Université Paris-Saclay - Année 2021-2022
Cours accéléré d'analyse numérique - M2 AMS

Schémas de différences finies pour les équations différentielles. I - EDO.

1 L'approximation numérique des équations différentielles.

On s'intéresse à l'approximation numérique des équations différentielles, ordinaires et aux dérivées partielles. C'est surtout les deuxièmes qui nous intéresseront mais pour introduire la méthode des différences finies, on commencera par les premières.

Les équations différentielles correspondent souvent à des modèles mathématiques décrivant des phénomènes physiques. Dans beaucoup de cas, on ne sait pas calculer une solution explicite d'une équation donnée, et on doit utiliser des techniques de résolution approchée. On s'intéresse alors à la *discrétisation* et à la *résolution numérique* de l'équation, autrement dit on remplace l'équation, qui est posée sur un domaine continu, par un problème discret, en discrétisant le domaine. Si l'équation originale est linéaire on obtient, en discrétisant, un système linéaire, si l'équation originale est non linéaire, on peut obtenir une équation non linéaire à résoudre, par exemple par une méthode de résolution approchée d'équations non linéaires comme la méthode de Newton. Les principales méthodes de discrétisation pour une équation différentielle sont les méthodes des différences finies, la méthode des éléments finis et la méthode des volumes finis. Il y a aussi d'autres classes de méthodes, comme les méthodes spectrales. Au début de ce cours on s'intéresse à la méthode des différences finies.

1.1 La méthode des différences finies - approximation des dérivées.

On considère une équation différentielle, dont l'inconnue est une fonction u, posée sur un domaine physique $\Omega \subseteq \mathbb{R}^d,\ d \geq 1$.

Le principe de la méthode des différences finies est d'approcher la solution u de cette équation en un ensemble discret (mais *grand*) $\{x_1, \ldots, x_N\}$ de points du domaine Ω, en remplaçant les dérivés de u aux points x_i par des quotients de différences faisant intervenir les points voisins de x_i. Cette approche sera valable si la solution du problème est régulière.

Par exemple, on a pour une fonction u d'une seule variable que sa dérivé en un point x de son domaine vérifie

$$u'(x) = \lim_{h\to 0} \frac{u(x+h)-u(x)}{h} = \lim_{h\to 0} \frac{u(x)-u(x-h)}{h} = \lim_{h\to 0} \frac{u(x+h)-u(x-h)}{2h} = \ldots,$$

et on peut donc approcher $u'(x)$ par les quotients

$$\frac{u(x+h)-u(x)}{h},\ \frac{u(x)-u(x-h)}{h},\ \frac{u(x+h)-u(x-h)}{2h}, \ldots,$$

avec h *petit*. On a aussi que si u est deux fois dérivable au point x, alors

$$u''(x) = \lim_{h\to 0} \frac{u(x+h)-2u(x)+u(x-h)}{h^2}$$

et on peut donc aussi approcher $u''(x)$ par le quotient de différences finies

$$\frac{u(x+h)-2u(x)+u(x-h)}{h^2},$$

avec h *petit*.

Une question que l'on se posera souvent dans le cadre de la méthode des différences finies est celle de l'*ordre* de l'approximation. On va expliciter cette notion plus loin, mais essayons de la comprendre dès

maintenant. Elle mesure l'erreur commise dans ces approximations.

Supposons $a,\ b \in \mathbb{R}$ et $u : [a,b] \longrightarrow \mathbb{R}$ une fonction de classe C^3. Soit $x \in]a,b[$.

