

Résolution d'équations différentielles ordinaires

A la fin du chapitre, l'étudiant doit être capable de:

1. Mettre en œuvre les schémas d'intégration d'Euler, Crank-Nicolson, Adams-Bashforth et Runge-Kutta pour les équations différentielles ordinaires du premier ordre
2. Calculer l'ordre d'une méthode de résolution d'une équation différentielle ordinaire du premier ordre
3. Mesurer numériquement l'ordre d'une méthode de résolution d'une équation différentielle ordinaire du premier ordre

Equation différentielle

- Dans de nombreux cas, un effort de modélisation, une démarche de type ingénieur, se solde par un problème aux valeurs initiales du type:

$$\frac{dy}{dt} = Y(t, y), \quad y(t_0) = y_0$$

- Exemples:
 - Principe fondamental de la dynamique:

$$\frac{dV}{dt} = F/m, \quad V(t = 0) = V_0$$

- Processus chimique

$$\frac{dC}{dt} = -aC^b, \quad C(t = 0) = C_0$$

Discrétisation

- Si une résolution analytique est parfois possible

$$\text{ex: } \frac{dy}{dt} = yt, \quad y(t = 0) = 1,$$

- ce n'est en général pas le cas

$$\text{ex: } \frac{dy}{dt} = \sqrt{y+t}, \quad y(t = 0) = 1,$$

- On utilise alors un algorithme de résolution qui génère une approximation de la solution $y(t)$ en un nombre fini d'instants:

$$t_n = t^n = n\Delta t \quad ; \quad y_n = y^n = y(n\Delta t)$$

- Δt est le pas de temps de discrétisation (supposé constant dans le cadre de ce cours)

Algorithme exact

- La condition initiale fournissant la valeur pour la valeur $n = 0$ de l'indice, un algorithme de résolution est en fait une relation de récurrence permettant de déterminer la valeur de la fonction inconnue à l'instant t^{n+1} connaissant ses valeurs aux instants précédents

$T_0 = 0$

$$y^0, y^1, \dots, y^{n-2}, y^{n-1}, y^n \rightarrow y^{n+1}$$



- Le point de départ est en général la relation exacte:

$$y^{n+1} = y^n + \int_{t^n}^{t^{n+1}} \frac{dy}{dt} dt = y^n + \int_{t^n}^{t^{n+1}} Y(t, y) dt$$

- Comment estimer l'intégrale de Y ?



Méthodes à un pas


- Ce sont les méthodes d'intégration qui s'écrivent sous la forme:

$$y^{n+1} = y^n + \Delta t \times \Phi$$

- Φ est le taux de variation de y au cours de l'itération

- Puisque $\frac{dy}{dt} = Y(t, y)$, sa valeur exacte est $\Phi_{exact} = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} Y(t, y) dt$ 

- La méthode est alors d'ordre p si: 

$$\Phi_{exact} - \Phi = O(\Delta t^p)$$
 

Méthodes à un pas classiques

- Euler explicite // Ordre 1:

$$\Phi = Y(t^n, y^n) = Y^n$$



- Euler implicite // Ordre 1:

$$\Phi = Y(t^{n+1}, y^{n+1}) = Y^{n+1}$$



- Crank-Nicolson // Ordre 2:

$$\Phi = \frac{1}{2} \times [Y^{n+1} + Y^n]$$



- Adams-Bashforth // Ordre 2:

$$\Phi = \frac{1}{2} \times [3Y^n - Y^{n-1}]$$



Augmentation de l'ordre

- Il suffit de prendre un développement de Taylor plus grand ...

$$y^{n+1} = y^n + \frac{dy}{dt} \Delta t + \frac{d^2 y}{dt^2} \frac{\Delta t^2}{2} + \dots + \frac{d^p y}{dt^p} \frac{\Delta t^p}{p!} + O(\Delta t^{p+1})$$

- Et de définir Φ comme: 

$$\Phi = \frac{dy}{dt}(t^n, y^n) + \frac{d}{dt} \left[\frac{dy}{dt} \right] (t^n, y^n) \frac{\Delta t}{2} + \dots$$

