similaires i.e.

$$L\left(u_m - \chi_m u_{m-1}\right) \quad \text{si } \alpha(q) = q,$$

ou

$$L\left(u_m - \sum_{n=1}^{m-1} \alpha_{m-n} u_n\right) \quad \text{si} \quad \alpha\left(q\right) = \sum_{k=1}^{+\infty} \alpha_k q^k,$$

où α_k est constante, ou :

$$L\left(u_m\right) - \sum_{k=1}^{m-1} \alpha_{m-k}\left(x,t\right) L\left(u_k\right) \qquad \text{si} \quad \alpha\left(x,t,q\right) = \sum_{k=1}^{+\infty} \alpha_k\left(x,t\right) q^k,$$

où $\alpha_k\left(x,t\right)$ est dépendant du x et t. Tous sont complètement déterminés par l'opérateur auxiliaire linéaire L et la fonction de déformation α.

Notez que nous avons une grande liberté de choisir l'opérateur auxiliaire linéaire L et la fonction de déformation $\alpha\left(q\right)$ ou $\alpha\left(x,t,q\right)$.

3. Pour l'équation de déformation d'ordre m : $u_0, u_1, u_2, ..., u_{m-1}$ sont connus et donc le terme droite de l'équation de déformation d'ordre supérieur est considéré comme connu .

   Donc, il est souvent pratique de résoudre successivement l'équation de déformation d'ordre supérieur linéaire en particulier à l'aide de système de calcul formel comme Maple et Mathématica .

4. Étant donné le type d'une équation de déformation d'ordre zéro, alors l'équation de déformation d'ordre supérieur correspondante est obtenue directement à l'aide des théorèmes prouvés ci-dessus.

   Généralement, il est nécessaire de calculer la dérivée d'homotopie $D_m\left[N\left(\phi\right)\right]$ Comme on l'a dans [1] pour toute fonction C^∞ donnée $f\left(\phi\right)$ où ϕ est une série de Maclaurin d'homotopie, nous pouvons toujours obtenir sa dérivée d'homotopie d'ordre m : $D_m\left[f\left(\phi\right)\right]$ à l'aide des théorèmes prouvée dans la section précédente notamment par la formule de récurrence (3.22). De même pour toute équation non linéaire donnée $N\left(u\right) = 0$, nous pouvons toujours obtenir le terme $D_m\left[N\left(\phi\right)\right]$ peu importe la complexité de l'opérateur non linéaire N. Cela indique l'importance de l'opérateur d'homotopie de dérivée D_m défini par (3.1), et révèle la raison pour laquelle les

théorèmes sur D_m sont prouvés dans la section précédente.
Notez que, comme montré dans [1] en utilisant la très grande liberté de construction l'homotopie d'une équation nous pouvons exprimer un paramètre physique inconnu comme une série de Maclaurin d'homotopie dans l'équation de déformation d'ordre zéro.
Donc tous les théorèmes ci-dessus sont valables quand nous remplaçons $D_k\left[N\left(\phi\right)\right]$ par $D_k\left[N\left(\phi,\Omega,q\right)\right]$ où $q\in[0,1]$ est le paramètre d'homotopie et

$$\Omega=\sum_{k=0}^{+\infty}\omega_k q^k$$

est une série de Maclaurin d'homotopie d'un paramètre physique ω.
En outre, nous pouvons traiter des conditions initiales ou aux limites d'une équation non linéaire d'une manière similaire à celle mentionnée ci-dessus.

3.3 Exemples :

Ici, nous utilisons quelques exemples simples pour montrer comment appliquer les théorèmes ci-dessus pour déduire les équations de déformation d'ordre supérieur de problèmes non linéaires.

Example 3.3.1 ***Considérons un problème de transfert de chaleur non linéaire (Abbasbandy, 2006)****[5]*

$$u+(1+\varepsilon u)u'=0,\qquad u(0)=1$$

Posons $L(u)=u'+u$ *, et*

$$N\left[\phi(t,q)\right]=\left(1+\varepsilon\phi\right)\phi'+\phi$$

où :

$$\phi=\sum_{k=0}^{+\infty}u_k(t)q^k$$

est une série de Maclaurin d'homotopie, on construit l'équation de déformation d'ordre zéro

$$[1-q]L\left[\phi(t,q)-u_0(t)\right]=cqN\left[\phi(t,q)\right]$$

Soumis à la condition initiale :

$$\phi(0,q) = 1$$

$u_0(t)$ *est la valeur initiale vérifiée la condition initiale* $u(0) = 1$.*D'après le théorème 3.2.1. Alors l'équation de déformation d'ordre* m *correspondant est :*

$$L[u_m(x,t) - \chi_m u_{m-1}(x,t)] = cD_{m-1}\left\{N[\phi(t,q)]\right\}$$

Soumis à la condition initiale :

$$u_m(0) = 0$$

D'après les théorèmes 1,2 et 3, on a :

$$\begin{aligned} D_{m-1}\left\{N\left[\phi(t,q)\right]\right\} &= D_{m-1}(\phi') + D_{m-1}(\phi) + \varepsilon D_{m-1}(\phi\phi') \\ &= u'_{m-1} + u_{m-1} + \varepsilon \sum_{k=0}^{m-1} u_{m-1-k}u'_k, \end{aligned}$$

La solution exprimée par la série d'homotopie :

$$u(t) = \sum_{k=0}^{+\infty} u_k(t),$$

est convergente pour tout paramètre physique $0 \leq \varepsilon < +\infty$ *si l'on choisit le paramètre de contrôle de la convergence* $c = -(1+\varepsilon)^{-1}$.*Pour plus de détails, voir(Abbasbandy, 2006)[5]*

Example 3.3.2 ***Considérons une équation d'oscillation non linéaire (Liao and Tan, 2007)***

$$\begin{gathered} u''(t) + \lambda u(t) + \varepsilon u^3(t) = 0 \\ u(0) = 1, \quad u'(0) = 0, \end{gathered}$$

où λ *et* ε *sont des paramètres physiques.*
Soit ω *désignant la fréquence inconnue de la solution périodique.Posons* $\tau = \omega t$*, l'équation ci-dessus devient :*

$$\begin{gathered} \gamma u''(\tau) + \lambda u(\tau) + \varepsilon u^3(\tau) = 0 \\ u(0) = 1, u'(0) = 0, \end{gathered}$$

où $\gamma = \omega^2$ *est un paramètre (physique) inconnu. Posons :*

$$N\left[\phi(\tau,q), \Gamma(q)\right] = \Gamma(q)\phi''(\tau,q) + \lambda\phi(\tau,q) + \varepsilon\phi^3(\tau,q)$$

et

$$\phi(\tau,q)=\sum_{k=0}^{+\infty}u_k(\tau)q^k \qquad\qquad \Gamma(q)=\sum_{k=0}^{+\infty}\gamma_k q^k,$$

sont des séries d'homotopie de Maclaurin, et prenons l'operateur auxiliaire lineaire comme

$$L(u)=u''+u,$$

On construit l'équation de déformation d'ordre zéro :

$$(1-q)L\left[\phi(\tau,q)-u_0(\tau)\right]=cqN\left[\phi(\tau,q),\Gamma(q)\right] \qquad q\in[0,1],$$

soumis à la condition initiale :

$$\phi(0,q)=1 \qquad \phi'(0,q)=0,$$

où $u_0(t)$ est la valeur initiale satisfaisant aux conditions initiales.Selon le théorème 15, l'équation de déformation d'ordre m correspondant est :

$$L[u_m(\tau)-\chi_m u_{m-1}(\tau)]=cD_{m-1}\left\{N[\phi(\tau,q),\Gamma(q)]\right\}$$

Soumis aux conditions initiales :

$$u_m(0)=1 \qquad u'_m(0)=0,$$

D'aprés les théorèmes 1, 3 et 6, on a :

$$\begin{aligned} D_{m-1}\left\{N\left[\phi(\tau,q),\Gamma(q)\right]\right\} &= D_{m-1}(\Gamma\phi'')+\lambda D_{m-1}(\phi)+\varepsilon D_{m-1}(\phi^3)\\ &= \sum_{k=0}^{m-1}\gamma_{m-1-k}u_k''+\lambda u_{m-1}+\varepsilon\sum_{k=0}^{m-1}u_{m-1-k}\sum_{i=0}^{k}u_{k-i}u_i, \end{aligned}$$

La série d'homotopie de la solution correspondante est donnée par :

$$u(t)=\sum_{k=0}^{+\infty}u_k(\omega t) \qquad\qquad \omega=\left(\sum_{k=0}^{+\infty}\gamma_k\right)^{\frac{1}{2}},$$

sont convergentes si le paramètre de contrôle de la convergence c est choisi correctement. Par exemple, lorsque $\lambda=0$, les série de solutions d'homotopie sont convergentes pour tout paramètre physique $0\le\varepsilon<+\infty$, en utilisant $c=-(1+\varepsilon)^{-1}$.Pour plus de détails voir chapitre 2 de [1].

Example 3.3.3 ***Considérons une équation différentielle non linéaire avec des coefficients de variables***

$$A(x)u''(x) + A'(x)u'(x) + \gamma \sin u(x) = 0,$$
$$u'(0) = 0, u'(\pi) = 0, u(0) = a,$$

où $A(x)$ est une fonction donnée,γ est une valeur propre inconnue, a est un constant donné.
Posons $u_0(x) = a\cos x$, $L(u) = u'' + u$ et :

$$N(\phi, \Gamma, q) = A(x)\phi''(x, q) + A'(x)\phi'(x, q) + \Gamma(q)\frac{\sin(q\phi)}{q},$$

on construit l'équation de déformation d'ordre zéro :

$$(1-q)L[\phi - u_0] = cqN(\phi, \Gamma, q) \qquad q \in [0, 1],$$
$$\phi'(0, q) = 0, \quad \phi'(\pi, q) = 0, \quad \phi(0, q) = a,$$

Selon le théorème 3.2.1, l'équation de déformation d'ordre m correspondante est :

$$L[u_m(\tau) - \chi_m u_{m-1}(\tau)] = cD_{m-1}\{N(\phi, \Gamma, q)\}$$

Soumis aux conditions initiales :

$$u_m(0) = a \qquad u'_m(\pi) = 0 \qquad u'_m(0) = 0,$$

D'après les théorèmes 1, 2 et 3, on a :

$$\begin{aligned} D_{m-1}[N(\phi, \Gamma, q)] &= D_{m-1}[A(x)\phi''(x, q)] + D_{m-1}[A'(x)\phi'(x, q)] + D_{m-1}\left[\Gamma(q)\tfrac{\sin(q\phi)}{q}\right] \\ &= A(x)u''_{m-1}(x) + A'(x)u'_{m-1}(x) + \sum_{k=0}^{m-1}\gamma_{m-1-k}D_k\left[\tfrac{\sin(q\phi)}{q}\right], \end{aligned}$$

où le terme $D_k[\sin(q\phi)/q]$ est donné par (3.31), qui est obtenu par le théorème 13. Pour plus de détails voir Liao (2009b)[8].

3.4 Théorèmes de convergence :

Dans le chapitre 2 de [1], deux théorèmes sur la convergence de la série d'homotopie d'une équation d'oscillation non linéaire sont donnés.
Qualitativement parlant, ces théorèmes ont significations générales.
Ici, deux théorèmes similaires sont prouvés en générale.

Théorème 3.4.1 *Soit γ est un paramètre physique inconnu. Soit $\alpha(x,t,q)$ une fonction de déformation, i.e. $\alpha(x,t,0)=0$ et $\alpha(x,t,1)=1$, et supposons que la série de Maclaurin :*

$$\alpha(x,t,q)=\sum_{k=1}^{+\infty}\alpha_k(x,t)q^k,$$

converge absolument pour $q=1$, telle que $\sum_{k=1}^{+\infty}\alpha_k(x,t)=1$, soit :

$$\sum_{k=0}^{+\infty}\beta_k(x,t)q^{k+1}$$

une série qui est absolument convergente si $q=1$ à une fonction non nulle $\beta(x,t)$, i.e.

$$\beta(x,t)=\sum_{k=0}^{+\infty}\beta_k(x,t)\neq 0,$$

où $\beta_0(x,t),\beta_1(x,t),...$dites des fonctions de contrôle de la convergence.
Soit :

$$\Gamma=\sum_{k=0}^{+\infty}\gamma_k q^k$$

une série de Maclaurin d'homotopie d'un paramètre physique inconnu γ,et

$$\phi=\sum_{k=0}^{+\infty}u_k(x,t)q^k,$$

est la série d'homotopie de Maclaurin de l'équation de déformation d'ordre zéro :

$$[1-\alpha(x,t,q)]\,L\,[\phi-u_0]=\left[\sum_{k=0}^{+\infty}\beta_k(x,t)q^{k+1}\right]N(\phi,\Gamma)\qquad q\in[0,1]\tag{3.45}$$

où u_m est régie par l'équation de déformation d'ordre m correspondante :

$$L\left[u_m(x,t)\right] - \sum_{n=1}^{m-1} \alpha_n(x,t) L\left[u_{m-1}(x,t)\right] = \sum_{k=1}^{m} \beta_{k-1}(x,t)\delta_{m-k}(x,t) \tag{3.46}$$

avec la définition :

$$\delta_k(x,t) = D_k\left[N\left(\phi,\Gamma\right)\right]$$

et la définition (3.1) de D_k. Si les séries d'homotopie :

$$u(x,t) = \sum_{k=0}^{+\infty} u_k(x,t), \qquad \gamma = \sum_{k=0}^{+\infty} \gamma_k,$$

convergent, et d'ailleurs $\sum_{k=0}^{+\infty} L\left[u_k(x,t)\right]$ aussi converge.
Alors :

$$\sum_{k=0}^{+\infty} \delta_k(x,t) = \sum_{k=0}^{+\infty} D_k\left[N\left(\phi,\Gamma\right)\right] = 0 \tag{3.47}$$

Preuve. *Selon (3.43), on a :*

$$\sum_{m=1}^{+\infty} \left\{ L\left[u_m\left(x,t\right)\right] - \sum_{n=1}^{m-1} \alpha_n\left(x,t\right) L\left[u_{m-n}\left(x,t\right)\right] \right\} = \sum_{m=1}^{+\infty} \sum_{k=1}^{m} \beta_{k-1}\left(x,t\right) \delta_{m-k}\left(x,t\right).$$

Puisque $\sum_{k=1}^{+\infty} \alpha_k\left(x,t\right)$ est absolument convergente et $\sum_{m=0}^{+\infty} L\left[u_m\left(x,t\right)\right]$ est convergente, à l'aide de théorème de Cauchy de produit on a :

$$\begin{aligned}
& \sum_{m=1}^{+\infty} \left\{ L\left[u_m\left(x,t\right)\right] - \sum_{n=1}^{m-1} \alpha_n\left(x,t\right) L\left[u_{m-n}\left(x,t\right)\right] \right\} \\
= & \sum_{m=1}^{+\infty} L\left[u_m\left(x,t\right)\right] - \sum_{m=1}^{+\infty} \sum_{n=1}^{m-1} \alpha_n\left(x,t\right) L\left[u_{m-n}\left(x,t\right)\right] \\
= & \sum_{m=1}^{+\infty} L\left[u_m\left(x,t\right)\right] - \sum_{n=1}^{+\infty} \sum_{m=n+1}^{+\infty} \alpha_n\left(x,t\right) L\left[u_{m-n}\left(x,t\right)\right] \\
= & \sum_{m=1}^{+\infty} L\left[u_m\left(x,t\right)\right] - \sum_{n=1}^{+\infty} \sum_{k=1}^{+\infty} \alpha_n\left(x,t\right) L\left[u_k\left(x,t\right)\right] \\
= & \sum_{m=1}^{+\infty} L\left[u_m\left(x,t\right)\right] - \left[\sum_{n=1}^{+\infty} \alpha_n\left(x,t\right)\right] \left\{ \sum_{k=1}^{+\infty} L\left[u_k\left(x,t\right)\right] \right\} \\
= & \left[1 - \sum_{n=1}^{+\infty} \alpha_n\left(x,t\right)\right] \left\{ \sum_{m=1}^{+\infty} L\left[u_k\left(x,t\right)\right] \right\},
\end{aligned}$$

ce qui donne à cause de $\sum_{n=1}^{+\infty} \alpha_n(x,t) = 1$ *:*

$$\sum_{m=1}^{+\infty} \left\{ L\left[u_m(x,t)\right] - \sum_{n=1}^{m-1} \alpha_n(x,t)\, L\left[u_{m-n}(x,t)\right] \right\}$$

De même, on a aussi $\sum_{k=1}^{+\infty} \beta_{k-1}(x,t)$ *est absolument convergente d'après le théorème de Cauchy de produit, on obtient :*

$$\begin{aligned}
& \sum_{m=1}^{+\infty} \sum_{k=1}^{m} \beta_{k-1}(x,t)\, \delta_{m-k}(x,t) \\
= & \sum_{k=1}^{+\infty} \sum_{m=k}^{+\infty} \beta_{k-1}(x,t)\, \delta_{m-k}(x,t) \\
= & \sum_{k=1}^{+\infty} \sum_{n=0}^{+\infty} \beta_{k-1}(x,t)\, \delta_n(x,t) \\
= & \left[\sum_{k=1}^{+\infty} \beta_{k-1}(x,t)\right] \left[\sum_{n=0}^{+\infty} \delta_n(x,t)\right] \\
= & \left[\sum_{m=0}^{+\infty} \beta_m(x,t)\right] \left[\sum_{n=0}^{+\infty} \delta_n(x,t)\right].
\end{aligned}$$

d'où :

$$\left[\sum_{k=1}^{+\infty} \beta_{k-1}(x,t)\right] \left[\sum_{n=0}^{+\infty} \delta_n(x,t)\right] = 0,$$

$$\sum_{m=0}^{+\infty} \beta_m(x,t) \neq 0 \Rightarrow \sum_{n=0}^{+\infty} \delta_n(x,t) = 0$$

i.e.

$$\sum_{n=0}^{+\infty} \left[D_n N(\phi, \Gamma)\right] = 0.$$

■

Théorème 3.4.2 *Soient* $\varphi(x,t,q)$ *et* $\Gamma(q)$ *représentent la solution de l'équation de déformation d'ordre zéro (3.42), dont les séries d'homotopie de Maclaurin :*

$$\phi = \sum_{m=0}^{+\infty} u_m(x,t) q^m, \qquad \gamma = \sum_{k=0}^{+\infty} \gamma_m q^m,$$

Supposons que les séries d'homotopie :

$$u(x,t) = \sum_{m=0}^{+\infty} u_m(x,t) \qquad \gamma = \sum_{m=0}^{+\infty} \gamma_m,$$

convergent, et d'ailleurs $\sum_{m=0}^{+\infty} L\left[u_m(x,t)\right]$ *aussi converge, de sorte que le théorème 21 a vérifié. Si* $N(\phi,\Gamma)$ *est analytique pour* $q \in [0,1]$.
Alors :

$$\sum_{m=0}^{+\infty} u_m(x,t) \quad et \quad \sum_{m=0}^{+\infty} \gamma_m$$

vérifient l'équation $N(u,\gamma) = 0$.