Exercice 1. Montrer que

$$\left| u'(x) - \frac{u(x+h) - u(x)}{h} \right| = \mathcal{O}(h), \quad \left| u'(x) - \frac{u(x) - u(x-h)}{h} \right| = \mathcal{O}(h)$$

et que

$$\left| u'(x) - \frac{u(x+h) - u(x-h)}{2h} \right| = \mathcal{O}(h^2).$$

Les différences ci-dessus s'appellent *erreurs de troncature* ou de *consistance* au point x. On dira que cette erreur est d'ordre 1 si elle se comporte en $\mathcal{O}(h)$, d'ordre $p > 0$ si elle se comporte en $\mathcal{O}(h^p)$. Plus p est grand, plus l'approximation choisie de la dérivée de u est précise.

Soit $h > 0$ tel qu'il existe $N \in \mathbb{N}$ tel que $\frac{b-a}{N} = h$. Considérons alors les $N+1$ points $x_n = a + nh,\ n = 0, \ldots, N$. L'ensemble discret formé par les points $x_0, \ldots, x_N$ s'appelle une discrétisation régulière ou uniforme de $[a,b]$ de pas h. Soit $U \in \mathbb{R}^{N+1}$ le vecteur $(U_0, \ldots, U_N)$ avec $U_n = u(x_n),\ n = 0, \ldots, N$.

Exercice 2.

1. Déterminer en fonction de h et de U un vecteur $V \in \mathbb{R}^{N+1}$ vérifiant

$$V_n = u'(x_n) + \mathcal{O}(h),\ n = 0, \ldots, N.$$

2. Déterminer en fonction de h et de U un vecteur $W \in \mathbb{R}^{N+1}$ vérifiant

$$W_n = u'(x_n) + \mathcal{O}(h^2),\ n = 0, \ldots, N.$$

3. Illustrer numériquement les deux approximations de la dérivée de u pour la fonction $u(x) = \sin(x)$ dans l'intervalle $[a,b] = [0, 2\pi]$. Pour ce faire représenter dans la même figure la fonction u', le vecteur V et le vecteur W, en fonction de x, avec $h = \frac{2\pi}{10}$ et puis avec $h = \frac{2\pi}{100}$. Représenter dans une autre figure les différences $|u' - V|$ et $|u' - W|$.

Remarquer que la difficulté dans les questions 1) et 2) consiste à écrire une telle approximation pour les points du bord du domaine x_0 et x_N.

1.2 Approximation des EDO.

Dans un premier temps, on s'intéresse à des méthodes numériques pour approcher les équations différentielles ordinaires.

On considère le problème de Cauchy pour une équation différentielle ordinaire, ou pour un système d'équations différentielles ordinaires, de la forme

$$\begin{cases} y'(t) = f(t, y(t)), \\ y(t_0) = y_0, \end{cases} \tag{1}$$

où $f : I \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$ est une fonction continue et localement Lipschitzienne par rapport à sa deuxième variable, avec $I \subseteq \mathbb{R}$ un intervalle ouvert de $\mathbb{R}$ et où $(t_0, y_0) \in I \times \mathbb{R}^n$ est donné. Le problème de Cauchy (1) admet alors une unique solution maximale définie dans un intervalle ouvert $J \subseteq I$: ce résultat est une conséquence du **théorème de Cauchy-Lipschitz**.

On souhaite calculer une solution approchée (*discrète*) de ce problème dans un intervalle de la forme $[t_0, t_f] = [t_0, t_0 + T] \subseteq J$, avec $T > 0$.

Pour cela on procède comme suit.

i) On commence par discrétiser (c'est-à-dire remplacer du continu par du discret) l'intervalle $[t_0, t_0 + T]$. Pour ce faire on se donne $N \in \mathbb{N}$ et on définit une sub-division de $[t_0, t_0 + T]$ en N sous-intervalles définis par les $N + 1$ points, appelées points de la discrétisation,

$$t_0 < t_1 < \cdots < t_N = t_0 + T.$$

On appelle aussi les points t_n les instants de temps, car dans le cas de nombreuses EDO modélisant des phénomènes physiques, la variable t représente le temps.