Augmentation de l'ordre

- En utilisant l'EDO vérifiée par y :

$$\frac{dy}{dt}(t^n) = Y(t^n, y^n) = Y^n$$

$$\frac{d^2 y}{dt^2}(t^n) = \frac{dY}{dt}(t^n, y^n) = \left[\frac{\partial Y}{\partial t} + \frac{\partial Y}{\partial y} \frac{dy}{dt} \right] = [Y_t^n + Y_y^n Y^n]$$

Y dérivé par t pris en t^n
et y^n

$$\frac{d^3 y}{dt^3}(t^n) = [Y_{tt}^n + Y_{yt}^n Y^n + (Y^n Y_y^n + Y_t^n) Y_y + Y^n Y_{ty}^n + Y^{n^2} Y_{yy}^n]$$

- Et donc: 

$$\Phi = Y^n + [Y_t^n + Y_y^n Y^n] \frac{\Delta t}{2} +$$

$$+ [Y_{tt}^n + Y_{yt}^n Y^n + (Y^n Y_y^n + Y_t^n) Y_y + Y^n Y_{ty}^n + Y^{n^2} Y_{yy}^n] \frac{\Delta t^2}{6} + \dots$$

Méthodes à pas multiples

- Plutôt que d'estimer des dérivées d'ordre élevées, il est préférable de mieux choisir les endroits où on évalue Y
- On pose pour cela

$$\Phi = A_1 \underbrace{Y(t^n, y^n)}_{Y^n} + A_2 Y(t^n + \alpha \Delta t, y^n + \beta \Delta t)$$

où A_1 , A_2 , α et β sont des paramètres à déterminer

- Par identification jusqu'à l'ordre 1 inclus on obtient:

$$Y^n + [Y_t^n + Y_y^n Y^n] \frac{\Delta t}{2} = (A_1 + A_2) Y^n + A_2 [\alpha \Delta t \times Y_t^n + \beta \Delta t \times Y_y^n]$$

$$\boxed{A_2 \alpha = \frac{1}{2} \quad A_2 \beta = \frac{Y^n}{2} \quad A_1 + A_2 = 1}$$



Runge-Kutta d'ordre 2

- Schéma à deux « étapes » ($A_1=0, A_2=1$):

$$k_1 = Y(t^n, y^n) = Y^n$$

$$k_2 = Y\left(t^n + \frac{\Delta t}{2}, y^n + \frac{k_1 \Delta t}{2}\right)$$

$$y^{n+1} = y^n + \Delta t \times k_2$$

Runge-Kutta ordre 4

- On peut reprendre la même démarche en conservant plus de termes dans le développement de Taylor. A l'ordre 4 on obtient la méthode à 4 pas suivante:

$$k_1 = Y(t^n, y^n) = Y^n$$

$$k_2 = Y\left(t^n + \frac{\Delta t}{2}, y^n + \frac{\Delta t}{2} k_1\right)$$

$$k_3 = Y\left(t^n + \frac{\Delta t}{2}, y^n + \frac{\Delta t}{2} k_2\right)$$

$$k_4 = Y(t^n + \Delta t, y^n + \Delta t k_3)$$

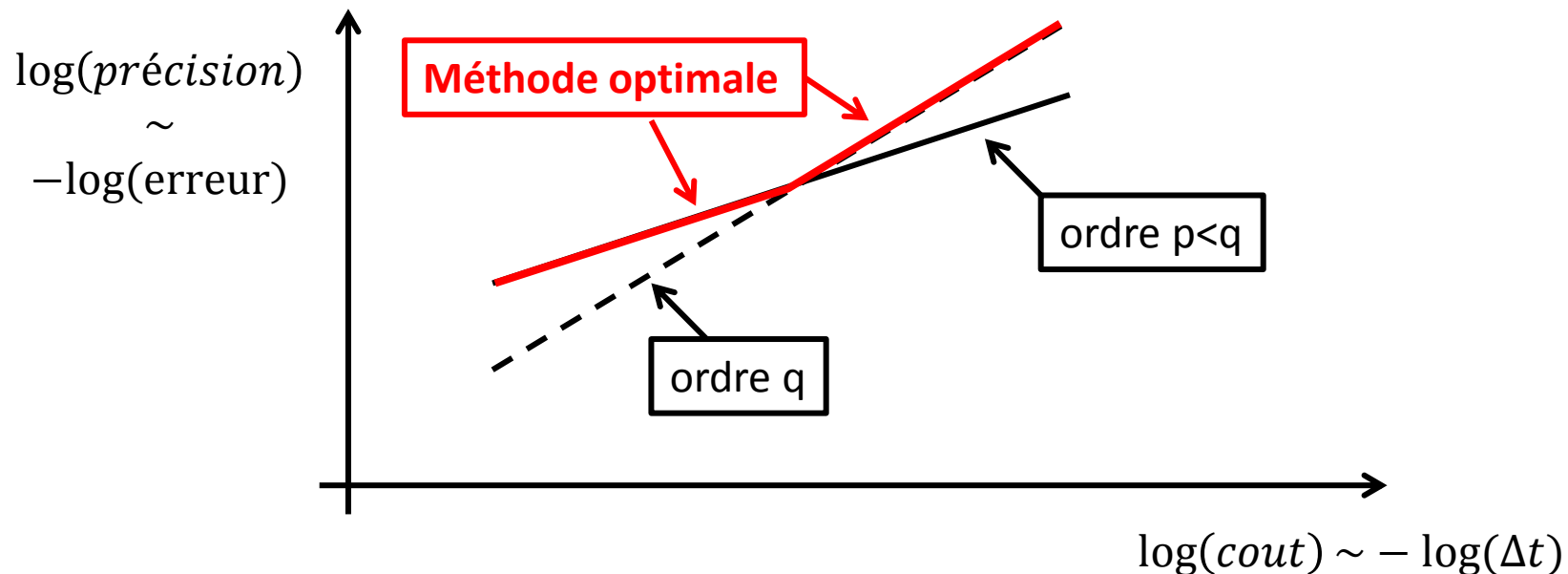
$$y^{n+1} = y^n + \frac{\Delta t}{6} \times [k_1 + 2k_2 + 2k_3 + k_4]$$

Mesure numérique de l'ordre

- Lorsque l'on utilise un logiciel de résolution d'une eq. diff., il est utile de pouvoir vérifier ses performances
- Il est possible de « mesurer » à partir d'une expérience numérique l'ordre effectif de la méthode implémentée
- Partant de la définition de l'ordre p , on peut déduire que $\Phi_{exact} - \Phi = \alpha \Delta t^p$ où α est une constante
- On en déduit que $\log|\Phi_{exact} - \Phi| = \log\alpha + p \log \Delta t$
- L'ordre effectif d'une méthode peut donc être mesuré en traçant dans un diagramme logarithmique l'erreur obtenue pour un problème donné et pour différentes valeurs du pas de temps; la pente de la droite obtenue est égale à l'ordre recherché.

Relation précision-coût

- Le coût d'une itération dépend du nombre de fois que de la fonction Y doit être calculée: n fois pour une méthode d'ordre n
- Ne pas déduire qu'une méthode est d'autant plus lente que son ordre est élevé; tout dépend de la précision désirée ...
- Le coût total dépend du nombre d'itérations, donc de $1/\Delta t$



Réalisations sous Matlab

1. Créer un programme qui résout une EDO de type $dy/dt = Y(t,y)$ où Y est une fonction fournie sous forme d'une fonction Matlab indépendante. On commencera par n'implémenter que les méthodes explicites vues en cours
2. Mesurer numériquement l'ordre des méthodes Euler explicite, Adams Bashforth, Runge-Kutta ordre 2, Runge-Kutta ordre 4
3. Implémenter une méthode implicite (ex: Euler implicite) et identifier la difficulté qui vous empêche de le faire dans le cas général d'une fonction $Y(t,y)$ quelconque
4. Terminer l'implémentation des méthodes Euler implicite et Crank-Nicolson dans le cas particulier où $Y(t,y)=y(t)^2$. Mesurer numériquement leur ordre de convergence