Preuve. *Depuis les deux séries d'homotopie :*

$$u(x,t) = \sum_{m=0}^{+\infty} u_m(x,t) \qquad \gamma = \sum_{m=0}^{+\infty} \gamma_m,$$

convergent, et $\sum_{m=0}^{+\infty} L\left[u_m\left(x,t\right)\right]$ *aussi converge d'après le théorème 21 on a*

$$\sum_{n=0}^{+\infty} \delta_n\left(x,t\right) = \sum_{n=0}^{+\infty} D_n\left[N\left(\phi,\Gamma\right)\right] = 0.$$

On remarque que

$$D_0\left[N\left(\phi,\Gamma\right)\right] = N\left(u_0,\gamma_0\right)$$

est le résidu de l'équation originale régissant pour les approximations initiales u_0 *et* γ_0, *donc* $N\left(\phi,\Gamma\right)$ *peut être considéré comme le résidu de l'équation gouverner pour* $q \in [0,1]$
Selon le théorème 10 sa série de Maclaurin d'homotopie est :

$$N\left(\phi,\Gamma\right) = \sum_{k=0}^{+\infty} \delta_k\left(x,t\right) q^k$$

il en resulte :

$$N\left(\sum_{m=0}^{+\infty} u_m q^m, \sum_{n=0}^{+\infty} \gamma_n\right) = \sum_{k=0}^{+\infty} \delta_k\left(x;t\right) q^k$$

Pour $q = 1$ *et on utilisant* $\sum_{n=0}^{+\infty} \delta_n(x,t) = 0$ *on obtient :*

$$N\left(\sum_{m=0}^{+\infty} u_m, \sum_{n=0}^{+\infty} \gamma_n\right) = \sum_{k=0}^{+\infty} \delta_k(x,t) = 0.$$

donc, les deux séries d'homotopie convergentes satisfont l'équation d'origine. ■

3.5 Expression de la solution :

Pour une équation non linéaire donnée, la clé de la HAM est de construire une bonne équation de déformation d'ordre zéro par le choix d'une approximation (valeur) initiale appropriée u_0, et un opérateur linéaire auxiliaire approprié L.
Comme indiqué ci-dessus, nous avons extrêmement large de liberté pour choisir l'approximation initiale u_0, et l'opérateur auxiliaire linéaire L : ce type de liberté fantastique qui différé HAM à d'autres techniques analytiques.
Cependant, toute chose ayant deux faces, la liberté très grande de construction de l'équation de déformation d'ordre zéro est donc difficile d'appliquation lors de l'apprentissage de HAM.
Ainsi, pour des applications multiples de la HAM en science et ingénierie, nous avons besoin de règles pour le choix de l'approximation initiale u_0 et l'opérateur linéaire auxiliaire L. Il est bien connu que le point de départ de techniques de perturbation est les paramètres physiques petit ou grand i.e. quantité de perturbation.
Cependant, HAM est indépendante des grands ou petits paramètres physiques (c'est l'un des avantages de HAM). Donc, nous avons besoin d'un nouveau mais différent point de départ de la HAM.
En essence, pour approximer analytiquement la fonction $f(t)$ dans un domaine Ω est d'exprimer par un ensemble complet des fonctions de base, par exemple :

$$f(t) = \sum_{k=1}^{+\infty} a_k e_k(t),$$

où $e_k(t)$ désigne la fonction de base et a_k est un coefficient, le choix de fonctions de base appropriée est déterminé non seulement par la propriété de $f(t)$, mais aussi par le domaine.

Par exemple, si $f(t)$ est périodique, alors les fonctions de base $e_k(t)$ sont périodiques . Par ailleurs, selon le théorème de **Weierstrass**, pour toute $f(t)$ donnée en $C[a,b]$ donnée et pour tout $\varepsilon > 0$, il existe un polynôme P_n pour un certain n suffisamment grand tel que $\| f(t) - P_n(t) \| \leq \varepsilon$.
Par conséquent, pour une fonction $u(x,t)$ donnée, le point clé est de trouver un ensemble approprié de fonctions de base $e_k(x,t)$ pour s'adapter, c'est à dire

$$u(x,t) \sim \sum_{k=1}^{+\infty} a_k e_k(x,t)$$

où $e_k(x,t)$ représente la fonction de base.

Définition 3.5.1 *Soit $e_k(x,t)$ la fonction de base, $u(x,t)$ est la solution de l'équation non linéaire $N(u) = 0$.*
Alors :

$$u(x,t) = \sum_{k=1}^{+\infty} a_k e_k(x,t) \tag{3.48}$$

est appelé l'expression de la solution de $u(x,t)$, si

$$\lim_{m \to \infty} \| u(x,t) - \sum_{k=1}^{+\infty} a_k e_k(x,t) \| \to 0 \tag{3.49}$$

Notez que cette fonction peut être exprimée par des fonctions de base différentes.
Par exemple, une fonction continue arbitraire $f(t) \in C[1,1]$ a une meilleure approximation en série de **Chebyshev**, i.e. :

$$f(t) = \sum_{n=0}^{+\infty} b_n T_n(t)$$

où $T_n(t)$ est un polynôme de **Tchebychev** de degré n, défini par la relation (Mason et Handscomb, 2003) :

$$T_n(t) = \cos(n\theta)$$

avec la formule de récurrence :

$$\begin{aligned} T_0(t) &= 1, \qquad T_1(t) = t, \\ T_n(t) &= 2tT_{n-1}(t) - T_{n-2}, \qquad n = 2, 3, 4, ... \end{aligned}$$

En outre, l'expression de solution de $f \in C[a, b]$ peut être un polynôme, d'après le théorème de **Weierstrass**, ou une série de Fourier (Mason et Handscomb, 2003).
En fait, la recherche de l'expression de la solution d'une équation donnée $N(u) = 0$ est l'objectif de la résolution de l'équation.
Autrement dit, il est le point final pour nous. Toutefois, ce genre de point final est utilisé comme point de départ de HAM.
Etant donné une équation différentielle non linéaire $N(u) = 0$; on doit d'abord poser une question importante : quels types de fonctions de base peuvent être utilisés pour approximer la solution inconnue u?

Pour certains types d'équations, comme les équations avec des solutions continues définies dans un domaine fini, il est facile de répondre à cette question.
Cependant, la réponse à cette question n'est pas toujours évidente, surtout pour les nouveaux types d'équations dont les motifs de retour physique ne sont pas trés claires.
Dans ce cas, il est utile d'avoir certaines propriétés asymptotiques des solutions, le plus sera le mieux.
Ces propriétés asymptotiques nous fournissent souvent des informations précieuses sur l'expression de la solution d'une équation non linéaire.

Quelques règles sont données ici pour le choix de l'expression de la solution d'une équation donnée $N(u) = 0$.

1. **Equations définies dans un domaine fini :**
   - Les polynômes et les séries de **Fourier** peuvent être toujours utilisés comme l'expression de la solution,
   - la série de **Tchebyshev** donne la meilleure approximation des solutions continues arbitraires de $N(u) = 0$.
2. **Equations définies dans un domaine infini :**
   - Les fonctions de base périodiques doivent être utilisées, si la solution est périodique,
   - Les fonctions de base non périodiques doivent être utilisées, si la solution n'est pas

périodique,

– L'expression de la solution vérifie aussi des propriétés asymptotiques de solution que possible, si la solution n'est pas périodique.

On notera que les règles ci-dessus ne sont pas absolues, en particulier en cas où la solution est inconnue non périodique et définie dans un domaine infini.
Heureusement, comme mentionné précédemment, on peut exprimer la solution unique de l'équation $N(u) = 0$ par différents types de fonctions de base.
Donc, même si on a un peu d'informations sur l'expression de la solution d'une équation donnée $N(u) = 0$, on peut toujours deviner certaines formes de son expression de la solution, puis vérifier si ces suppositions sont correctes ou non.
Le concept d'expressions de la solution des équations non linéaires est facile à comprendre pour les mathématiciens appliqués qui ont une connaissance physique riche. Par exemple, il est bien connu que les oscillations d'un système dynamique conservateur sont pour la plupart périodiques, bien que sa période et l'amplitude des oscillations ne sont pas connues.
En outre, il est bien connu que les flux laminaires visqueux une connaissance varient énormément à proximité d'une frontière solide (c'est à dire le débit de la couche limite) mais ont tendance à l'écoulement uniforme de façon exponentielle à l'infini.
Ainsi, les expressions de la solution des équations différentielles non linéaires lié aux flux laminaires visqueux en mécanique des fluides contiennent les fonctions exponentielles qui exponentielle tendent vers zéro à l'infini.
En outre, toutes les vagues de déplacement périodiques peuvent être exprimé par des fonctions de base périodiques.

Toutes ces connaissances physiques, si possible, sont plutôt utilisées pour le choix de l'expression de la solution d'une équation non linéaire donnée.
Le concept de l'expression de la solution mentionnée ci-dessus est important dans le cadre de HAM, car il est le point de démarrage pour que nous choisissons l'approximation initiale et l'opérateur auxiliaire linéaire L.

3.5.1 Choix de l'approximation initiale :

Supposons qu'une expression de la solution (3.45) est choisie pour une équation donnée $N(u) = 0$.

Notre objectif est de trouver une approximation initiale u_0 correcte et un bon opérateur auxiliaire linéaire L tel que la série d'homotopie correspondant convergent.
De toute évidence, l'approximation initiale u_0 doit obéir à l'expression de la solution (3.45). Puisque nous avons la liberté de choisir l'approximation initiale, on pourrait choisir de ce type de l'approximation initial :

$$u_0(x,t) = \sum_{k=1}^{n_0} \bar{a}_k e_k(x,y), \tag{3.50}$$

ou $e_k(x,t)$ est la fonction de base, $\bar{a}_k$ des constantes inconnue, et n_0 est égal ou supérieur au nombre de conditions initiales / aux limites linéaires désignés par κ.
Puis, l'application de l'approximation initiale ci-dessus pour satisfaire les conditions limite/initiale linéaire, nous avons $n_0 - \kappa$ coefficients inconnus laissés.
Ainsi, lorsque $n_0 = \kappa$, alors tous les coefficients de l'approximation initiale sont connus de sorte qu'il est complètement déterminé.
Toutefois, lorsque $n_1 = n_0 - \kappa > 0$, nous avons n_1 coefficients inconnus, notés par $b_1, b_2, ..., b_{n_1}$, pour obtenir une approximation initiale optimale, nous définissons le carré résiduel de l'équation qui régit par :

$$E_0(b_1, b_2, ..., b_{n_1}) = \int_\Omega \left[N(u_0)\right]^2 d\Omega \tag{3.51}$$

Il est bien connu que le minimum du carré résiduel $E_0(b_1, b_2, ..., b_{n_1})$ est déterminé par un ensemble d'équations algébriques non linéaires

$$\frac{\partial E_0}{\partial b_k} = 0, \qquad 1 \leq \kappa \leq n_1,$$

dont la solution donne les valeurs optimales $b_1^*, b_2^*, ..., b_{n_1}^*$ des coefficients inconnus.
De cette façon, on obtient une approximation optimale u_0.
Evidemment, le nombre plus grand est n_1 , c'est-à-dire il ya des coefficients (en plus) inconnus, l'approximation initiale optimale est meilleure, mais il a besoin de plus de temps **CPU** pour résoudre l'ensemble des équations algébriques non linéaires plus compliquées.

En cas de n_1 très grand , l'ensemble des équations algébriques non linéaires deviennent difficiles à résoudre, et il devient la méthode des moindres carrés.
Ainsi, un équilibre est nécessaire.
Dans la pratique, il est souvent suggéré d'avoir un coefficient inconnu dans l'approxima-

tion initiale (ie $n_1 = 1$), dont la valeur optimale est déterminée par le minimum de valeur de carré résiduel défini par (3.48).

La plupart du temps, ce genre d'approximations initiales optimales avec un coefficient optimal sont assez bonnes, comme illustré dans le chapitre 2 dans [1].
Donc, tant que l'expression d'une solution d'une équation donnée $N(u) = 0$ est connue, il est direct d'obtenir une approximation initiale optimale de u_0 de manière mentionnée ci-dessus.

3.5.2 Choix de l'opérateur linéaire auxiliaire :

Pour obéir à l'expression de la solution (3.45) d'une équation donnée $N(u) = 0$, u_0 la valeur initiale et l'opérateur auxiliaire linéaire L doit être choisi de telle sorte que la solution um de l'équation de déformation d'ordre élevé existe et obéit (3.45) , et d'ailleurs la série d'homotopie :

$$u_0 + \sum_{k=1}^{+\infty} u_k$$

converge.
Le choix de l'opérateur auxiliaire linéaire L est principalement déterminé par l'expression de la solution (3.45), mais parfois aussi par les conditions aux limites / initial.
Par exemple, si la solution $u(t)$ est une fonction périodique avec une fréquence de ω connue, alors l'opérateur linéaire auxiliaire doit être

$$L(u) = u'' + \omega^2 u,$$

où le prime dénote la différenciation par rapport à t.
Si la solution $u(t)$ est une fonction périodique de fréquence inconnue ω, nous utilisons d'abord la transformation $\tau = \omega t$ et choisir un tel opérateur linéaire auxiliaire :

$$L(u) = u'' + u$$

ou le prime dénote la différenciation par rapport à τ , si l'expression de la solution de $u(t)$ est polynomiale, alors on peut tout simplement choisir l'opérateur linéaire auxiliaire

$$L(u) = \frac{d^\sigma u}{dt^\sigma} \tag{3.52}$$

ou l'entier $\sigma > 0$ représente l'ordre le plus élevé de dérivé de $N(u) = 0$.

En général, soit le nombre entier $\sigma > 0$ représentent le degré le plus élevé de la dérivée d'une équation différentielle ordinaire $N[u(t)] = 0$, $u_m(t)$ une solution spéciale de l'équation de déformation correspondant d'ordre m, respectivement.
Soit :

$$L(u) = u^{(\sigma')} + \sum_{k=1}^{\sigma'} \mu_k(t) u^{(\sigma'-k)} \tag{3.53}$$

désigne l'opérateur linéaire auxiliaire inconnu, ou $u^{(k)}$ désigne la dérivée d'ordre k de $u(t)$, est le plus élevé afin de dériver, et le coefficient inconnu μ_i est déterminé plus tard.
Soit

$$u_m(t) = u_m^*(t) + \sum_{k=1}^{\sigma_1'} A_k e_k(t) + \sum_{k=1}^{\sigma_2'} B_k \bar{e}_k(t) \tag{3.54}$$

désigne la solution commune de l'équation de déformation d'ordre m, où $\sigma_1' + \sigma_2' = \sigma$, A_k, B_k sont des coefficients inconnus, $e_k(t)$ sont les fonctions de base, mais $\bar{e}_k(t)$ ne sont pas, i.e :

$$\bar{e}_k(t) \notin \{e_1(t), e_2(t), e_3(t), ...\}$$

Evidemment, pour obéir à l'expression de la solution, il faut tenir :

$$B_k = 0, \qquad 1 \leq k \leq \sigma_2' \tag{3.55}$$

Ainsi, la solution commune est :

$$u_m(t) = u_m^*(t) + \sum_{k=1}^{\sigma_1'} A_k e_k(t) \tag{3.56}$$

L'opérateur auxiliaire linéaire inconnu L doit être choisi de façon que les coefficients inconnus A_k de l'expression ci-dessus sont uniquement déterminés par le biais de toutes conditions limites / initiales liées de l'équation de déformation d'ordre m , disons, la solution $u_m(t)$ existe uniquement et obéit à l'expression de la solution ailleurs.