Le point t_0 est appelé l'**instant initial** et le point t_N l'**instant final**. On appelle aussi :

$h_n := t_{n+1} - t_n$ le **pas de temps** ou **le pas de discrétisation entre t_n et t_{n+1}**, pour $n = 0, \ldots, N-1$;

$h := \max_{n=0,\ldots,N-1} h_n$ le **pas de temps** ou **pas de la discrétisation**.

La discrétisation est dite **uniforme** si $h_n = h$, pour tout $n = 0, \ldots, N-1$. Dans ce cas la pas h de la discrétisation est défini par

$$h = \frac{T}{N}$$

et les $N + 1$ points de discrétisation sont définis par

$$t_n = t_0 + nh, \quad n = 0, \ldots, N.$$

ii) On construit $N + 1$ valeurs $y^0, \ldots, y^N$ qui approchent la solution exacte y de (1), aux points $t_0, \ldots, t_N$, autrement dit on construit des valeurs $y^0, \ldots, y^N$ tels que

$$y^n \approx y(t_n), \text{ pour } n = 0, \ldots, N.$$

Une **méthode numérique** ou un **schéma numérique** est donc la donnée d'une telle construction. Plus précisément, c'est la donnée d'une suite $(y^0, \ldots, y^N)_{N\in\mathbb{N}}$, indexée par $N \in \mathbb{N}$, de valeurs approchant les valeurs exactes $(y(t_0), \ldots, y(t_N))$ de la solution y de (1) aux points de la discrétisation $(t_0, \ldots, t_N)$. On appelle souvent la suite $(y^0, \ldots, y^N)_{N\in\mathbb{N}}$ de **solution approchée** ou **solution numérique** et, pour $N \in \mathbb{N}$ donnée, le terme d'ordre N de cette suite solution approchée (ou solution numérique) associée à une discrétisation de l'intervalle $[t_0, t_0+T]$ de $N+1$ points ou de pas $h = \max_{0,\ldots,N} h_n$.

Comme la valeur de la solution exacte à l'instant initial $y(t_0) = y_0$ est connue, on prendra $y^0 = y_0$.

La dépendance de la suite $(y^0, \ldots, y^N)$ par rapport à N n'est pas toujours spécifiée mais il faut la garder à l'esprit, ainsi que la dépendance de la suite $(t_0, \ldots, t_N)$ définissant les points de la discrétisation de l'intervalle $[t_0, t_0 + T]$.

Lorsque N augment, le nombre de points de la discrétisation augmente et on s'attend à ce que les valeurs approchées $y^0, \ldots, y^N$ approchent de manière de plus en plus précise les valeurs exactes $y(t_0), \ldots, y(t_N)$.

Nous allons nous restreindre à des discrétisations uniformes de l'intervalle $[t_0, t_0 + T]$ (*le cas d'une discrétisation à pas h_n variable est souvent utilisé lorsque l'on veut adapter le pas de la discrétisation pour améliorer la précision du schéma*). Dans ce cas, comme $h = \frac{T}{N}$, on peut de manière équivalente voir la suite approchée $(y^0, \ldots, y^N)$ comme une suite indexée par $N \in \mathbb{N}$ ou comme une suite indexée par $h \in]0, 1]$; on a choisi l'intervalle $]0, 1]$ pour h mais on aurait pu choisir n'importe quel autre intervalle de la forme $]0, h_{max}]$. Ce qui nous intéressera sera le comportement de la solution approchée $(y^0, \ldots, y^N)$ lorsque N tend vers l'infini (*i.e., lorsque le nombre de points de la discrétisation tend vers l'infini, ce qui correspond à ce que l'ensemble discret qui représente l'intervalle $[t_0, t_0+T]$ tend vers cet intervalle*), ou de manière équivalente, lorsque h tend vers 0 (*i.e. lorsque la distance entre deux points consécutifs de la discrétisation tend vers 0*).

Dans cette feuille on va considérer les schémas numériques suivants afin de calculer une approximation y^n de $y(t_n)$ pour $n = 0, \dots, N$.
On pose $y^0 = y_0,$ et, pour $n = 0, \dots, N-1$,

Euler explicite :	$y^{n+1} = y^n + hf(t_n, y^n)$	
Heun :	$y^{n+1} = y^n + \frac{h}{2}\big(f(t_n, y^n) + f\big(t_n + h, y^n + hf(t_n, y^n)\big)\big)$	
Runge Kutta 4 :	$y^{n+1} = y^n + \frac{h}{6}(p_1 + 2p_2 + 2p_3 + p_4)$	où $p_1 = f(t_n, y^n)$, $p_2 = f(t_n + \frac{h}{2}, y^n + \frac{h}{2}p_1)$, $p_3 = f(t_n + \frac{h}{2}, y^n + \frac{h}{2}p_2)$, $p_4 = f(t_n + h, y^n + hp_3)$

Ces trois méthodes sont des méthodes explicites (la valeur y^{n+1} se calcule explicitement à partir de valeurs antérieures), à un pas (la valeur y^{n+1} se calcule uniquement à partir de la valeur précédente y^n). Ces méthodes peuvent s'écrire sous une forme générique

$$(S) \quad y^{n+1} = y^n + h\Phi(t_n, y^n, h),$$

pour $n = 0, \dots, N-1$, avec y^0 donné (nous considérerons toujours $y^0 = y_0$), où $\Phi : [t_0, t_0 + T] \times \mathbb{R}^n \times [0, 1] \longrightarrow \mathbb{R}^n$ est une fonction continue. Par exemple, dans le cas de la méthode d'Euler, $\Phi(t, y, h) = f(t, y)$.

1.2.1 Questions théoriques - convergence d'une méthode numérique.

La convergence d'une méthode numérique pour une EDO est liée à deux notions : la **consistance** et la **stabilité**. Nous allons retrouver ces notions dans le cadre des méthodes de différences finies pour les EDP.

Considérons une méthode numérique définie par (S). Soit y la solution exacte du problème de Cauchy (1). On appelle **erreur locale à l'instant** t_n de la méthode, la quantité

$$e^n = y(t_n) - y^n.$$

La méthode définie par (S) est dite **convergente** sur l'intervalle $[t_0, t_0+T]$ si pour toute donnée initiale y_0,

$$\lim_{h \to 0} \Big(\max_{n=0,\dots,N} \|e^n\| \Big) = 0,$$

autrement dit si pour toute donnée initiale y_0,

$$\lim_{h \to 0} \Big(\max_{n=0,\dots,N} \|y(t_n) - y^n\| \Big) = 0.$$

On remarque que la solution exacte y de (1) vérifie

$$y(t_{n+1}) = y(t_n) + h\Phi(t_n, y(t_n), h) + \varepsilon^n, \quad n = 0, \dots, N-1,$$

où la quantité

$$\varepsilon^n = y(t_{n+1}) - y(t_n) - h\Phi(t_n, y(t_n), h)$$

s'appelle **erreur de consistance** de la méthode (S) à l'étape n, relative à la solution y. C'est l'erreur commise par une itération du schéma sur la solution exacte. La méthode est dite **consistante** si la somme des erreurs de consistance est petite :

$$\lim_{h \to 0} \sum_{n=0}^{N-1} \|\varepsilon^n\| = 0;$$

la méthode est dite **consistante d'ordre au moins** p s'il existe une constante $C > 0$, indépendante de h, tel que

$$\forall\ N \in \mathbb{N},\quad \sum_{n=0}^{N-1} \|\varepsilon^n\| \leq Ch^p.$$

La méthode est consistante si l'erreur commise par une étape du schéma est petite car la discrétisation est cohérente avec l'EDO.