Si cela est pas vrai, alors nous avons dû changer le nombre σ_1' jusqu'à ce que satisfait , si elle devient réalité, alors les coefficients inconnus $\mu_k(1 \leq k \leq 0)$ de l'opérateur linéaire auxiliaire correspondant L est déterminé en résolvant l'ensemble d'équations algébriques

linéaires :

$$L\left[e_k\left(t\right)\right]=0 \qquad 1\leq k\leq \sigma_1',$$
$$L\left[\bar{e}_k\left(t\right)\right]=0 \qquad 1\leq k\leq \sigma_2'.$$

De cette manière, on obtient l'opérateur auxiliaire linéaire L défini par (3.49).
Par exemple, s'il vous plait se référer à **Liao** et **Tan** (2007)[X].
Notez que, bien que l'on peut choisir $\sigma=\sigma'$ dans la plupart des cas, ce n'est cependant pas absolument nécessaire [1], principalement parce que nous avons extrémement large liberté de choisir l'opérateur linéaire auxiliaire L.
De même, la méthode ci-dessus peut être utilisée pour trouver un opérateur linéaire auxiliaire bon pour les équations aux dérivées partielles non linéaires, aussi.
Il est intéressant que l'équation de déformation d'ordre zéro :

$$\left(1-q\right)L\left[\phi-u_0\right]=cqN\left[\phi\right] \qquad c\neq 0, q\in\left[0,1\right]$$

peut être réecrite sous la forme :

$$\left(1-q\right)\bar{L}\left[\phi-u_0\right]=qN\left[\phi\right]$$

où $\bar{L}=L/c$, et c est appellé le paramètre de contrôle de convergence , Donc en substance, le choix du paramètre de contrôle de convergence c est une partie du choix de l'opérateur auxiliaire linéaire.
Par conséquent, le paramètre contrôle de convergence optimal correspond à l'opérateur linéaire auxiliaire optimal ! Soit $\parallel L^{-1}\parallel$ désignent la norme de l'opérateur inverse L^{-1}.
Ensuite, nous avons $\parallel \bar{L}^{-1}\parallel = \mid c\mid\parallel L^{-1}\parallel$.
Ainsi, nous pouvons ajuster la norme de L^{-1} ce qui est la raison essentielle pour laquelle nous ne pouvons garantir la convergence de série d'homotopie au moyen de choisir un bon paramètre contrôle de convergence c .
De même, plus l'équation de déformation d'ordre zéro :

$$\left(1-q\right)L\left[\phi-u_0\right]=q\left(\sum_{k=0}^{+\infty}c_kq^k\right)N\left[\phi\right] \qquad c\neq 0, q\in\left[0,1\right]$$

peut être réécrite sous la forme "de base" :

$$\left(1-q\right)\tilde{L}\left[\phi-u_0\right]=qN\left[\phi\right]$$

ou c_k sont des paramètres contrôle de convergence et :

$$\tilde{L} = \left(\sum_{k=0}^{+\infty} c_k q^k \right)^{-1} L,$$

Par conséquent, le choix des paramètres de contrôle de convergence c_k est essentiellement une partie de choisir l'opérateur linéaire auxiliaire L.
Evidemment, la norme $\| \tilde{L}^{-1} \|$ est déterminée par les paramètres de contrôle de convergence c_k, et les paramètres de contrôle de convergence optimaux c_k correspondent à l'opérateur linéaire auxiliaire optimal.
Autrement dit, nous choisissons l'opérateur auxiliaire linéaire en deux étapes : l'opérateur auxiliaire linéaire L "de base" est d'abord choisie en fonction selon que l'on appelle l'expression de solution , puis l'opérateur linéaire auxiliaire est modifié par le choix des paramètres contrôle de convergence optimaux .
Par conséquent, même si la "base" de l'opérateur auxiliaire linéaire L n'est pas parfaite, nous pouvons toujours garantir la convergence des séries d'homotopie en choisissant les paramètres de contrôle de convergence optimaux c_k.
En résumé, l'expression de la solution est un concept important de HAM, qui nous fournit un point de départ pour choisir l'approximation initiale et la "base" de l'opérateur auxiliaire linéaire L.

Chapitre 4

Programme Mathematica BVPh2.0 et exemples pratiques

4.1 Introduction :

La méthode d'analyse d'homotopie (HAM) a été proposé par Liao pour obtenir des approximations analytiques des problèmes fortement non linéaires.
La HAM a quelques avantages par rapport aux méthodes d'approximations analytiques traditionnelles. Tout d'abord, contrairement aux techniques de perturbation, la HAM est indépendant de petits grands paramètres physiques, donc elle est valable dans les cas plus généraux.
En outre, différente de toutes les autres techniques analytiques, la HAM nous offre un moyen pratique pour garantir la convergence de série de solution.
De plus, la HAM offre une grande liberté du choix de l'estimation initiale et l'équation de type sous-problèmes linéaires.
Il se trouve que beaucoup de problèmes aux limites non linéaires en sciences, en ingénierie et en finance peuvent être résolus facilement par la HAM en cas où l'intervalle est fini ou pas.
Basé sur la HAM. le package Mathematica BVPh1.0 utilisé pour les problèmes aux limites non-linéaires et de valeurs propres avec singularité (et/ou) conditions aux limites multipoint a été écrit par Liao en mai 2012. Comme illustré par Liao [1], le BVPh 1.0 est valable pour beaucoup de problèmes aux limites BVP non linéaires .
Cependant, le BVPh 1.0 est valable pour les équations différentielles ordinaires uniques

(ODE), il ne peut pas résoudre des systèmes des équations différentielles. Maintenant, la nouvelle version de `BVPh`, le `BVPh 2.0` peut faire face à de nombreux systèmes des équations différentielles ordinaires couplées (EDO) définies dans des intervalles finis ou semi-infinis. En outre,de nouveaux algorithmes sont utilisés dans certains modules de `BVPh 2,0`.

Par conséquent, `BVPh 2.0` est beaucoup plus rapide que `BVPh 1.0` dans la plupart des cas. Dans ce chapitre, nous allons montrer l'utilisation de `BVPh 2.0` pour résoudre différents types de systèmes d'équations différentielles ordinaires, y compris un système d'équations différentielles couplées dans un intervalle fini, un système d'équations différentielles couplées dans un intervalle semi-infini, un système d'équations différentielles couplées à la propriété algébrique à l'infini, un système d'équations différentielles avec un paramètre inconnu à déterminer et un système d'équations différentielles sur des intervalles différents.

Ce chapitre est organisé comme suit :

Nous avons expliqué (montrons) comment charger le paquet `BVPh 2.0`.

Puis, un exemple illustratif est pris pour l'utilisation de `BVPh 2.0` en détail.

Et nous prenons un coup d'œil sur les modules, d'entrée, de sortie et paramètres de contrôle dans le paquet.

D'autres exemples sont donnés dans [3] pour expliquer l'utilisation du package.

A la fin, certaines conclusions sont données.

Il convient de souligner que tous les exemples de ce chapitre ont été résolus analytiquement (par la `HAM`) et numériquement avant. Le paquet `BVPh 2.0` donne des résultats acceptables lorsque les mêmes paramètres physiques sont donnés.

Le paquet `BVPh 2.0` est développé sur **Mathematica 7.0**. Comme une nouvelle fonction de **Mathematica 7.0** sont utilisés, nous vous recommandons fortement d'utiliser le `BVPh 2.0` dans **Mathematica 7.0** ou version supérieure.

4.2 Modules clés

`BVPh :` Le module `BVPh[K_,m_]` donne les approximations d'homotopie d'ordre k à m d'un système des équations différentielles ordinaires (EDO) soumis à des conditions aux limites.

Le système peut avoir un paramètre inconnu (lorsque `TypeEQ=2`) à déterminer ou ne pas avoir un paramètre inconnu (lorsque `TypeEQ=1`) à déterminer.

C'est le module de base.
Par exemple, `BVPh[1,10]` donne les approximations d'homotopie d'ordre $1ere - 10ème$.
D'autre part, `BVPh[11,15]` donne encore l'approximations d'homotopie d'ordre 11 jusqu'a 15 .
Pour les problèmes avec un paramètre inconnu, l'estimation initiale du paramètre inconnu est déterminé par une équation algébrique.
Ainsi, s'il n'y a plus d'estimation initiale du paramètre inconnu, il est nécessaire de choisir l'un d'entre eux en introduisant un nombre entier, une telle 1 ou 2, correspondant à la première ou la deuxième estimation initiale , respectivement.

`iter` : Le module `iter[K_,m_,r_]` nous fournit des itération d'approximations d'homotopie d'ordre k à m à l'aide de la formule d'itération d'ordre r , il appelle au module de base BVPh.
Par exemple, `iter[1,10,3]` donne des itérations d'approximations homotopiques du premier au dixième ordre par la formule d'itération de $3ème$ ordre. En outre, `iter[11,20,3]` donne les itérations d'approximations d'homotopie du 11 au 20 .
L'itération s'arrête lorsque le carré résiduel du système est inférieur à une valeur critique de `ErrReq`, dont la valeur par défaut est 10^{-20}.

`GetErr` : Le module `GetErr[K_]` donne le carré résiduel de l'équation régissant de l'approximation d'homotopie d'ordre k obtenue par le module **BVPh**.
on remarque que `errer[i,k]` fournit le résidu de $i - ème$ de l'équation régissant à l'approximation d'homotopie d'ordre K obtenue par **BVPh**, et `ErrTotal[k]` donne le moyenne totale de carré résiduel du système à l'aprroximation d'homotopie d'ordre K acquise par **BVPh**.

`hp` : Le module `hp[f_,m_,n_]` donne l'approximation d'homotopie de pad'e `[m,n]` d'une liste d'approximations d'homotopie `f`, où `f[[i+1]]` désigne l'approximation d'homotopie d'ordre i de même équation .

`GetBC` : Le module `GetBC[i_,K_]` donne la condition aux limites "i" de l'équation de déformation d'ordre k .

4.3 Paramètres de contrôle :

`TypeEQ` : un paramètre de commande pour le type d'équations régissant :
`TypeEQ=1` correspond à un problème non linéaire sans un paramètre inconnu, `TypeEQ=2` correspond à un problème non linéaire avec un paramètre inconnu (appelée problème aux valeurs propres ou un problème de valeur propre).

`TypeL` : Un paramètre de contrôle pour le type de l'opérateur linéaire auxiliaire : `TypeL = 1` correspond à l'approximation polynômiale, et `TypeL=2` correspond à une approximation trigonométrique ou une approximation de base hybride, respectivement.

`ApproxQ` : Un paramètre de commande pour rapprochement (approximé) des solutions.
Lorsque `ApproxQ=1`, le terme de droite de toutes les équations de déformation d'ordre supérieur ont approximé par un polynôme de Chebyshev , ou par approximation en focntion de base hybride.
Lorsque `ApproxQ=0`, il n'y a pas ce type d'approximation.
Le `TypeL=2`, `ApproxQ=1` est valable seulement pour des problèmes sur un intervalle fini $z \in [0, a]$, où $a > 0$ est un constant.

`HYBRID` : Un paramètre de commande pour les fonctions de base hybrides.
Lorsque `HYBRID=1`, fonctions de base hybrides sont utilisés dans l'approximation. Lorsque `HYBRID=0`, fonctions trigonométriques sans polynômes sont utilisées dans l'approximation. Ce paramètre est généralement utilisé en conjonction avec `TypeBase`, et est valide uniquement si `TypeL=2` et `ApproxQ=1` pour les problèmes définis sur un intervalle fini $z \in [0, a]$.

`TypeBase` : Un paramètre de commande pour le type d'approximation en série de Fourier : `TypeBase=1` correspond à l'approximation de Fourier impaire, `TypeBase=2` correspond à l'approximation de Fourier paire.
Ce paramètre est généralement utilisé en conjonction avec `HYBRID`, et est valable seulement si `TypeL=2` et `ApproxQ=1` pour les problèmes définis dans un intervalle fini $z \in [0, a]$.

`Ntruncated` : Un paramètre de commande pour déterminer le nombre de termes tronquées utilisés pour approximer l'équation de déformation d'ordre supérieur .
les meilleures approximations est pour Ntruncated plus grand, mais le plus temps de **CPU**.

Il est valable seulement que si `ApproxQ=1`. La valeur par défaut est 10.

`NtermMax` : Un entier positif utilisé dans le module `truncated`, qui ignore tous les termes du polynôme dont l'ordre supérieur à `NtermMax`. La valeur par défaut est 90.

`ErrReq` : Une valeur critique de carré résiduele de l'équation régissant pour arrêter le calcul. La valeur par défaut est de 10^{-20}.

`NgetErr` : Un entier positif utilisé dans le module BVPh.
Le carré résiduel d'équations régissant est calculé lorsque l'ordre d'approximation est divisible par NgetErr. La valeur par défaut est 2.

`Nintegral` : Nombre de points discrets equidistant, qui sont utilisés pour calculer l'intégrale d'une fonction numériquement . Il est utilisé dans le module `int`. la valeur par défaut est 50.

`ComplexQ` : Un paramètre de contrôle pour des variables complexes.
`ComplexQ=1` correspond à l'existence des variables complexes dans les équations régissant avec des conditions aux limites.
`ComplexQ=0` correspond à l'absence des variables complexes. La valeur par défaut est 0.

`FLOAT` : Un paramètre de contrôle pour calculer point flottant.
Lorsque le flotteur = 1, nombres de points flottante sont utilisés dans le calcul.
Lorsque le flotteur = 0, les nombres rationnels sont utilisés dans le calcul.
La valeur par défaut est 1.

4.4 Input :

`NumEQ` : Le nombre d'équations régissant.

`f`$[$`i_`$, $`z_`$, \{$`u_`$, ...\}, $`lambda_`$]$: les "i" équations régissant avec ou sans paramètre inconnu, correspondant à `F`$_i[$`z`$, $`u`$, ...]$ ou `F`$_i[$`z`$, $`u`$, ..., \lambda]$ dans un intervalle fini $z \in [a, b]$ ou un intervalle infini $z \in [b, +\infty)$, où a et b sont bornées.
Notez que le paramètre formel `lambda` désigne le paramètre inconnu λ à déterminer,ou

la valeur propre λ pour les problèmes de valeurs propres, mais n'a pas de sens pour les problèmes sans un paramètre inconnu λ, ou des problèmes n'ont pas des valeurs propres.

`NumBC` : Le nombre de conditions aux limites.

`BC`$[k\_, z\_, \{u\_...\}]$: La condition aux limites correspondant à $\mathtt{B}_k[\mathtt{z}, \mathtt{u}, ...]$,où $1 \leq k \leq$`NumBC`. A noter que le symbole infinité représente ∞ dans les conditions aux limites.

`U[i,0]` : L'approximation initiale $u_{i,0}(z)$.

`c0[i]` : Le paramètre de contrôle de la convergence $c_{0,i}$, correspondant à l'équation *ième*.

`H[z_,i_]` : La fonction auxiliaire correspondant à l'équation *ième* gouverner.
La valeur par défaut est $H[i\_, z\_] := 1$.

`L[f_,i_]` : L'opérateur linéaire auxiliaire correspondant à l'équation *i-ème* gouverner.

`zL[i]` : Le point gauche de l'intervalle de la solution correspondant à l'équation régissant .
Notez que les intervalles des solutions ne sont pas nécessairement les même en particulier pour les problèmes d'écoulement à plusieurs couches.

`zR[i]` : Le point droit de l'intervalle de la solution correspondant à l'équation régissant .

`zLintegral[i]` : Le point gauche de l'intervalle intégrant pour calculer le carré résiduel de l'équation gouverner.
Lorsque le point gauche de l'intervalle de solution pour léquation régissant est un nombre fini, `zLintegral[i]` est automatiquement réglé sur `zL[i]`. Sinon, l'utilisateur doit spécifier la valeur de `zLintegral[i]`.

`zRintegral[i]` : Le point droit de l'intervalle intégrant pour calculer le carré résiduel de l'équation gouverner. Lorsque le point droit de l'intervalle de solution pour l'équation gouverner est un nombre fini, `zRintegral[i]` est automatiquement réglé sur `zR[i]`. Dans le cas contraire, l'utilisateur doit spécifier la valeur de `zRintegral[i]`.

4.5 Output (sortie) :

`U`$[\mathtt{i}, \mathtt{k}]$: L'approximation d'homotopie d'ordre K de la solution de l'*ième* équation régissant donnée par le module de base `BVPh` .

`V[i,k]` : L'itération de l'approximation d'homotopie d'ordre k de solution de l'équation *ième* régissant donnée par le module d'itération `iter`.

`Lambda[k]` : L'approximation d'homotopie d'ordre k de valeur propre λ où le paramètre inconnu λ donnée par le module de base `BVPh` .

`LAMBDA[k]` : L'itération d'approximation d'homotopie d'ordre k de la solution de l'équation *ième* régissant donnée par le module d'itération `iter`.

`error[i,k]` : Le résidu de l'équation *ième* régissant donné par l'approximation d'homotopie d'ordre k (obtenue par le module de base BVPh).

`Err[k]` : Une liste de carré résiduel de chaque équation régissant donné par l'approximation d'homotopie d'ordre k (obtenue par le module BVPh).

`ErrTotal[k]` : Le carré résiduel total pour chaque équation régissant, donnée par l'approximation d'homotopie d'ordre k (obtenu par le module BVPh).

`ERR[k]` : Une liste moyenne du carré résiduel de chaque équation régissant donné par l'itération d'approximation d'homotopie d'ordre k (obtenu par le Module d'itération `iter`).

`ERRTotal[k]` :Le carré résiduel total pour chaque équation régissant donnée par l'itération d'approximation d'homotopie d'ordre k (obtenu par le module d'itération `iter`).

4.6 variables globales :

Tous les paramètres de commande et les variables de sortie mentionnés ci-dessus sont globales. en plus de ceux-ci, les variables et les paramètres suivants sont également mondiaux.

z : La variable indépendante z.

`u[i,k]` : La solution de l'équation de déformation d'ordre k de l'équation *ième* gouverner .

`lambda[k]` : Une variable constante, correspondant à λ_k.

`delta[i,k]` : Une fonction dépend de z, correspondant au terme droit de $\delta_{i;k}(z)$ dans l'équation de déformation d'ordre supérieur.

`L[i,w]` : L'opérateur auxiliaire linéaire ième L_i appliquée aux w.

`Linv[i,f]` : L'opérateur inverse de L_i :L_i^{-1}, appliqué à f.

`sNum` : Un entier positif pour déterminer laquelle estimation initiale λ_0 est choisie quand il ya plusieurs solutions de λ_0.