La méthode est dite **stable** s'il existe une constante $M > 0$, indépendante de h, tel que pour toutes suites y^n et $\widetilde{y}^n$ définies par

$$\begin{aligned} y^{n+1} &= y^n + h\Phi(t_n, y^n, h), \quad n = 0, \ldots, N-1, \quad y^0 \text{ donné}, \\ \widetilde{y}^{n+1} &= \widetilde{y}^n + h\Phi(t_n, \widetilde{y}^n, h) + \rho^n, \quad n = 0, \ldots, N-1, \quad \widetilde{y}^0 \text{ donné}, \end{aligned}$$

avec $\rho^n \in \mathbb{R},\ n = 0, \ldots, N-1$, on ait

$$\max_{n=0,\ldots,N} \|y^n - \widetilde{y}^n\| \leq M\Big(\|y^0 - \widetilde{y}^0\| + \sum_{n=0}^{N-1} \|\rho^n\|\Big).$$

Une méthode stable est alors une méthode telle que, lorsque l'on introduit une perturbation ρ^n à chaque étape du schéma, l'erreur totale de la méthode est contrôlée par le cumul des perturbations.

Exercice 3. Montrer que si la méthode est stable et consistante, alors elle est convergente. Montrer que si la méthode est stable et consistante d'ordre au moins p alors il existe une constante $\tilde{C}$ indépendante de h tel que

$$\max_{n=0,\ldots,N} \|y(t_n) - y^n\| \leq \tilde{C}h^p.$$

On utilise souvent la formulation « méthode d'ordre p » au lieu de « méthode consistante d'ordre p »pour une méthode dont l'erreur de consistance est d'ordre p. Ce dernier exercice montre que si la méthode est consistante d'ordre p, l'erreur globale du schéma se comporte aussi comme $\mathcal{O}(h^p)$.

Exercice 4. Supposons la solution y du problème de Cauchy (1) régulière. On peut montrer, en utilisant des développements de Taylor appropriés, que la méthode d'Euler explicite est consistante d'ordre 1 et que la méthode de Heun est consistante d'ordre 2 (celle de RK4 est consistante d'ordre 4 mais les calculs sont plus compliqués).

1.2.2 Implémentation d'une méthode numérique sous python.

Quelques conseils pour l'organisation de votre programme : *vous allez programmer plusieurs schémas numériques et éventuellement les tester sur plusieurs exemples d'EDO. Il est souhaitable de bien organiser votre script pour ne pas se perdre. Par exemple vous pourrez définir au tout début les fonctions second membre des EDO que vous allez tester, en les appelant par exemple $f1$, $f2, \ldots$. Vos fonctions doivent avoir toujours deux variables, t et y, même si elles ne dépendent pas de t (comme ça les fonctions définissant les schémas seront adaptées à tous les cas).Vous pourrez ensuite définir les solutions exactes de ces équations, si elles sont connues, en les appelant par exemple $yex1$, $yex2, \ldots$. Également, vous pouvez définir ensuite les fonctions que vous allez construire pour chacun des schémas. Lorsque vous faites appelle à une fonction pour tester un certain schéma dans une situation concrète, il faut définir avant les paramètres du cas que vous allez tester. Par exemple, faire*

```
t0=0
tf=1
y0=2
h=0.5
euler_exp(f1, t0, tf, y0, h)
```

au lieu de

```
euler_exp(f1,0,1,2,0.5)
```

Cela vous permet de facilement changer les paramètres s'il le faut, et la visibilité de votre programme sera aussi facilitée.

Exercice 5.

1. Pour chacune des trois méthodes numériques données, écrire une fonction python de la forme

   ```
   meth(fct, t0, tf, y0, h)
   ```

   avec meth = euler_exp, Heun ou RK4, prenant en argument fct la fonction $f(t,y)$ de (1), les extrémités t0 et tf de l'intervalle de temps $[t_0, t_f] = [t_0, t_0 + T]$, la donnée initiale y0 et le pas de temps h. Cette fonction devra retourner deux tableaux :
   — $[t_0, t_1, ..., t_N]$, tableau numpy unidimensionnel de taille $(N+1) \times 1$ représentant la subdivision de l'intervalle $[t_0, t_f]$ de pas h considérée,
   — $[y^0, y^1, ..., y^N]$, tableau numpy de taille $(N+1) \times n$ représentant la solution approchée aux instants t_n, $n = 0, \ldots, N$, donnée par la méthode choisie.
   Remarque 1 : *votre fonction meth pourra plutôt dépendre de N, le nombre de pas de la subdivision considérée, au lieu de dépendre du pas h. Dans les deux cas,* **faire très attention** *à ce que le nombre de points de la discrétisation soit cohérent avec le pas de la discrétisation.*
   Remarque 2 : *la structure de votre fonction pourra aussi être autre que celle proposée.*
   Remarque 3 : *vous pouvez dans un premier temps traiter le cas $n = 1$, d'une EDO scalaire, puis essayez d'étendre votre fonction au cas $n > 1$.*

2. Testez vos trois fonctions sur le modèle logistique

$$(P_1) \qquad \begin{cases} y'(t) = cy(1 - \frac{y}{b}), \\ y(0) = a, \end{cases}$$

   dont la solution exacte est

$$y(t) = \frac{b}{1 + \frac{b-a}{a} e^{-ct}},$$

   avec les données $c = 1$, $b = 2$, $a = 0.1$, dans l'intervalle $[0, 15]$, avec un pas $h = 0.2$. Tracer sur la même fenêtre la solution exacte et les solutions approchées, obtenue avec le pas h et dans une autre fenêtre avec un pas égal à $\frac{h}{2}$.

3. Testez ensuite vos fonctions dans le cas vectoriel $n > 1$ ($n = 2$) sur le problème

$$(P_2) \qquad \begin{cases} y''(t) = -y(t) + \cos(t) \\ y(0) = 5, \quad y'(0) = 1, \end{cases}$$

   dont la solution exacte est

$$y(t) = \frac{1}{2}\sin(t)t + 5\cos(t) + \sin(t),$$

   dans l'intervalle $[0, 15]$, avec un pas $h = 0.2$.
   Pour ce faire, il faut écrire l'équation d'ordre 2 de (P_2) sous la forme d'un système de deux équations d'ordre 1 dans les nouvelles variables $u(t) = y(t)$ et $v(t) = y'(t)$. On se ramènera alors à la résolution d'une équation de la forme

$$X' = F(t, X), \quad \text{avec } X = (u, v) = (y, y')^T.$$

Représenter à nouveau la solution exacte et les solutions approchées dans une même fenêtre graphique.
La solution y de (P_2) correspond à la première composante du vecteur X ci-dessus. Votre fonction euler_exp retournera dans ce cas, si vous avez respecté la structure conseillée, un tableau de taille $(N+1)\times 2$, N étant le nombre de points de la discrétisation. Ce tableau donne les valeurs approchées de X au points de la discrétisation considérée, la première colonne correspondant à la première composante de X, la seconde à la seconde composante de X. La solution approchée de (P_2) que l'on cherche correspond alors à la première colonne de ce tableau.

Exercice 6. [Étude de l'ordre de la méthode]
On se donne un problème de Cauchy de la forme (1) et une méthode numérique pour approcher la solution de (1) dans un intervalle de la forme $[t_0, t_0+T]$. Pour un certain pas de temps h fixé (ou pour un certain nombre de points de la discrétisation N fixé), l'erreur globale entre la solution approchée associée à une discrétisation de pas h de l'intervalle $[t_0, t_0+T]$ et la solution exacte est donnée par :

$$E_h = \max_{k=0,\cdots,N}(|y(t_k) - y^k|).$$

Ci dessus, $y(t_k)$ est la solution exacte à l'instant t_k et y^k la valeur approchée de $y(t^k)$, donnée par le schéma numérique.