4.7 Exemples :

4.7.1 1.Un système d'équations différentielles dans un intervalle fini :

Considérons un système d'équations différentielles couplées ODE's :

$$(1+K)f^{(4)} - \mathrm{Re}\, M f'' + 2\,\mathrm{Re}\, f f'' - K g'' = 0, \tag{4.1}$$

$$(1+\frac{K}{2})g'' - \mathrm{Re}\, K[2g - f''] + \mathrm{Re}[2fg' - f'g] = 0, \tag{4.2}$$

Soumis aux :

$$f(0) = 0, f(1) = 0, f'(1) = 0, f''(0) = 0, \tag{4.3}$$

$$g(0) = 0, g(1) = 0, \tag{4.4}$$

où K est le rapport des viscosités, Re est le nombre de Reynolds et M est le nombre de Hartman. **Hayat 9** [12] a résolu ce problème par la `HAM`.
Ici, nous résolvons ce problème en `BVPh 2.0`. Comme il ya deux équations différentielles en Système (5.1) - (5.2) sans un inconnue à déterminer, nous avons `NumEQ = 2` et `TypeEQ = 1`.
Le système est entré en tant que

```
TypeEQ = 1 ;
NumEQ = 2 ;
f[1,z_,{f_,g_},Lambda_] :=(1+K)*D[f,{z,4}]
-Rey*M*D[f,{z,2}]+2*Rey*f*D[f,{z,3}]-K*D[g,{z,2}] ;
f[2,z_,{f_,g_},Lambda_] :=(1+K/2)*D[g,{z,2}]
-Rey*K*(2*g-D[f,{z,2}])+Rey*(2*f*D[g,z]-D[f,z]*g) ;
```

Les conditions aux limites sont donnée par :

```
NumBC = 6 ;
BC[1,z_,{f_,g_}] :=f/.z->0 ;
BC[2,z_,{f_,g_}] :=f/.z->1 ;
BC[3,z_,{f_,g_}] :=(D[f, z]-1)/.z->1 ;
BC[4,z_,{f_,g_}] :=D[f,{z,2}]/.z->0 ;
```

```
BC[5,z_,{f_,g_}] :=g/.z->1 ;
BC[6,z_,{f_,g_}] :=g/.z->0 ;
```

Maintenant, nous allons entrer les intervalles de solution

```
zL[1]=0 ; zR[1]=1 ;
zL[2]=0 ; zR[2]=1 ;
```

Puisque tous les intervalles de solution sont dans un intervalles finis, nous n'avons pas à préciser l'intervalle intégrant pour calculer le carré résiduel .
Les estimations initiales sont choisies comme $f_0 = (z^3 - z)/2$ et $g_0 = 0$. Elles sont entrées comme

$$\begin{aligned} U[1,0] &= (z\hat{}3 - z)/2; \\ U[2,0] &= 0; \end{aligned}$$

Les opérateurs linéaires auxiliaires sont choisis comme $L_1 = \frac{\partial^4}{\partial z^4}$ et $L_2 = \frac{\partial^2}{\partial z^2}$ Ils sont définis comme

$$\begin{aligned} L[1,u\_] &:= D[u,\{z,4\}]; \\ L[2,u\_] &:= D[u,\{z,2\}]; \end{aligned}$$

Notez que nous utilisons la cession retardé `SetDelayed` (: =) pour définir ces opérateurs linéaires.
Sans perte de généralité, nous considérons le cas où $Re = M = 2$ et $K = 1/2$. Ces paramètres physiques sont :

$$\begin{aligned} Rey &= M = 2; \\ K &= 1/2; \end{aligned}$$

A cette époque, nous avons toutes les données d'entrée pour ce problème, à l'exception des paramètres de contrôle de convergence $c0[k]$.

Hayat 9[12] choisir les paramètres de contrôle de convergence $c0[1] = c0[2] = -0,7$

par $h - courbe$.

Ici, nous minimisons l'erreur de carré résiduel des approximations d'ordre 4 pour obtenir des valeurs optimales pour $c0[k]$

```
GetOptiVar[4,{},{c0[1],c0[2]}] ;
```

Les paramètres de contrôle de la convergence $c0[1]$ et $c0[2]$ se trouvent à environ $-0,5825$ et -0.721452 respectivement.
Ensuite, nous appelons le module principal `BVPh` pour obtenir les approximations d'ordre 20e

```
BVPh[1,20] ;
```

Les approximations d'ordre 20e sont stockés dans `U[i,20],i=1,2`, alors que l'erreur de carré résiduel correspondante est `ErrTotal[20]`.
Nous pouvons utiliser :

```
Plot[{U[1,20],U[2,20]},{z,0,1},AxesLabel->{"z",""},
PlotStyle->{{Thin, Red}, {Dashed, Blue}},
PlotRange->{{0, 1}, {-0.2, 0.2}}]
```

pour tracer les approximations d'ordre 20 , ce qui est représenté sur la figure 3.
Cette figure d'accord avec Hayat[12]. Fig9 et Fig. 12 [3] lorsque $M = 2, Re = 2$ et $K = 0,5$.
Les approximations d'ordre 20 donnent les valeurs de $f''(1) = 3.61076396287$ et $g'(1) = -0.738463496789$ qui sont la même chose avec le résultat de **Hayat**, L'erreur totale du système à chaque deux ordre d'approximations est tracée sur la figure. 4 par la commande :

```
ListLogPlot[Table[{2 i, ErrTotal[2*i]}, {i, 1, 20}],
Joined -> True, Mesh -> All,
PlotRange -> {{2, 20}, {10^(-34), 1}},
AxesLabel -> {"m", "error"}]
```

Nous pouvons voir dans ce que l'erreur diminue magnifiquement.

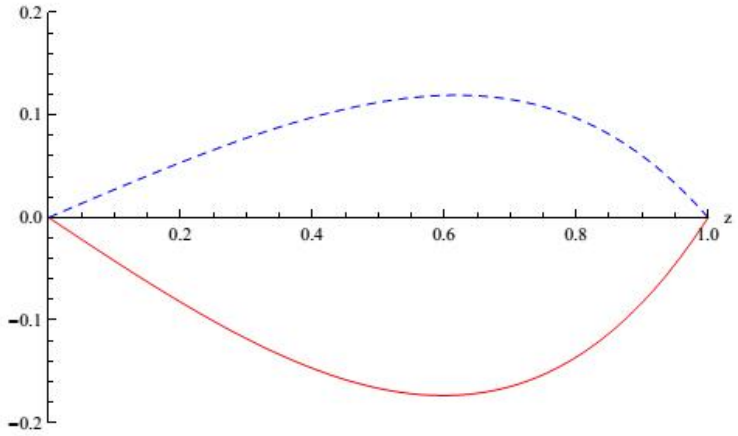


Fig. 3 : La courbe de $f(z)$ (solide), $g(z)$ (en pointillés) pour l'exemple1.

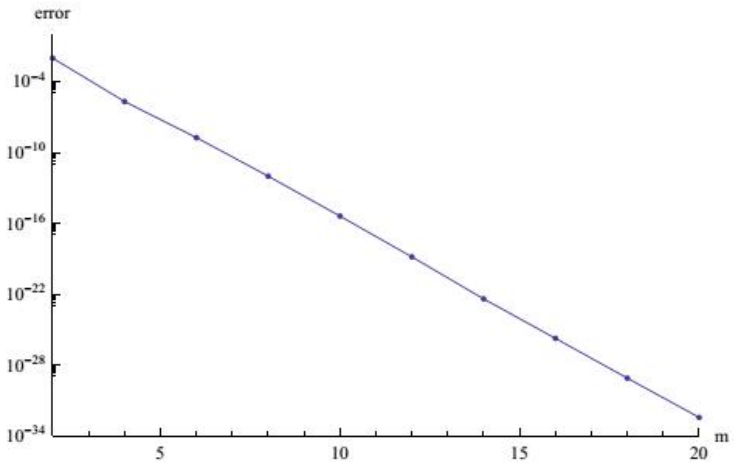


Fig. 4 :Erreur totale en fonction de l'ordre d'approximation pour l'exemple 1.

4.7.2 2.un système d'équations différentielles couplées à la propriété algébrique à l'infini :

Considérons un ensemble de deux équations différentielles non linéaires couplées :

$$f'''(\eta) + \theta(\eta) - f'^2(\eta) = 0, \tag{4.5}$$

$$\theta''(\eta) = 3\sigma f'(\eta)\theta(\eta), \tag{4.6}$$

Soumis aux :

$$f(0) = f'(0) = 0, \quad \theta(0) = 1, \quad f'(+\infty) = \theta(+\infty) = 0, \tag{4.7}$$

où le premier représente la différenciation par rapport à la variable de similarité η, σ est le nombre de Prandtl, $f(\eta)$ et $\theta(\eta)$ rapporter le profil de vitesse et la distribution de température de la couche limite, respectivement.
Liao [1] a utilisé la HAM pour résoudre ce système analytiquement.
Maintenant, nous utilisons la `BVPh 2.0` pour le résoudre.
En vertu de la transformation

$$\xi = 1 + \lambda\eta, \quad F(\xi) = f'(\eta), \quad S(\xi) = \theta(\eta), \tag{4.8}$$

Eqs (5.5) et (5.6) devient :

$$\lambda^2 F''(\xi) + S(\xi) - F^2(\xi) = 0, \tag{4.9}$$

$$\lambda^2 S''(\xi) = 3\sigma F(\xi)S(\xi), \tag{4.10}$$

Soumis aux

$$F(1) = 0, \quad S(1) = 1, \quad F(+\infty) = S(+\infty) = 0. \tag{4.11}$$

Car il ya deux équations différentielles dans le système (5.9) et (5.10) sans une inconnue à déterminer, nous avons `NumEQ = 2` et `1 = TypeEQ`.
Ce nouveau système est défini comme :

```
TypeEQ = 1 ;
NumEQ = 2 ;
f[1,z_,{F_,S_},Lambda_] :=la^2*D[F,{z,2}]+S-F^2 ;
f[2,z_,{F_,S_},Lambda_] :=la^2*D[S,{z, 2}]-3*sigma*F*S ;
```

Les quatre conditions aux limites sont définies comme :

```
NumBC = 4 ;
```

```
BC[1,z_,{F_, S_}] := F /. z -> 1;
BC[2,z_,{F_, S_}] := (G - 1) /. z -> 1;
BC[3,z_,{F_, S_}] := F /. z -> infinity;
BC[4,z_,{F_, S_}] := G /. z -> infinity;
```

Puis,nous laissons entré les intervalles de solution et les intervalles intégrales pour calculer l'erreur de carré résiduel :

```
zL[1] = 1;
zR[1] = infinity;
zL[2] = 1;
zR[2] = infinity;
zRintegral[1] = 10;
zRintegral[2] = 10;
```

Les estimations initiales sont choisies comme $F_0 = \gamma(\xi^{-2} - \xi^{-3}), S_0 = \xi^{-4}$ et sont entrées comme :

```
U[1, 0] = gamma*(z^(-2) - z^(-3));
U[2, 0] = z^(-4);
```

Les opérateurs linéaires auxiliaires sont $L_F = \frac{\xi}{3}\frac{\partial^2}{\partial\xi^2} + \frac{\partial}{\partial\xi}$ et $L_S = \frac{\xi}{5}\frac{\partial^2}{\partial\xi^2}$ qui sont définis comme :

```
L[1, u_] := D[u, {z, 2}]*z/3 + D[u, z];
L[2, u_] := D[u, {z, 2}]*z/5 + D[u, z];
```

Sans perte de généralité, nous laissons considérer le cas où $\sigma = 1, \gamma = 3$ et $\lambda = 1/3$. Ces paramètres physiques et les paramètres de commande $c0[k]$ sont définis comme :

```
sigma = 1;
gamma = 3;
la = 1/3;
c0[1] = -1/2;
c0[2] = -1/2;
```

Ensuite, nous appelons le module principal `BVPh` :

```
BVPh[1,20];
```

pour obtenir les approximations d'ordre 20ème.

Si nous ne sommes pas satisfaits avec la précision de l'approximation d'ordre 20e ,

nous pouvons utiliser `BVPh [21,40]`, au lieu de `BVPh [1,40]`, pour obtenir les approximations d'ordre 40e ou des approximations d'ordre supérieur.
Notez que `U[1,40]` et `U[2,40]` sont les approximations d'ordre 40e du système transformé (14), (15) et (16). Pour tracer la courbe des approximations d'ordre 40e pour le problème initial, nous remplaçons premierement z par $1 + \lambda\eta$ pour obtenir les approximations d'ordre 40e pour $f'(\eta)$ et $g(\eta)$, puis tracer la courbe que nous voulons.
Ceci est fait en Mathemcatica par la commande suivante :

```
trans = {z -> 1 + la*\[Eta]};
Plot[Evaluate[{U[1, 40], U[2, 40]} /. trans],
{\[Eta], 0, 10},PlotRange -> {{0, 10}, {0, 1}},
AxesLabel -> {"\[Eta]", ""},
PlotStyle -> {{Thin, Red}, {Dashed, Blue}}]
```

et la courbe est représentée sur la Fig. 5. Voici `trans={z>1+la*\[ETA]}` est la transformation correspondante, `\[ETA]` est le symbole η dans Mathematica .
L'erreur totale `ErrTotal[k]` du système transformé pour toutes approximations d'ordre deux est tracée sur la figure. 6 par la commande suivante

```
ListLogPlot[Table[{2 i, ErrTotal[2*i]}, {i, 1, 20}],
Joined -> True, Mesh -> All,
PlotRange -> {{2, 40}, {10^(-10), 0.01}},
AxesLabel -> {"m", "error"}]
```

Notez que `ErrTotal[k]` ne mesure pas seulement la précision des approximations d'ordre K pour le problème transformé, mais aussi mesure les approximations correspondant pour

le problème d'origine.

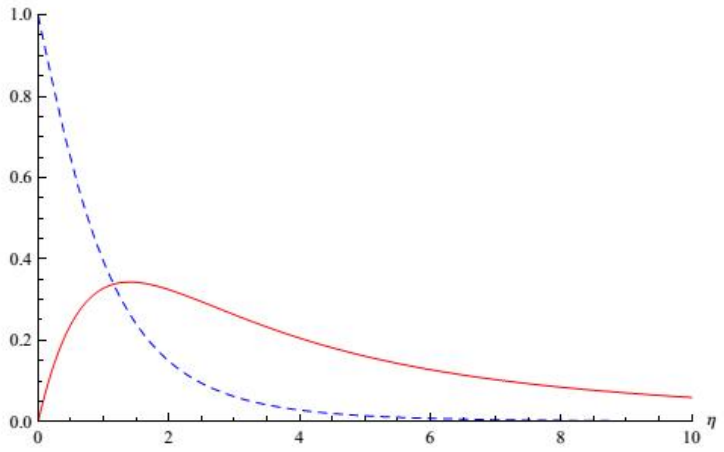


Fig. 5 :La courbe de $f(\eta)$ (solide) et $\theta(\eta)$ (en pointillés) pour l'exemple 2.

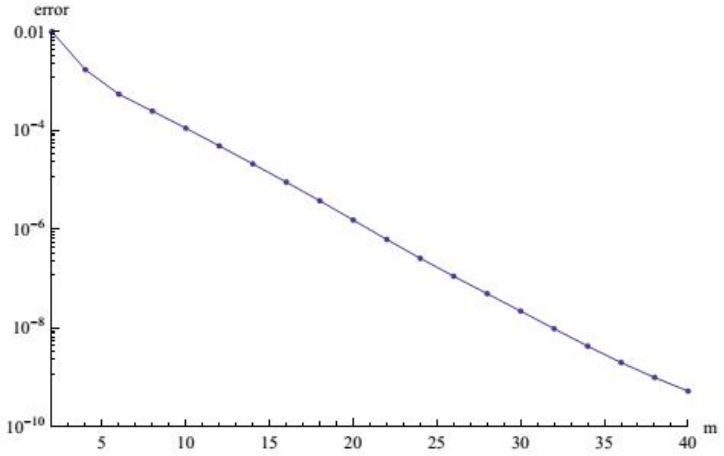


Fig. 6 : Erreur totale en fonction de l'ordre d'approximation pour l'exemple 2.

Les approximations d'ordre $40e$ de $f(0)$ et $g(0)$ sont $0,693268$ et $-0,769879$, respectivement. Résultat numérique de Kuiken est $f''(0) \approx 0,693212$ et $g(0) \approx 0,769861$. Pour obtenir des résultats plus précis, nous avons deux choix.

Le premièr consiste à appeler le module `BVPh` pour obtenir des approximation d'ordre plus élevé comme on a vu précedament, l'autre est d'appliquer l'approximation de Pad'e aux approximations actuelles.

```
hp[Table[D[(U[1,i]/.trans),\[Eta]]/.\[Eta]->0,
{i,0,40}],20,20]
hp[Table[D[(U[2,i]/.trans),\[Eta]]/.\[Eta]->0,
{i,0,40}],20,20]
```

qui donnent 0.693212 et 0.769861, les approximations de **Pad'e**-d'homotopie $[20, 20]$ de $f''(0)$ et $g'(0)$, respectivement.

Notez que nous pouvons comparer la courbe de l'approximation d'ordre 2nth et la courbe de l'approximation de Pad'e-d'homotopie $[n, n]$ d'une manière simple et efficace. ici, nous comparons $U[1, 40]$ et l'approximation Pad'e-d'homotopie $[20, 20]$ de `U[1,i],i=0,...,40`, dans le Mathematica par la commande suivante :

```
Plot[{U[1, 40]/.trans, hp[Table[U[1,i]/.trans,
{i, 0, 40}],20, 20]},{\[Eta],0,10},PlotRange->Full,
AxesLabel->{"\[Eta]", ""},
PlotStyle->{{Thin,Red},{Dashed,Blue}}
```

La comparaison est représentée sur la Fig. 7 De là, nous pouvons voir que les deux sont presque les mêmes.

Cette validation de la convergence des approximations dans une certaine mesure.

La commande ci-dessus est très efficentte, car la commande du terrain dans Mathematica substitue d'abord les points d'échantillonnage dans l'expression, puis applique `hp` de la liste de valeurs numériques, plutôt que s'applique `hp` à une liste d'expressions et alors substitue les points d'échantillonnage dans l'expression résultante.

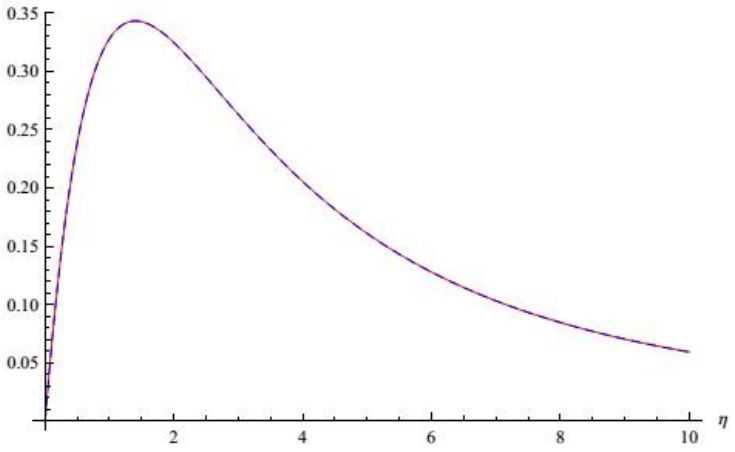


Fig. 7 : La courbe d'approximation d'ordre $40e$ de $f(\eta)$ (solide) et $[20, 20]$ l'approximations de Pad'e-d'homotopie de $f(\eta)$ (en pointillés) pourl'exemple2.

Chapitre 5

Applications de HAM à plusieurs types d'équations

Les applications suivante sont issues d'articles publiés ; la référence est en début de chaque exemple.

En effet chaque problème a été fait d'une manière abrrégée, nous allons faire tout le détail qui manque.

5.1 Exemple 1 (Article de Shijun Liao 1)

On considère le prblème de shijun liao [13] :

$$f'(z) + 2zf^2(z) = 0$$

avec la condition initiale :

$$f(0) = 1$$

qui a la solution exacte[13] : $f(z) = \frac{1}{z^2+1}$.
Ici, nous résolvons ce problème en `BVPh 2.0`. Comme il ya une équation différentielle sans une inconnue à déterminer, nous avons `NumEQ = 1` et `TypeEQ = 1`.

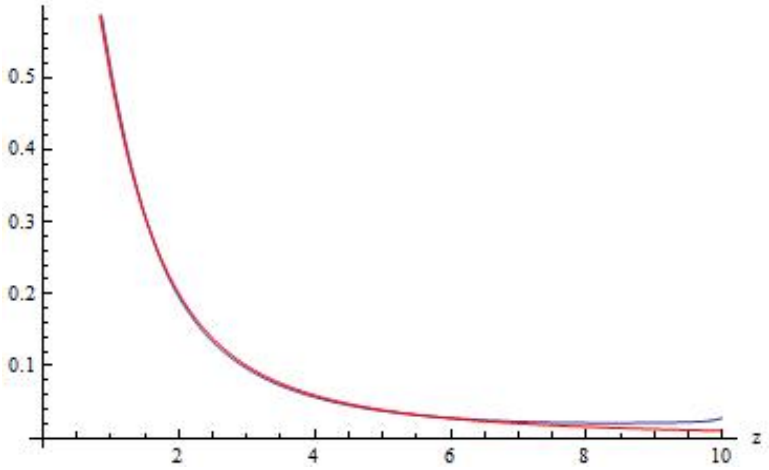
```
Le système est entré en tant que :
(* Filename : Illustrative.m *)
Print["The input file ",$InputFileName," is loaded !"] ;
```

```
(* Modify the control parameters in BVPh if necessary *)
(* Define the governing equation *)
TypeEQ = 1;
NumEQ = 1;
f[1, z_, {f_}, Lambda_] := 2*z*f^2 + D[f, {z, 1}];
(* Define Boundary conditions *)
NumBC = 1;
BC[1, z_, {f_}] := f -1/. z -> 0;
(* Define solution interval and integral interval for
error *)
zL[1] = 0;
zR[1] = infinity;
zRintegral[1] = 10;
(* Define initial guess *)
U[1, 0] = 1/(1+z);
(* Define the auxiliary linear operator *)
L[1, u_] := D[u, {z, 1}];
(* Print input data *)
PrintInput[{f[z]}];
(* Get optimal c0 *)
GetOptiVar[3, {}, {c0[1]}];
(* Gain 10th-order HAM approximation *)
BVPh[1, 10];
```

la courbe est représentée sur la Fig :

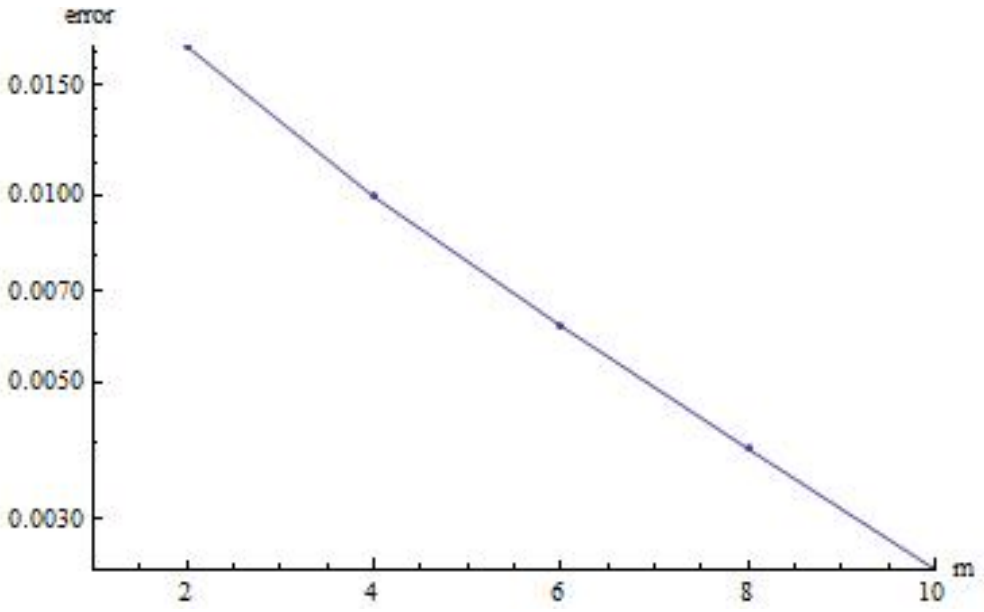
```
(* Plot the solution curves *)
Plot[{U[1,20],1/(z^2+1)},{z,0,10},AxesLabel->{"z",""},
```

```
PlotStyle -> {Thin, Red}]
```



L'erreur totale `ErrTotal[k]` du système transformé pour toutes approximations d'ordre deux est tracée sur la figure suivante par la commande suivante :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2i,ErrTotal[2*i]},{i,1,10}],Joined -> True,
Mesh -> All,PlotRange->{{1, 10},{0,.1}},
AxesLabel -> {"m", "error"}]
```



5.2 Exemple 2 (Article de Shijun Liao 2)

On considère le problème suivante [1],[6] :

$$\frac{\partial^3 u}{\partial z^3}(z,t) + (1-t)\frac{z}{2}\frac{\partial^2 u}{\partial t^2}(z,t) - t(1-t)\frac{\partial^2 u}{\partial z \partial t} + tu(z,t)\frac{\partial^2 u}{\partial z^2}(z,t) - t\left[\frac{\partial u}{\partial z}(z,t)\right]^2 = 0$$

Soumis aux conditions

$$\begin{aligned} u(0,t) &= 0 \\ \frac{\partial u}{\partial z}(0,t) &= 1 \\ \frac{\partial u}{\partial z}(\infty,t) &= 0 \end{aligned}$$

Ici, nous résolvons ce problème en `BVPh 1.0`. Comme il ya une équation différentielle sans une inconnue à déterminer, nous avons `TypeEQ = 1`.

Le système est entré en tant que [6] :

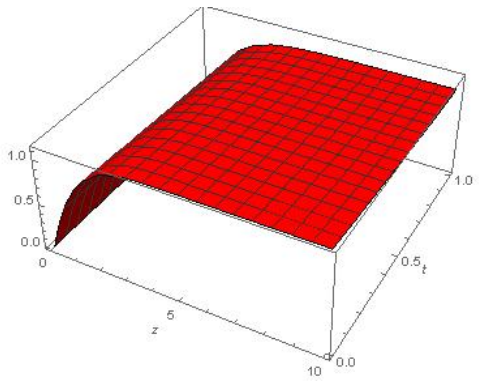
```
(* Install BVPh 1.0 for Mathematica 5.2. *)
(* For Mathematica 8.0, please replace BVPh1_ 0.txt by
BVPh1_ 1.txt *)
<<BVPh1_0.txt ;
(* Define the physical and control parameters *)
(* TypeEQ = 1 -> BVP in a finite domain [0,a] *)
(* TypeEQ = 2 -> eigenvalue problem in a finite domain
[0,a] *)
(* TypeL = 1 -> Chebyshev polynomial as base function *)
(* TypeL = 2 -> Hybrid-base approximation *)
(* TypeBase = 1 -> sine + polynomial *)
(* TypeBase = 2 -> cosine + polynomial *)
(* ApproxQ = 0 -> do NOT approximate the function *)
(* Appprox = 1 -> approximate the function *)
TypeEQ = 1 ;
ApproxQ = 0 ;
ErrReq = 10^(-10) ;
NgetErr = 100 ;
zRintegral = 10 ;
(* Define the governing equation *)
```

```
f[z_,u_,lambda_] := Module[{temp},
temp[1] =D[u,{z,3}]+(1-t)*z/2*D[u,{z,2}]-t*(1-t)*D[u,z,t] ;
temp[2] = u*D[u,{z,2}] - D[u,z]^2 ;
temp[1] + t* temp[2] // Expand
] ;
 (* Define Boundary conditions *)
zR = infinity ;
OrderEQ = 3 ;
BC[1,z_,u_,lambda_] := Limit[u, z -> 0] ;
BC[2,z_,u_,lambda_] := Limit[D[u,z] - 1, z -> 0] ;
BC[3,z_,u_,lambda_] := Limit[D[u,z] , z -> zR ] ;
(* Define initial guess *)
u[0] = 1 - Exp[-z] ;
(* Defines the auxiliary linear operator *)
L[u_] := D[u,{z,3}] - D[u,z] ;
(* Define output term *)
output[z_,u_,k_] := Print["output =
",D[u[k],{z,2}]/.{z->0,t->0}//N] ;
(* Define Getdelta[k] *)
Getdelta[k_] :=Module[{temp,i},
uz[k] = D[u[k],z]//Expand ;
uzz[k] = D[uz[k],z]//Expand ;
uzzz[k] = D[uzz[k],z]//Expand ;
uzuz[k] = Sum[uz[i]*uz[k-i],{i,0,k}]//Expand ;
uzzu[k] = Sum[uzz[i]*u[k-i],{i,0,k}]//Expand ;
uzt[k] = D[uz[k],t]//Expand ;
temp[1] = uzzz[k] +
(1-t)*z/2*uzz[k]-t*(1-t)*uzt[k]//Expand ;
temp[2] = t*(uzzu[k] - uzuz[k]) ;
delta[k] = temp[1] + temp[2]//Expand ;
] ;
(* Print input and control parameters *)
PrintInput[u[z,t]] ;
(* Set convergence-control parameter c0 *)
```

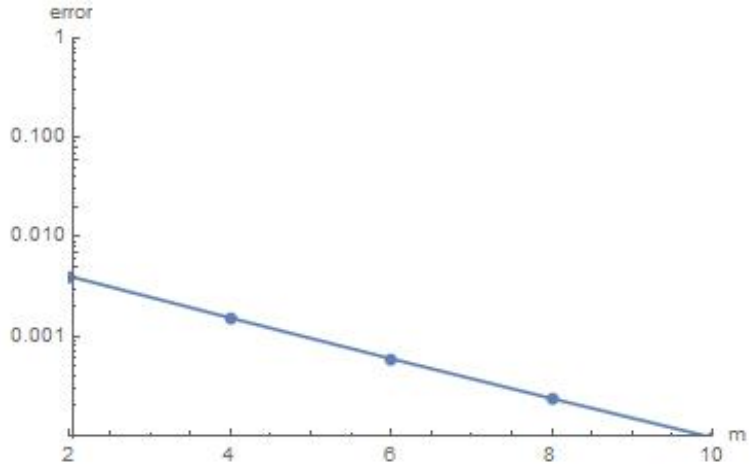
```
c0 = -1/4 ;
Print["c0 = ",c0] ;
(* Gain 6th-order HAM approximation *)
Print[" c0 = ",c0] ;
BVPh[1,10] ;
(* Calculate the squared residual *)
For[k=2, k<=10, k=k+2,
GetErr[k] ;
err[k] = Integrate[Err[k],{t,0,1}] ;
Print[" k = ", k, " Squared residual = ", err[k]//N] ;
] ;
```

pour tracer les approximations d'ordre 10 et la soution exacte, qui est représentée sur la figure suivante, nous pouvons utiliser :

```
Plot3D[{U[10]},{z,0,1},{t,0,1},PlotRange->Full,AxesLabel->{"z",""}, PlotStyle
-> {{Thin,Brown},{Dashed,Blue},{DotDashed, Black}}]
```



L'erreur totale du système à chaque deux ordre d'approximations est tracée sur la figure suivante par la commande :



5.3 Exemple 3 (Article de Shijun Liao 3)

On considère le problème suivant [1] :

$$u^{(4)}(z) + \alpha z u^{(3)}(z) + 3u''(z) + Ru(z)\,u^{(3)}(z) - u'(z)\,u''(z) = 0,$$

Ou α et R sont des paramètre physiques donnés, soumis aux conditions aux limites :

$$\begin{cases} u(0) = 0\ ,\ u''(0) = 0 \\ u(1) = 1\ ,\ u'(1) = 0 \end{cases}$$

Ici, nous résolvons ce problème en `BVPh 1.0`. Comme il ya une équation différentielle sans une inconnue à déterminer, nous avons `NumEQ = 1` et `TypeEQ = 1`.

Le système est entré en tant que :

```
by    BVPh1_ 1.txt *)
<<BVPh1_0.txt ;
TypeEQ = 1 ;
TypeL = 1 ;
TypeBase = 2 ;
ApproxQ = 0 ;
NgetErr = 1 ;
```

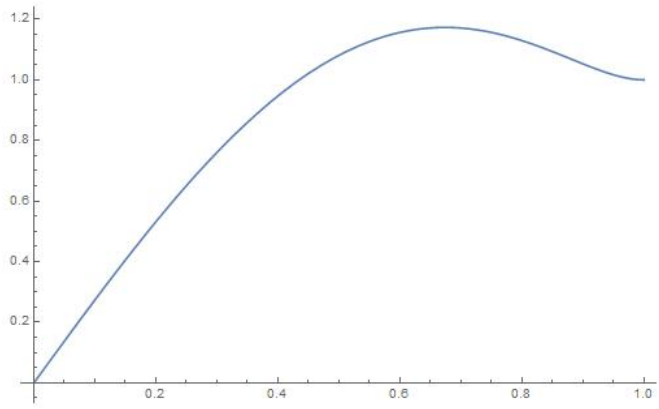
```
(* Define the governing equation *)
f[z_,u_,lambda_] := D[u,{z,4}]+alpha*(z*D[u,{z,3}]+
3*D[u,{z,2}])+R(u*D[u,{z,3}]-D[u,z]*D[u,{z,2}]) ;
alpha = 3/2 ;
R = -11 ;
(* Define Boundary conditions *)
zR = 1 ;
OrderEQ = 4 ;
BC[1,z_,u_,lambda_] := Limit[u, z->0] ;
BC[2,z_,u_,lambda_] := Limit[D[u,{z,2}], z->0 ] ;
BC[3,z_,u_,lambda_] := u - 1 /. z->zR ;
BC[4,z_,u_,lambda_] := D[u,z] /. z->zR ;
(* Define initial guess *)
u[0] = sigma*z +(5-4*sigma)/2*z^3-(3-2*sigma)/2*z^5 ;
sigma = . ;
(* Define output term *)
output[z_,u_,k_] := Print["output = ",D[u[k],z]/.z->0//N] ;
(* Defines the auxiliary linear operator *)
omega[1] = Pi/zR ;
omega[2] = Pi/zR ;
L[f_] := Module[{temp,numA,numB,i},
If[TypeL == 1,
 temp[1] = D[f,{z,OrderEQ}],
 numA = IntegerPart[OrderEQ/2] ;
 numB = OrderEQ - 2* numA//Expand ;
 temp[0] = D[f,{z,numB}] ;
 For[i=1, i<=numA, i++,
 temp[1] = D[temp[0],{z,2}] + (kappa*omega[i])^2*temp[0] ;
 temp[0] = temp[1] ; ] ;] ;
temp[1]//Expand] ;
(* Print input and control parameters *)
PrintInput[u[z]] ;
(* Set c0 and sigma *)
c0 =-1/2 ;
```

```
sigma = 2 ;
Print[" c0 = ",c0, " sigma = ",sigma] ;
(*Gain HAM approximations by the 3rd-order iteration approach*)
iter[1,40,3] ;
```

Les approximations d'ordre 40e sont stockées dans `V[i],i=1,40`, alors que l'erreur correspondante est `ERR[40]`. Nous pouvons utiliser :

```
Plot[{V[40]},{z, 0, 1}, PlotRange->Full,AxesLabel->{"z", ""},
PlotStyle -> {{Thin, Red}}]
```

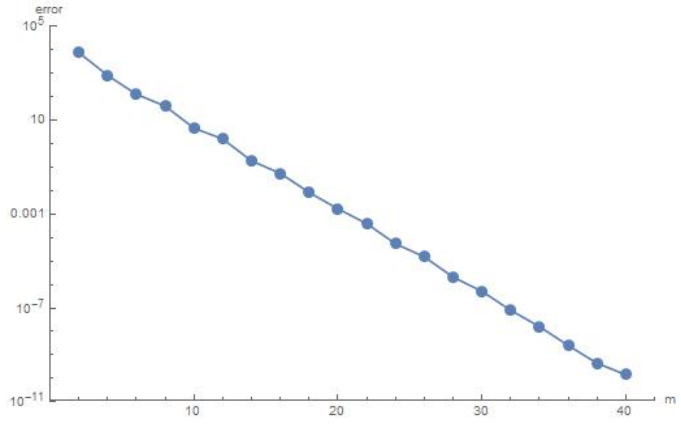
pour tracer les approximations d'ordre 40 et la solution exacte, qui est représentée sur la figure suivante :



L'erreur totale du l'équation à chaque deux ordre d'approximation est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2 i, ERR[2*i]}, {i, 0, 20}], Joined -> True, Mesh -> All,
```

```
PlotRange -> {{0, 42}, {0.00000000001, 10^5}}, AxesLabel -> {"m", "error"}]
```



5.4 Exemple 4 (Article d'Abbasbandy) :

On considère l'équation de Riccati quadratique [19] :

$$u'(z) = 2u(z) - u^2(z) + 1$$

avec condition initiale $u(0) = 0$, La solution exacte est :

$$u(z) = 1 + \sqrt{2}\tanh\left(\sqrt{2}t + \frac{1}{2}\log\left(\frac{\sqrt{2}-1}{\sqrt{2}+1}\right)\right).$$

On choisit l'opérateur linéaire : $\frac{du}{dt} - 2u$ et on prenons comme une approximation initiale $u(0) = z$. L'équation de déformation d'ordre m est donc :

$$L\left[u_m - \chi_m u_{m-1}\right] = cR_m\left(\vec{u_{m-1}}\right)$$

où

$$R_m\left(\vec{u_{m-1}}\right) = \frac{\partial u_m}{\partial z} - 2u_m + \sum_{k=0}^{m-1} u_k u_{m-1-k} - 1;$$

La solution de l'équation de déformation est :

$$u_m(z) = \chi_m u_{m-1} + cL^{-1}\left[R_m\left(\vec{u_{m-1}}\right)\right]$$

Usant des CI : $u_m(0) = 0$, nous obtenons :

$$u(z) = t + t^2 + \frac{1}{3}t^3 - \frac{1}{3}t^4 - .... = u(z) = 1 + \sqrt{2}\tanh\left(\sqrt{2}t + \frac{1}{2}\log\left(\frac{\sqrt{2}-1}{\sqrt{2}+1}\right)\right).$$

Le système est entré en tant que :

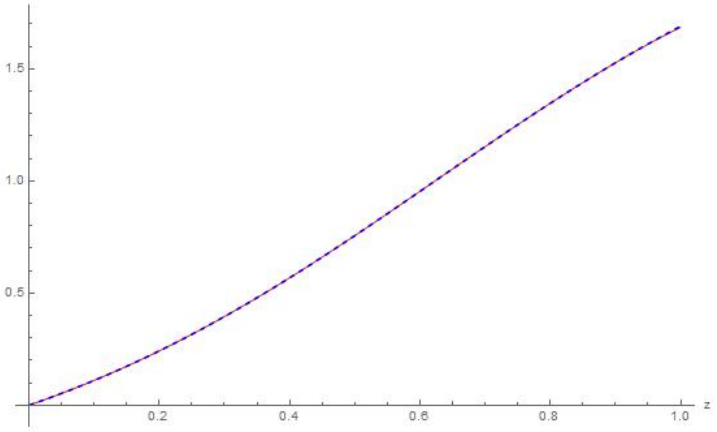
```
(* Filename : Example.m *)
   Print["The input file ",$InputFileName," is loaded!"] ;
   (* Modify control parameters in BVPh if necessary *)
   ErrReq=10^-30 ;
   (* Define the governing equation *)
   TypeEQ = 1 ;
   NumEQ = 1 ;
   TypeL = 1 ;
   f[1,z_,{f_},Lambda_] := D[f,{z,1}] - 2*f + f^2 -1 ;
   (* Define Boundary conditions *)
   NumBC = 1 ;
   BC[1,z_,{f_}] := f /. z->0 ;
   (* Define solution interval *)
   zL[1]=0 ;
   zR[1]=1 ;
   (* Define initial guess *)
   U[1,0]= z ;
   (*z^2-5/2 z+7/2 *)
   (* Define the auxiliary linear operator *)
   L[1,u_] :=D[u,{z,1}] - 2*u ;
   (* Print input data *)
   PrintInput[{f[z]}] ;
   (* Get optimal c0 *)
   GetOptiVar[2,{},{c0[1]}] ;
   (* Gain 10th-order HAM approximation *)
   BVPh[1, 15] ;
```

Les approximations d'ordre 10e sont stockées dans `U[i,10],i=1,10`, alors que l'erreur de carré résiduel correspondante est `ErrTotal[10]`.