Remarque : *l'erreur globale E dépend du pas h, ou, de manière équivalente, du nombre de points N de la discrétisation. Parfois dans la littérature on ne spécifie pas la dépendance de E par rapport à h, mais on doit toujours garder à l'esprit cette dépendance et que l'erreur est donc une fonction de h.*

Considérons le problème

$$(P_3) \qquad \begin{cases} y'(t) = \frac{\cos(t)-y(t)}{1+t}, \\ y(0) = -\frac{1}{4}, \end{cases}$$

dont la solution exacte est

$$y(t) = \frac{\sin(t) - 1/4}{1+t}.$$

1. Calculez les solutions approchées de (P_3) obtenues avec le schéma d'Euler explicite, avec $h = 1/2^s$ pour $s = 1, 2, ..., 8$; Calculer, pour chaque valeur de $h = 1/2^s$, $s = 1, 2, ..., 8$, l'erreur globale E_h correspondante. Représentez ensuite, en échelle logarithmique, l'erreur en fonction du pas de temps h, autrement dit, représentez $\log(E_h)$ en fonction de $\log(h)$. Vous devez obtenir des points qui sont à peu près alignés sur une droite de pente 1. Vérifiez graphiquement que c'est le cas, en estimant la pente de la droite passant au plus prêt des points (ou en représentant une droite de pente 1 qui passe par un des points et en vérifiant que tous les points sont à peu près sur cette droite).

   Ceci signifie que $\log(E_h) \sim C + \log(h)$ et donc que $E_h \sim \widetilde{C}h$, pour certaines constantes C et $\widetilde{C}$. On dit que la méthode d'Euler explicite est d'ordre 1 : c'est l'ordre de la puissance de h dans cette relation. On a donc que l'erreur globale E_h tend vers 0 comme h. L'ordre d'une méthode donne une indication sur sa vitesse de convergence. Une méthode d'ordre p est une méthode dont l'erreur globale tend vers 0 comme h^p. Donc plus l'ordre est élevé, plus la méthode converge plus vite.

   Remarque : *pour étudier numériquement l'ordre d'une méthode, on utilise souvent l'échelle logarithmique pour tracer l'erreur en fonction du pas de discrétisation h. La pente de la droite obtenue donne l'ordre p de la méthode : si $E_h \sim Ch^p$ alors $\log(E_h) \sim \log(C) + p\log(h)$.*

2. Vous pouvez refaire l'exercice pour la méthode de Heun et de RK4. Vous allez conclure que la méthode de Heun est d'ordre 2 et la méthode de RK4 d'ordre 4.

1.3 Un exemple de méthode implicite : simulation du système de Van der Pol par la méthode d'Euler implicite en utilisant la méthode de Newton.

On considère le problème de Cauchy pour le système de Van der Pol :

$$\begin{cases} x'(t) = \varepsilon \left(x(t) - \frac{x(t)^3}{3} \right) + y(t), \\ y'(t) = -x(t), \end{cases}$$

de donnée initiale

$$x(0) = x_0, \quad y(0) = y_0.$$

Nous allons calculer une solution approchée de ce problème par la **méthode d'Euler implicite**, dont l'itération est donnée par

$$Y^{n+1} = Y^n + hf(t^{n+1}, Y^{n+1}).$$

Pour ce faire, nous allons utiliser la méthode de Newton pour résoudre l'équation non linéaire définissant l'itération de la méthode d'Euler implicite.

Le système de Van der Pol est un exemple d'un problème dit « raide ». Il s'agit de problèmes dont la solution présente d'importantes variations dans un intervalle de temps très court. Nous verrons qu'au contraire des méthodes explicites, les méthodes implicites se comportent en général bien vis à vis de problèmes dits « raides ».