Nous pouvons utiliser :

```
Plot[{U[1,15],1+Sqrt[2]*Tanh[Sqrt[2]*z+1/2*Log[(Sqrt[2]-1)/(Sqrt[2] +1)]]},{z,
0, 1}, PlotRange->Full,AxesLabel->{"z", ""},
PlotStyle -> {{Thin, Red}, {Dashed, Blue}}]
```

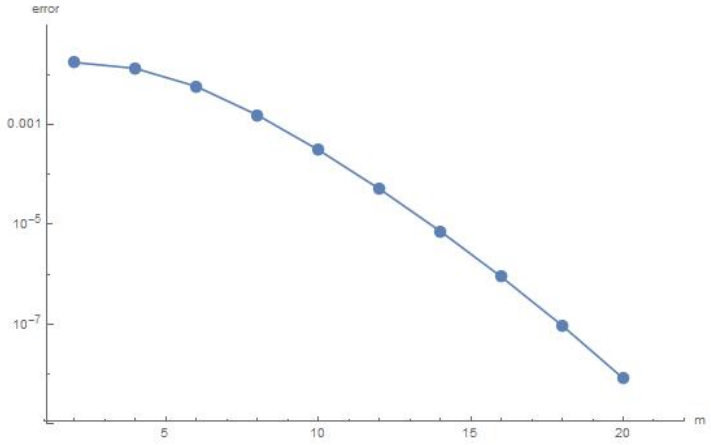
pour tracer les approximations d'ordre 10 et la solution exacte, qui est représentée sur la figure suivante :



L'erreur totale du système à chaque deux ordre d'approximation est tracée sur la figure suivante par la commande :

```
   (* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2i,ErrTotal[2*i]},{i,1,8}], Joined->True,Mesh->All, PlotRange
```

```
-> {{1, 17}, {10^-7, 0.1}}, AxesLabel -> {"m", "error"}]
```



5.5 Exemple 5 (Article de Jeffry)

1. On considère le problème :

$$f^{(2)}(z) = 2f^3(z) - 6f(z) - 2z^3 \qquad 1 \leq z \leq 2,$$

Soumis aux conditions aux limites :

$$\begin{aligned} f(1) &= 2 \\ f(2) &= \tfrac{5}{2} \end{aligned}$$

qui a la solution exacte[14] : $f(z) = z + \frac{1}{z}$.

Ici, nous résolvons ce problème en `BVPh 2.0`. Comme il ya une équation différentielle sans une inconnue à déterminer, nous avons `NumEQ = 1` et `TypeEQ = 1`.

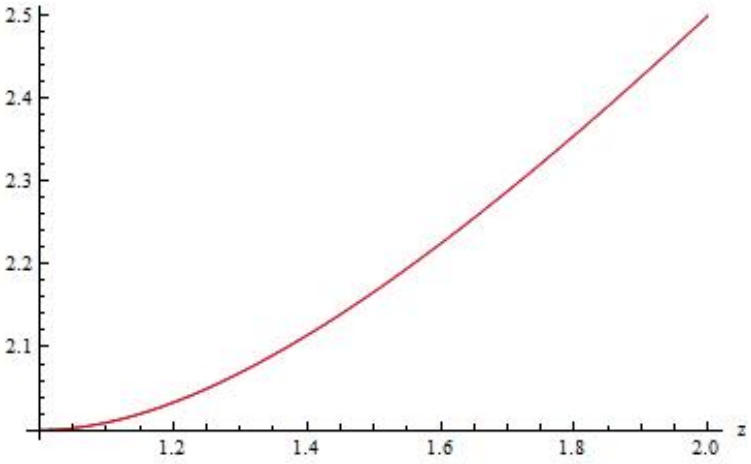
Le système est entré en tant que :

```
(* Define solution interval *)
zL[1]=1 ;
zR[1]=2 ;
(* Define initial guess *)
U[1,0]= 1/2 z+3/2 ;
(*z^2-5/2 z+7/2 *)
(* Define the auxiliary linear operator *)
```

```
L[1,u_] :=D[u,{z,2}] ;
(* Print input data *)
PrintInput[{f[z]}] ;
(* Get optimal c0 *)
GetOptiVar[2,{},{c0[1]}] ;
(* Gain 10th-order HAM approximation *)
BVPh[1, 30] ;
```

la courbe de solution obtenue par **HAM** et la solution exacte est représentée sur la Fig suivante :

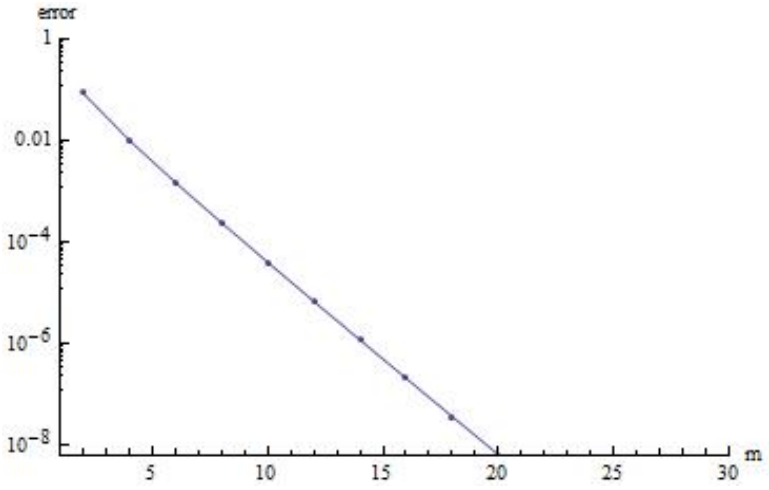
```
(* Plot the solution curves *)
Plot[{U[1,30],z+1/z}, {z, 1, 2},AxesLabel->{"z",""},
 PlotStyle -> {Thin, Red}]
```



L'erreur totale du système à chaque deux ordre d'approximation est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2i,ErrTotal[2*i]},{i,1,10}],Joined->True,
```

```
Mesh->All,PlotRange->{{1,30},{0,1}},AxesLabel->{"m","error"}]
```



2 .On considère le problème :

$$f^{(4)}(z) + f^2(z) = \frac{z^{\frac{-5}{2}}}{16}\left(9 + 30z + 105z^2\right) + z^3(1-z)^4 \qquad 0 \le z \le 1,$$

Soumis aux conditions aux limites :

$$\begin{aligned} f(0) &= 0 \\ f'(0) &= 0 \\ f(1) &= 0 \\ f'(1) &= 0 \end{aligned}$$

qui a la solution exacte[14] : $f(z) = z^{\frac{3}{2}}(1-z)^2$.

Ici, nous résolvons ce problème en `BVPh 2.0`. Comme il ya une équation différentielle sans une inconnue à déterminer, nous avons `NumEQ = 1` et `TypeEQ = 1`.

Le système est entré en tant que :

```
(* Filename : Example.m *)
Print["The input file ",$InputFileName," is loaded!"] ;
(* Modify control parameters in BVPh if necessary *)
ErrReq=10^-30 ;
(* Define the governing equation *)
TypeEQ = 1 ;
```

```
NumEQ = 1;
TypeL = 1;
f[1,z_,{f_},Lambda_] := D[f,{z,4}]+f^2-z^(-5/2)/16 (9+30z+105z^2)-z^3
(1-z)^4;
(* Define Boundary conditions *)
NumBC = 4;
BC[1,z_,{f_}] := f-2 /. z->0;
BC[2,z_,{f_}] := D[f,{z,1}]/. z->0;
BC[3,z_,{f_}] := f /. z->1;
BC[4,z_,{f_}] := D[f,{z,1}] /. z->1;
(* Define solution interval *)
zL[1]=0;
zR[1]=1;
(* Define initial guess *)
U[1,0]= z^4-2z^3+z^2;
(*z^2-5/2 z+7/2 *)
(* Define the auxiliary linear operator *)
L[1,u_] :=D[u,{z,4}];
(* Print input data *)
PrintInput[{f[z]}];
(* Get optimal c0 *)
GetOptiVar[2,{},{c0[1]}];
(* Gain 10th-order HAM approximation *)
BVPh[1, 10];
```

Les approximations d'ordre 10e sont stockées dans `U[i,10],i=1,10`, alors que l'erreur de carré résiduel correspondante est `ErrTotal[10]`.

Nous pouvons utiliser :

```
(* Plot the solution curves *)
Plot[{U[1,8],z^(3/2)(1-z)^2},{z,0,1},AxesLabel->{"z",""},
PlotStyle->{Thin,Red}]
```

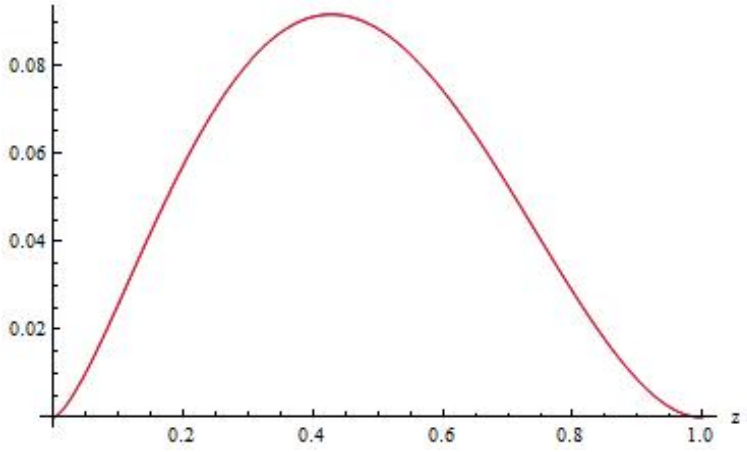
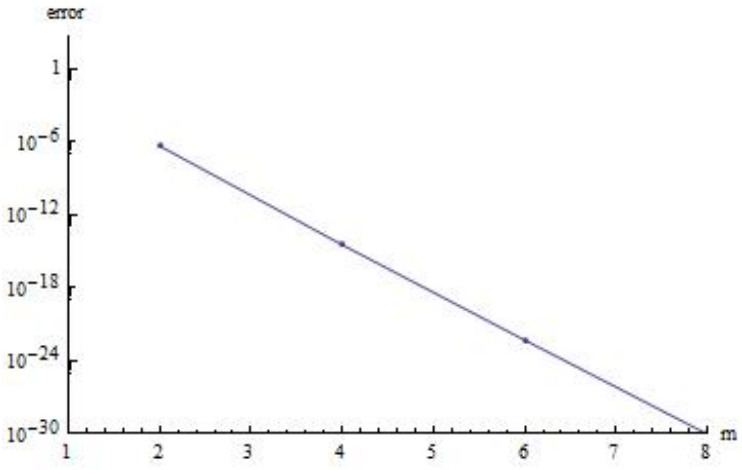


FIG. 5-1 –

pour tracer les approximations d'ordre 10 et la solution exacte, qui est représentée sur la figure suivante :
L'erreur totale du système à chaque deux ordre d'approximation est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2i,ErrTotal[2*i]},{i,1,10}],Joined->True,
 Mesh->All, PlotRange -> {{1, 8}, {0, 1}},
 AxesLabel->{"m", "error"}]
```



5.6 Exemple 7 (Article de D.D Ganji et Miansari)

1.On considère le problème :

$$\begin{cases} y_1'(z) &= y_3(z) - \cos z \\ y_2'(z) &= y_3(z) - e^z \\ y_3'(z) &= y_1(z) - y_2(z) \end{cases}$$

Soumis aux :

$$\begin{cases} y_1(0) &= 1 \\ y_2(0) &= 0 \\ y_3(0) &= 2 \end{cases}$$

qui a la solution exacte [18] :

$$\begin{cases} y_1(z) &= e^z \\ y_2(z) &= \sin z \\ y_3(z) &= \cos z + e^z \end{cases}$$

Ici, nous résolvons ce problème en `BVPh 2.0`. Comme il ya trois équations différentielles en Système sans une inconnue à déterminer, nous avons `NumEQ = 3` et `TypeEQ = 1`.

Le système est entré en tant que :

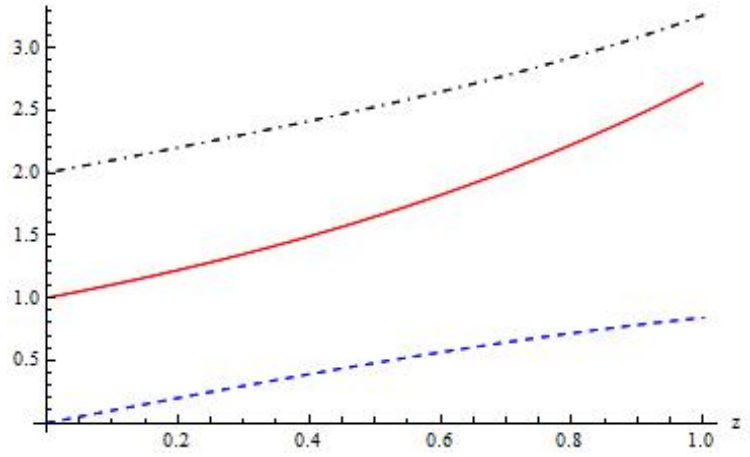
```
(* Filename : Example.m *)
Print["The input file ",$InputFileName," is loaded !"] ;
(* Modify control parameters in BVPh if necessary *)
ErrReq=10^-30 ;
(* Define the governing equation *)
TypeEQ = 1 ;
NumEQ = 3 ;
f[1,z_,{f_,g_,h_},Lambda_] :=D[f,{z,1}]-h+Cos[z] ;
f[2,z_,{f_,g_,h_},Lambda_] :=D[g,{z,1}]-h+E^z ;
f[3,z_,{f_,g_,h_},Lambda_] :=D[h,{z,1}]-f+g ;
(* Define Boundary conditions *)
NumBC = 3 ;
BC[1,z_,{f_,g_,h_}] :=f-1/.z->0 ;
```

```
BC[2,z_,{f_,g_,h_}] :=g/.z->0 ;
BC[3,z_,{f_,g_,h_}] :=h-2/.z->0 ;
(* Define solution interval *)
zL[1]=0 ;
zR[1]=1 ;
zL[2]=0 ;
zR[2]=1 ;
zL[3]=0 ;
zR[3]=1 ;
(* Define initial guess *)
U[1,0]=z+1 ;
U[2,0]=z ;
U[3,0]=z+2 ;
(* Define the auxiliary linear operator *)
L[1,u_] :=D[u,{z,1}] ;
L[2,u_] :=D[u,{z,1}] ;
L[3,u_] :=D[u,{z,1}] ;
(* Print input data *)
PrintInput[{f[z],g[z],h[z]}] ;
(* Get optimal c0 *)
GetOptiVar[4,{},{c0[1],c0[2],c0[3]}] ;
(* Gain 10th-order HAM approximation *)
BVPh[1, 10] ;
```

pour tracer les approximations d'ordre 10 et la solution exacte, qui est représentée sur la figure suivante, nous pouvons utiliser :

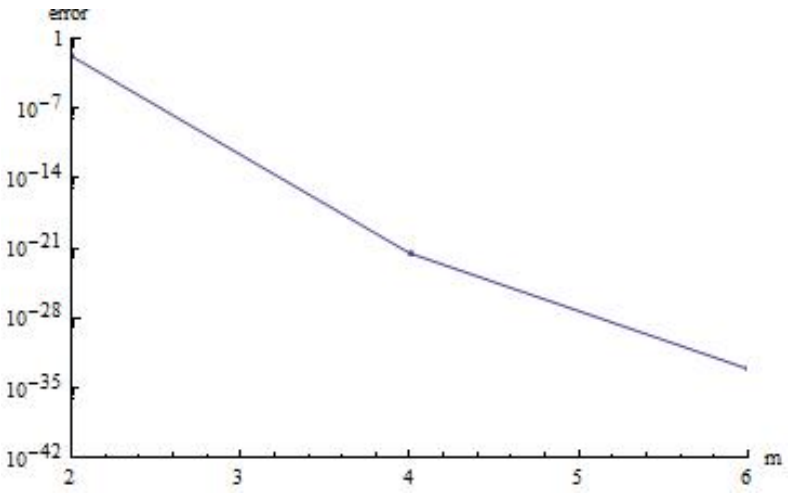
```
(* Plot the solution curves *)
   Plot[{U[1,6], U[2,6], U[3,6], E^z, Sin[z], Cos[z]+E^z},{z,
   0, 1}, PlotRange -> Full, AxesLabel -> {"z", ""},
    PlotStyle -> {{Thin, Red}, {Dashed, Blue}, {DotDashed,
```

```
Black}}]
```



L'erreur totale du système à chaque deux ordre d'approximation est tracée sur la figure suivante par la commande :

```
ListLogPlot[Table[{2i,ErrTotal[2*i]},{i,1,20}],Joined->True,
Mesh -> All, PlotRange -> {{2, 6}, {10^(-42), 1}},
AxesLabel -> {"m", "error"}]
```



2.On considère le problème [18] :

$$\begin{cases} y_1'(z) &= 2y_2^2(z) \\ y_2'(z) &= e^{-z}y_1(z) \\ y_3'(z) &= y_2(z)+y_3(z) \end{cases}$$

Soumis aux :

$$\begin{cases} y_1(0) &= 1 \\ y_2(0) &= 1 \\ y_3(0) &= 0 \end{cases}$$

qui a la solution exacte [18] :

$$\begin{cases} y_1(z) &= e^{2z} \\ y_2(z) &= e^{z} \\ y_3(z) &= ze^{z} \end{cases}$$

Ici, nous résolvons ce problème en `BVPh 2.0`. Comme il ya trois équations différentielles en Système sans une inconnue à déterminer, nous avons `NumEQ = 3` et `TypeEQ = 1`.

Le système est entré en tant que :

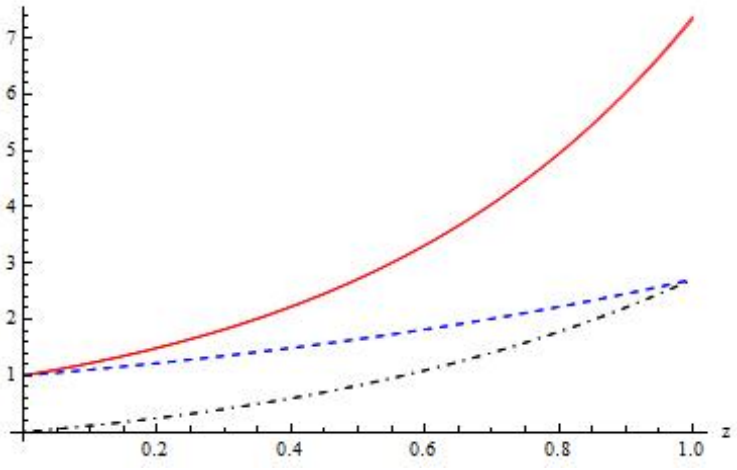
```
(* Filename : Example.m *)
Print["The input file ",$InputFileName," is loaded!"];
(* Modify control parameters in BVPh if necessary *)
ErrReq=10^-30;
(* Define the governing equation *)
TypeEQ = 1;
NumEQ = 3;
f[1,z_,{f_,g_,h_},Lambda_] :=D[f,{z,1}]-2*g^2;
f[2,z_,{f_,g_,h_},Lambda_] :=D[g,{z,1}]-E^-z*f;
f[3,z_,{f_,g_,h_},Lambda_] :=D[h,{z,1}]-g-h;
(* Define Boundary conditions *)
NumBC = 3;
BC[1,z_,{f_,g_,h_}] :=f-1/.z->0;
BC[2,z_,{f_,g_,h_}] :=g-1/.z->0;
BC[3,z_,{f_,g_,h_}] :=h/.z->0;
(* Define solution interval *)
```

```
zL[1]=0 ;
zR[1]=1 ;
zL[2]=0 ;
zR[2]=1 ;
zL[3]=0 ;
zR[3]=1 ;
(* Define initial guess *)
U[1,0]=z+1 ;
U[2,0]=z+1 ;
U[3,0]=z ;
(* Define the auxiliary linear operator *)
L[1,u_] :=D[u,{z,1}] ;
L[2,u_] :=D[u,{z,1}] ;
L[3,u_] :=D[u,{z,1}] ;
(* Print input data *)
PrintInput[{f[z],g[z],h[z]}] ;
(* Get optimal c0 *)
GetOptiVar[4,{},{c0[1],c0[2],c0[3]}] ;
(* Gain 10th-order HAM approximation *)
BVPh[1,30] ;
```

pour tracer les approximations d'ordre 30 et la solution exacte, qui est représentée sur la figure suivante, nous pouvons utiliser :

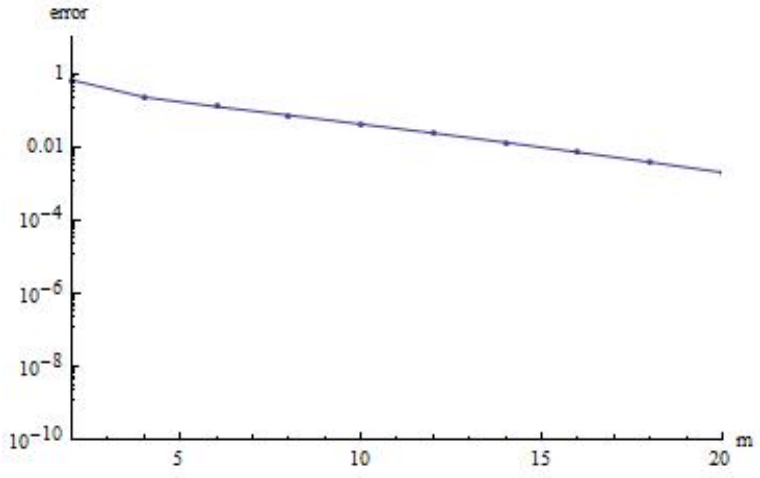
```
(* Plot the solution curves *)
Plot[{U[1,20],U[2,20],U[3,20],E^(2z),E^z,z*E^z},{z,0,1},
 PlotRange ->Full, AxesLabel -> {"z", ""},
```

```
PlotStyle ->{{Thin,Red},{Dashed,Blue},{DotDashed,Black}}]
```



L'erreur totale du système à chaque deux ordre d'approximation est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2i,ErrTotal[2*i]},{i,1,20}],Joined->True,
 Mesh -> All, PlotRange -> {{2, 20}, {10^(-10), 10}},
 AxesLabel -> {"m", "error"}]
```



3.On considère le problème [18] :

$$y^{(3)}(z) = \frac{1}{z} y(z) + y''(z).$$

Soumis aux :

$$y(0) = 0, \quad y'(0) = 1, \quad y''(0) = 2.$$

qui a la solution exacte [18] :

$$y(z) = ze^z,$$

Ici, nous résolvons ce problème en `BVPh 2.0` en deux methodes. la première est :

Comme il ya une équations différentielles sans un inconnue à déterminer, nous avons `NumEQ = 1` et `TypeEQ = 1`.

Le système est entré en tant que :

```
(* Filename : Example.m *)
Print["The input file ",$InputFileName," is loaded!"];
(* Modify control parameters in BVPh if necessary *)
ErrReq=10^-30;
(* Define the governing equation *)
TypeEQ = 1;
NumEQ = 1;
TypeL = 1;
f[1,z_,{f_},Lambda_] := D[f,{z,3}] -1/z*f-D[f,{z,1}];
(* Define Boundary conditions *)
NumBC = 3;
BC[1,z_,{f_}] := f/. z->0;
BC[2,z_,{f_}] := D[f,{z,1}]-1/. z->0;
BC[3,z_,{f_}] := f-E/. z->1;
(* Define solution interval *)
zL[1]=0;
zR[1]=1;
(* Define initial guess *)
U[1,0]= (E-1)*z^2 +z;
(* Define the auxiliary linear operator *)
L[1,u_] :=D[u,{z,3}];
```

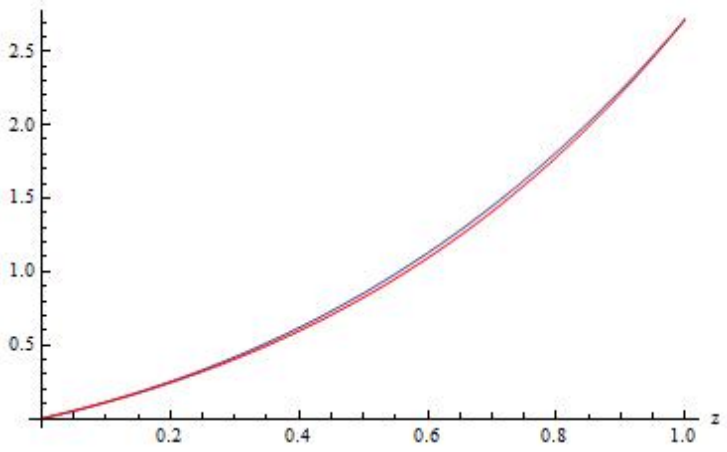


FIG. 5-2 –

```
(* Print input data *)
PrintInput[{f[z]}] ;
(* Get optimal c0 *)
GetOptiVar[2,{},{c0[1]}] ;
(* Gain 10th-order HAM approximation *)
BVPh[1, 20] ;
```

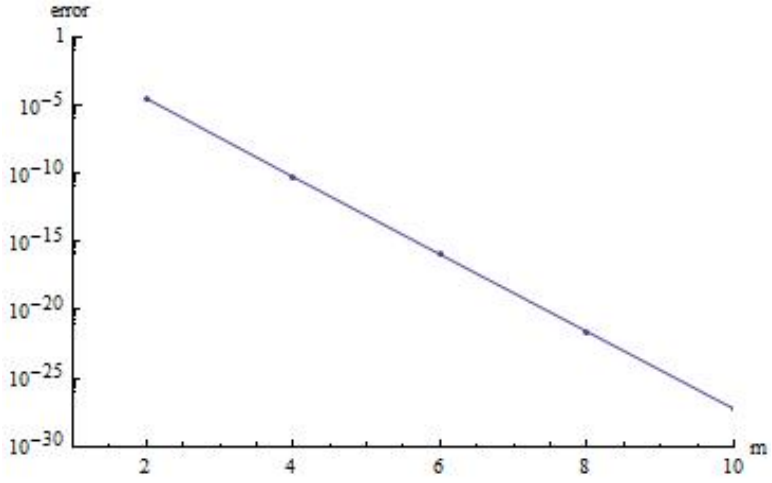
pour tracer les approximations d'ordre 20 et la solution exacte, qui est représentée sur la figure suivante, nous pouvons utiliser :

```
(* Plot the solution curves *)
Plot[{U[1, 12], z*E^z},{z,0,1},AxesLabel->{"z",""},
PlotStyle->{Thin, Red}]
```

L'erreur totale du système à chaque deux ordre d'approximations est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2i, ErrTotal[2*i]},{i,1,10}],
Joined ->True,Mesh->All, PlotRange ->{{1,10},{0,1}}, AxesLabel -> {"m",
```

"error"}]



La deuxième methode a transféférée cette équation en système d'équation différentielle, si on pose :

$$\begin{cases} y_1(z) &= y \\ y_2(z) &= y' \\ y_3(z) &= y'' \end{cases}$$

on dérive pour obtient le système différentiel :

$$\begin{cases} y_1'(z) &= y'(z) = y_2(z) \\ y_2'(z) &= y''(z) = y_3(z) \\ y_3'(z) &= y'''(z) = \frac{1}{z}y_1(z) + y_3(z) \end{cases}$$

Comme il ya trois équations différentielles en Système sans une inconnue à déterminer, nous avons `NumEQ = 3` et `TypeEQ = 1`.

Le système est entré en tant que :

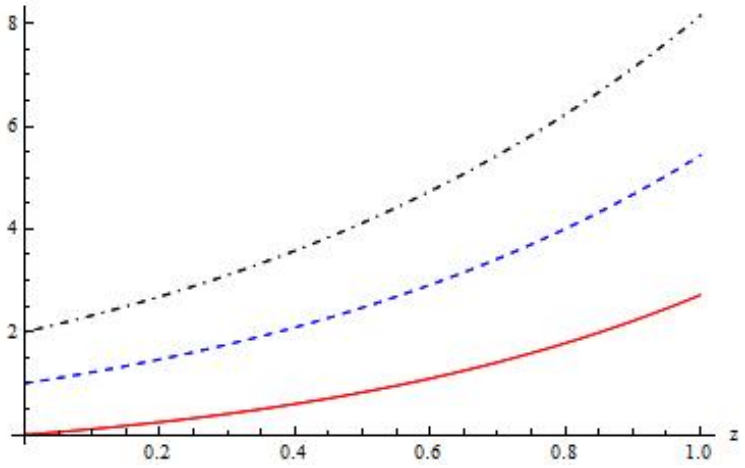
```
(* Filename : Example.m *)
Print["The input file ",$InputFileName," is loaded !"] ;
(* Modify control parameters in BVPh if necessary *)
ErrReq=10^-30 ;
(* Define the governing equation *)
TypeEQ = 1 ;
NumEQ = 3 ;
```

```
f[1,z_,{f_,g_,h_},Lambda_] :=D[f,{z,1}]-g;
f[2,z_,{f_,g_,h_},Lambda_] :=D[g,{z,1}]-h;
f[3,z_,{f_,g_,h_},Lambda_] :=D[h,{z,1}]-1/z*f-h;
(* Define Boundary conditions *)
NumBC = 3;
BC[1,z_,{f_,g_,h_}] :=f/.z->0;
BC[2,z_,{f_,g_,h_}] :=g-1/.z->0;
BC[3,z_,{f_,g_,h_}] :=h-2/.z->0;
The input file $InputFileName is loaded!
(* Define solution interval *)
zL[1]=0;
zR[1]=1;
zL[2]=0;
zR[2]=1;
zL[3]=0;
zR[3]=1;
(* Define initial guess *)
U[1,0]=z;
U[2,0]=z+1;
U[3,0]=z+2;
(* Define the auxiliary linear operator *)
L[1,u_] :=D[u,{z,1}];
L[2,u_] :=D[u,{z,1}];
L[3,u_] :=D[u,{z,1}];
(* Print input data *)
PrintInput[{f[z],g[z],h[z]}];
(* Get optimal c0 *)
GetOptiVar[4,{},{c0[1],c0[2],c0[3]}];
(* Gain 10th-order HAM approximation *)
BVPh[1, 10];
```

pour tracer les approximations d'ordre 10 et la solution exacte, qui est représentée sur la figure suivante, nous pouvons utiliser :

```
(* Plot the solution curves *)
```

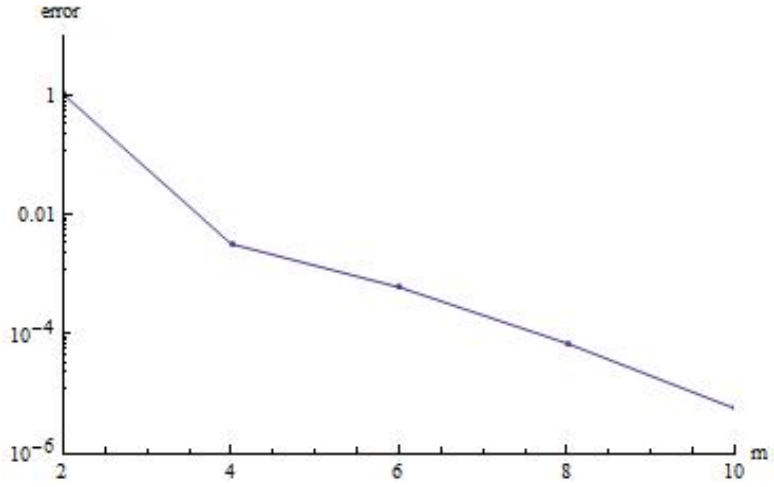
```
Plot[{U[1, 10], U[2, 10], U[3, 10],
z*E^z,(1+z)E^z,(2+z)E^z},{z, 0, 1}, PlotRange -> Full,
AxesLabel -> {"z", ""},
PlotStyle -> {{Thin,Red},{Dashed,Blue},{DotDashed,Black}}]
```



L'erreur totale du système à chaque deux ordre d'approximation est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2i,ErrTotal[2*i]},{i,1,10}],Joined->True,
Mesh->All,PlotRange->{{2, 10},{10^(-6),10}},AxesLabel->
```

```
{"m", "error"}]
```



5.7 Exemple 8 (Article de R.Rajaraman)

On considère le problème suivante :

$$\frac{\partial^2 u}{\partial z^2}(z,t) - \frac{\partial^2 u}{\partial t^2}(z,t) - 4\frac{\partial u}{\partial t}(z,t) - 4u(z,t) = 0$$

Soumi aux conditions

$$\begin{aligned} u(0,t) &= 1 + e^{-2t} \\ \frac{\partial u}{\partial z}(0,t) &= -2 \end{aligned}$$

La solution exacte de ce problème est [11] : $u(z,t) = e^{-2z} + e^{-2t}$.
Ici, nous résolvons ce problème en `BVPh 1.0`. Comme il ya une équation différentielle sans un inconnue à déterminer, nous avons `TypeEQ = 1`.

Le système est entré en tant que :