1.4 Définition des fonctions du système.

Exercice 7.
Écrire le système de Van der Pol sous la forme

$$Y' = F(Y),$$

avec $Y = (x, y)^t$ et $F : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$. Définir la fonction F, ainsi que sa matrice Jacobienne DF en complétant les fonctions python suivantes (respecter la structure donnée : F retourne un array de taille 2 de composantes F_1 et F_2 et DF un tableau de taille 2×2 avec les dérivées partielles de F) .

```
# ep= A completer avec la valeur du parametre epsilon

def F(Y):
    x=Y[0]
    y=Y[1]
    #F1=... # A completer : premiere composante de F
    #F2=... # A completer : deuxieme composante de F
    return np.array([F1,F2])

def DF(Y):
    x=Y[0]
    y=Y[1]
    # DFL1=np.array([ ... , ... ]) # A completer : premiere ligne de la matrice
                                   # jacobienne de F en Y=(x,y)
    # DFL2=... # A completer : deuxieme ligne de la matrice jacobienne de F
    return np.array([DFL1,DFL2])
```

1.5 La méthode de Newton.

La fonction python suivante calcule une solution approchée de l'équation

$$G(Y) = 0,$$

où $G : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, par la méthode de Newton, que l'on rappelle est définie par la suite

$$Y_{n+1} = Y_n - (DG(Y_n))^{-1} G(Y_n).$$

Y_0 étant donné, sous certaines conditions la suite $\{Y_n\}_n$ converge vers une solution Y de l'équation $G(Y) = 0$.

Pour utiliser la fonction Newton, il faut au préalable se donner :

- une valeur *eps* définissant la tolérance avec laquelle on va calculer une solution de $G(Y) = 0$;
- la fonction G et sa matrice jacobienne DG ;
- l'initialisation Y_0 ;
- un nombre maximal *nmax* d'itérations à ne pas dépasser.

Exercice 8.
Compléter la fonction python suivante :

```
def Newton(eps,G,DG,Y0,nmax):
    tol=np.linalg.norm(G(Y0))
    n=1 # n = nombre d'iterations qui seront effectuees
    Y=Y0
    while (tol>eps and n<nmax):
        # A completer ....
    return n,Y
```

Pour implémenter la méthode d'Euler implicite pour le système de Van der Pol, on remarque que l'itération de la méthode s'écrit sous la forme

$$Y^{n+1} = Y^n + hF(Y^{n+1}).$$

On peut donc la re-écrire sous la forme

$$G_n(Y^{n+1}) = 0,$$

où $G_n(Y) = Y - Y^n - hF(Y)$. Pour implémenter le méthode d'Euler implicite, on doit donc à chaque itération trouver la solution Y de $G_n(Y) = 0$. La fonction G_n dépend de Y^n et elle change alors à chaque itération. On peut la définir à chaque itération à partir de la fonction F en utilisant la fonction *lambda* de python.

Exercice 9. Construire une fonction donnant la solution approché Y de l'EDO $Y' = F(Y)$, de condition initiale $y(t_0) = Y0$, associée à une discrétisation uniforme de pas h de l'intervalle $[t_0, tf]$, en suivant le modèle suivant :

```
def EI(F,t0,tf,Y0,h):
    #... A completer
    return Y # vecteur contenant les valeurs approchees
             # obtenues par la methode d'Euler implicite
```

Exercice 10. On considère $\varepsilon = 10,\ x_0 = 2,\ y_0 = 0$. Compléter votre code pour obtenir la solution approchée obtenue par la méthode d'Euler implicite dans l'intervalle $[0, 30]$ et avec un pas $h = 0.05$. Représenter la solution approchée x ainsi obtenue, en fonction du temps.

Dans un autre graphique, représenter dans le plan (x, y) les trajectoires des solutions approchées obtenues.

Exercice 11. Refaire l'exercice en utilisant les méthodes programmées auparavant. On pourra aussi utiliser un solveur d'équations différentielles ordinaires fournis par python, la fonction `odeint` du package `scipy.integrate` de python. Comparer les résultats obtenus.