```
(* Install BVPh 1.0 for Mathematica 5.2. *)
(* For Mathematica 8.0, please replace BVPh1_ 0.txt by
BVPh1_ 1.txt *)
<<BVPh1_0.txt ;
(*****************************************)
(* Define the physical and control parameters *)
(* TypeEQ = 1 -> BVP in a finite domain [0,a] *)
```

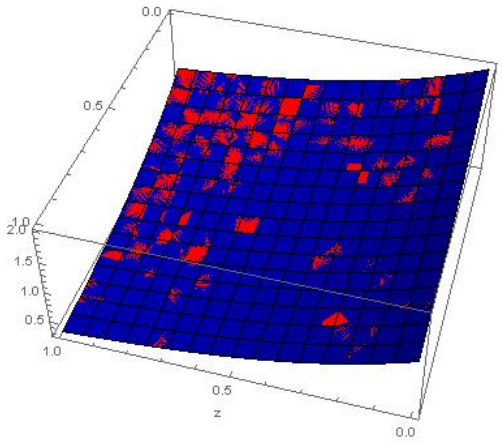
```
(* TypeEQ = 2 -> eigenvalue problem in a finite domain
[0,a] *)
(* TypeL = 1 -> Chebyshev polynomial as base function *)
(* TypeL = 2 -> Hybrid-base approximation *)
(* TypeBase = 1 -> sine + polynomial *)
(* TypeBase = 2 -> cosine + polynomial *)
(* ApproxQ = 0 -> do NOT approximate the function *)
(* Appprox = 1 -> approximate the function *)
(*****************************************)
TypeEQ = 1;
ApproxQ = 0;
ErrReq = 10^(-10);
NgetErr = 100;
zRintegral = 10;
(* Define the governing equation *)
f[z_,u_,lambda_] := Module[{temp},
temp[1] = D[u,{z,2}] -4*u;
temp[2] = - D[u,{t,2}] - 4*D[u,t];
temp[1] + temp[2] // Expand
];
(* Define Boundary conditions *)
zR = 1;
OrderEQ = 2;
BC[1,z_,u_,lambda_] := Limit[u -1-E^(-2*t), z -> 0];
BC[2,z_,u_,lambda_] := Limit[D[u,z] +2, z -> 0];
(* Define initial guess *)
u[0] = 1-2*z + Exp[-2*t];
(* Defines the auxiliary linear operator *)
L[u_] := D[u,{z,2}];
(* Define output term *)
output[z_,u_,k_] := Print["output =
",D[u[k],{z,2}]/.{z->0,t->0}//N];
(* Define Getdelta[k] *)
Getdelta[k_] :=Module[{temp,i},
```

```
uz[k] = D[u[k],z]//Expand;
uzz[k] = D[uz[k],z]//Expand;
ut[k] = D[u[k],t]//Expand;
utt[k] = D[ut[k],t]//Expand;
temp[1] = uzz[k] //Expand;
temp[2] = - 4*u[k] -utt[k] - 4*ut[k];
delta[k] = temp[1] + temp[2]//Expand;
];
(* Print input and control parameters *)
PrintInput[u[z,t]];
(* Set convergence-control parameter c0 *)
c0 = -1;
Print["c0 = ",c0];
(* Gain 6th-order HAM approximation *)
Print[" c0 = ",c0];
BVPh[1,50];
(* Calculate the squared residual *)
For[k=2, k<=50, k=k+2,
GetErr[k];
err[k] = Integrate[Err[k],{t,0,1}];
Print[" k = ", k, " Squared residual = ", err[k]//N];
];
```

pour tracer les approximations d'ordre 50 et la solution exacte, qui est représentée sur la figure suivante, nous pouvons utilisers :

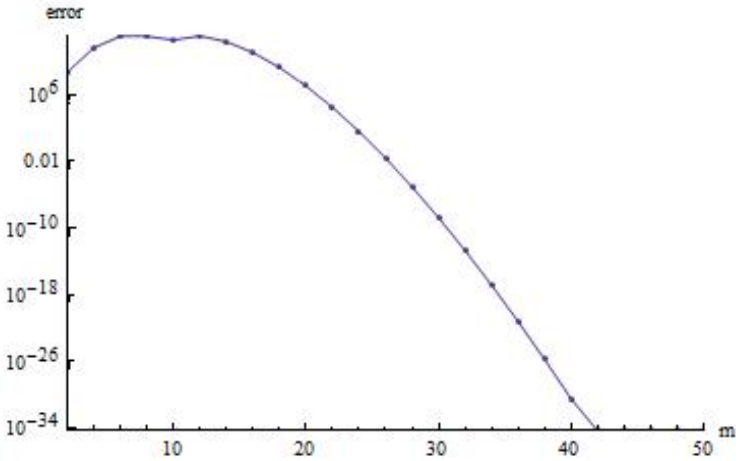
```
(* Plot the solution curves *)
Plot3D[{U[50], E^(-2 t) + E^(-2 z)}, {z, 0, 1}, {t, 0, 1},
 PlotRange -> Full, AxesLabel -> {"z", ""},
```

```
PlotStyle -> {{Thin, Red}, {Dashed, Blue}}]
```



L'erreur totale du système à chaque deux ordre d'approximations est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
ListLogPlot[Table[{2 i, err[2*i]}, {i, 1, 25}], Joined ->
True,
Mesh -> All, PlotRange -> {{2, 50}, {0, 1}},
AxesLabel -> {"m", "error"}]
```



5.8 Exemple 9

On considère le problème suivante :

$$\frac{\partial^3 u}{\partial z^3}(z,t) + \frac{\partial^2 u}{\partial z^2}(z,t) + \frac{\partial u}{\partial z}(z,t) - \frac{\partial^3 u}{\partial t^3}(z,t) - \frac{\partial^2 u}{\partial t^2}(z,t) - \frac{\partial u}{\partial t}(z,t) = 0$$

Soumi aux conditions

$$\begin{aligned} u(0,t) &= t^3 \\ \frac{\partial u}{\partial z}(0,t) &= 3t^2 \\ \frac{\partial^2 u}{\partial z^2}(0,t) &= 3(t+1)^2 \end{aligned}$$

La solution exacte de ce problème est $[X]$: $u(z,t) = (z+t)^3$.
Ici, nous résolvons ce problème en `BVPh 1.0`. Comme il ya une équation différentielle sans un inconnue à déterminer, nous avons `TypeEQ = 1`.

Le système est entré en tant que :

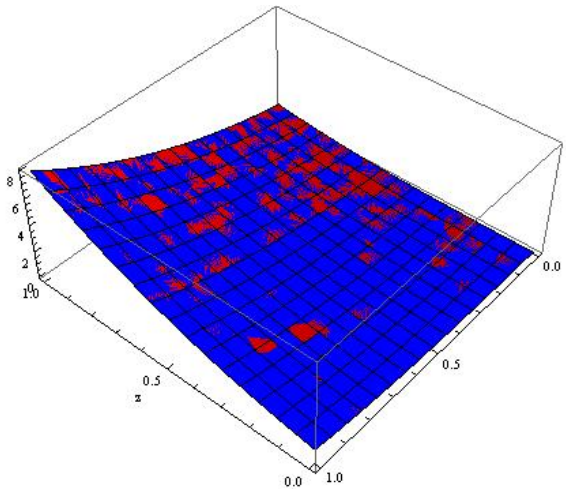
```
(* Install BVPh 1.0 for Mathematica 5.2. *)
(* For Mathematica 8.0, please replace BVPh1_ 0.txt by
BVPh1_ 1.txt *)
<<BVPh1_0.txt ;
(* Define the physical and control parameters *)
(* TypeEQ = 1 -> BVP in a finite domain [0,a] *)
(* TypeEQ = 2 -> eigenvalue problem in a finite domain
[0,a] *)
(* TypeL = 1 -> Chebyshev polynomial as base function *)
(* TypeL = 2 -> Hybrid-base approximation *)
(* TypeBase = 1 -> sine + polynomial *)
(* TypeBase = 2 -> cosine + polynomial *)
(* ApproxQ = 0 -> do NOT approximate the function *)
(* Appprox = 1 -> approximate the function *)
TypeEQ = 1 ;
ApproxQ = 0 ;
ErrReq = 10^(-10) ;
NgetErr = 100 ;
zRintegral = 10 ;
(* Define the governing equation *)
```

```
f[z_,u_,lambda_] := Module[{temp},
temp[1] = D[u,{z,3}] + D[u,{z,2}] + D[u,z] ;
temp[2] = - D[u,{t,3}] - D[u,{t,2}] - D[u,t] ;
temp[1] + temp[2] // Expand
] ;
(* Define Boundary conditions *)
zR = 1 ;
 OrderEQ = 3 ;
BC[1,z_,u_,lambda_] := Limit[u-t^3, z -> 0] ;
BC[2,z_,u_,lambda_] := Limit[D[u,z] - 3*t^2, z -> 0] ;
BC[3,z_,u_,lambda_] := Limit[D[u,z] - 3(1+t)^2, z -> zR ] ;
(* Define initial guess *)
u[0] = 3/2*(2t+1)*z^2+3*t^2*z+t^3 ;
(* Defines the auxiliary linear operator *)
L[u_] := D[u,{z,3}] ;
(* Define output term *)
output[z_,u_,k_] := Print["output =
",D[u[k],{z,2}]/.{z->0,t->0}//N] ;
(* Define Getdelta[k] *)
Getdelta[k_] :=Module[{temp,i},
uz[k] = D[u[k],z]//Expand ;
uzz[k] = D[uz[k],z]//Expand ;
uzzz[k] = D[uzz[k],z]//Expand ;
ut[k] = D[u[k],t]//Expand ;
utt[k] = D[ut[k],t]//Expand ;
uttt[k] = D[utt[k],t]//Expand ;
temp[1] = uzzz[k] + uzz[k] + uz[k]//Expand ;
temp[2] = - uttt[k] - utt[k] - ut[k] ;
delta[k] = temp[1] + temp[2]//Expand ;
] ;
(* Print input and control parameters *)
PrintInput[u[z,t]] ;
(* Set convergence-control parameter c0 *)
c0 = -0.69999999999999995285195701006007050520139939477 0
```

```
602498012487757478163076497554382089814342853722112`100. ;
Print["c0 = ",c0] ;
(* Gain 6th-order HAM approximation *)
Print[" c0 = ",c0] ;
BVPh[1,30] ;
(* Calculate the squared residual *)
For[k=2, k<=30, k=k+2,
GetErr[k] ;
err[k] = Integrate[Err[k],{t,0,1}] ;
Print[" k = ", k, " Squared residual = ", err[k]//N] ;
] ;
```

pour tracer les approximations d'ordre 30 et la solution exacte, qui est représenté sur la figure suivante, nous pouvons utilisers :

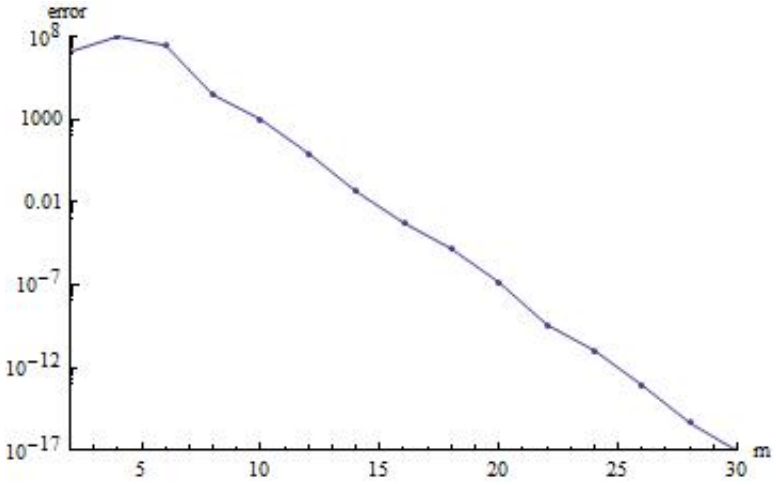
```
(* Plot the solution curves *)
Plot3D[{U[30],(z+t)^3},{z,0,1},{t,0,1},PlotRange->Full,
AxesLabel -> {"z", ""},
PlotStyle -> {{Thin, Red}, {Dashed, Blue}}]
```



L'erreur totale du système à chaque deux ordre d'approximations est tracée sur la figure suivante par la commande :

```
(* Plot the error vs. m (order of approx) *)
```

```
ListLogPlot[Table[{2i,err[2*i]},{i,1,20}],Joined->True,
Mesh -> All, PlotRange -> {{2, 30}, {0, 1}},
AxesLabel -> {"m", "error"}]
```



Bibliogrphie

[1] Shijun Liao : *Homotopy Analysis Method in Non Linear Differential Equations.* Springer Heidelberg Dordrecht London New York.2012.
[2] Liao, S.J : Beyond Perturbation – *Introduction to the Homotopy Analysis Method.*Chapman & Hall/ CRC Press, Boca Raton (2003b).
[3] Nayfeh, A.H : *Introduction to Perturbation Techniques.* John Wiley &Sons, New York, 1981.
[4] S.J. Liao, *The Proposed Homotopy Analysis Technique for the Solution of Nonlinear Problems, PhD thesis*, Shanghai Jiao Tong University, Shanghai, 1992.
[5] S. Abbasbandy : *The application of homotopy analysis method to nonlinear equations arising in heat transfer.* Physics Letters A 360 (2006) 109–113.
[6] Shijun Liao. *Advances in the homotopy analysis method.* World Scientific Publishing Co. Pte. Ltd. 5 Toh Tuck Link, Singapore 596224
[7] Liao, S.J. : *On the homotopy analysis methodfor nonlinear problems.* Appl. Math. Comput.147,499 – 513(2004).
[8] Liao, S.J. : *Notes on the homotopy analysis method– Some definitions and theorems.* Commun. Nonlinear Sci. Numer. Simulat.14, 983 – 997(2009).
[9] Liao, S.J. : *An optimal homotopy-analysis approach for strongly nonlinear differential equations.* Commun. Nonlinear Sci. Numer. Simulat. 15, 2003 – 2016 (2010b).
[10] A. Barari·M. Omidvar·Abdoul R. Ghotbi·D.D. Ganji : *Application of Homotopy Perturbation Method and Variational Iteration Oscillator Differential Equations.* Acta Appl Math (2008) 104 : 161–171
[11] R.Rajaraman : *Analytical solution for the differnt forms of telegraph Equations by Homotopy Analysis Method e*, Global Journal Of Science Frontier Research Mathematics And Decision Sciences. May 2012. Online ISSN : & Print ISSN : 0975-5896.
[12] Syed Tauseef Mohyud-Din.Amjad Hussain and Ahmet Yildirim : *Homotopy Analysis Method for Parametric Differential Equations.* World Applied Sciences Journal 11 (7) : 851-856, 2010.
[13] Shijun Liao. Homotopy analysis method : *A new analytic method for nonlinear problems.* Applied Mathematics and Mechanics (English Edition, Vol. 19, No. 10, Oct. 1998).
[14] Songxin Liang David J. Jeffrey : *An analytical approach for solving nonlinear boun-*

dary value problems in nite domains.
[15] Davood domiri ganji : *An analytical approach for solving nonlinear boundary value problems in nite domains.* Journal of Applied Sciences 01/2008 ; DOI :10.3923/jas.2008.1256.1261.
[16] S. Abbasbandy : *Homotopy analysis method for heat radiation equations.* International Communications in Heat and Mass Transfer 34 (2007) 380–387.
[17] S. Abbasbandy, E. Magyari, E. Shivanian : *The homotopy analysis method for multiple solutions of nonlinear boundary value problems.* Commun Nonlinear Sci Numer Simulat 14 (2009) 3530–3536.
[18] D.D.Ganji, H.Mirgolbabei. Me.Miansari and Mo.Miansari : *Application of Homotopy Perturbation Method to solve linear and non linear systems of ordinary differential equations and differential equation of order three.* Journal of Applied Sciences 8(7) : 1256.1261,2008.
[19] S.Abbasbandy . *Homotopy perturbation method for quadratic Riccati differential equation and comparison with Adomian's decomposition method Homotopy,* Applied Mathematics and Computation 172 (2006) 485–490
[20] Saeid Abbasbandy, Mahnaz Ashtiani, and Esmail Babolian : *Analytic Solution of the Sharma-Tasso-Olver Equation by Homotopy Analysis Method.* Z. Naturforsch.65a,285 – 290 (2010)
[21] V.G. Gupta and Sumit Gupta : *Application of the homotopy analysis method for solving nonlinear Cauchy problem.*Surveys in Mathematics and its Applications.ISSN 1842-6298 (electronic), 1843-7265 (print).
[22] Liao, S.J. and Tan, Y. : *A general approach to obtain series solutions of nonlinear differential equations.* Studies in Applied Mathematics, 119 : 297-354 (2007).
[23] A. Roozi, E. Alibeiki, S.S. Hosseini, S.M. Shafiof, M. Ebrahimi : *Homotopy perturbation method for special nonlinear partial differential equations.* Journal of King Saud University (Science) (2011)23,99–103.
[24] Sirajul Haq, M. Idrees, S. Islam : *Application of optimal Homotopy asymptotic method toeighth order initial and boundary value problems.* International Journal of Applied Mathematics and Computation .Volume 2(4),pp 73–80, 2010.
[25] A. Sami Bataineh, M.S.M. Noorani, I. Hashim : *On a new reliable modification of homotopy analysis method.*Communications in Nonlinear Science and Numerical Simulation 14 (2009) 409–423
[26] Li, Y.J., Nohara, B.T. and Liao, S.J. : *Series solutions of coupled Van der Pol equation by means of homotopy analysis method.* J. Mathematical Physics, 51 : 063517 (2010).

[27] Xu, H., Lin, Z.L., Liao, S.J., Wu, J.Z. and Majdalani, J. : *Homotopy-based solutions of the Navier–Stokes equations for a porous channel with orthogonally moving walls.* Physics of Fluids, 22(5) : 053601 (2010).
[28] Liao, S.J. : *An approximate solution technique which does not depend upon small parameters : a special example.* Int. J. Non-Linear Mechanics, 30 : 371-380 (1995).
[29] Songxin Liang, David J. Jeffrey : *Comparison of homotopy analysis method and homotopy perturbation method through an evolution equation.*
[30] A. Ranjbar, S. H. Hosseinnia, H. Soltani and J. Ghasemi : *A solution of riccati nonlinear differential equation using enhanced homotopy perturbation method (EHPM).* IJE Transactions B : Applications Vol. 21, No. 1, April 2008 - 27.
[31] Muhammet Kurulay : *Approximate analytic solutions of the modified Kawahara equation with homotopy analysis method.* KurulayAdvances in Difference Equations2012,2012 :178.
[32] Nayfeh AH. *Introduction to perturbation techniques.* New York : John Wiley & Sons ; 1981.
[33] Nayfeh AH. *Problems in perturbation.* New York : John Wiley & Sons ; 1985
[34] Molabahrami A, Khani F. The homotopy analysis method to solve the Burgers–Huxley equation. Nonlinear Anal B : Real World Appl, doi :10.1016/ j.nonrwa.2007.10.014 [online].