


Resolution Numerique de systemes differentiels

II. Méthodes d'intégration numérique

II.1. Cadre théorique général

1.1. Théories physiques et EDO (Équations Différentielles Ordinaires)

- Les *Théories Physiques* proposent des modèles descriptifs simples (*lois*) pour représenter l'évolution de systèmes complexes soumis à des contraintes extérieures.
- Ces lois décrivent à la fois le comportement du système et l'action des contraintes (*cf Cours n°1*)
 - lois de Newton : $m \cdot \vec{a}(t) = \Sigma \vec{F}(t)$
 - lois de Kepler (*référentiel non galiléen*)
 - dynamique du solide
 - milieux granulaires
 - visco-élasticité
 - dynamique des fluides incompressibles
 - dynamique des fluides compressibles
 - physique statistiques
 - électro-magnétisme
 -
- Chaque 'loi' décrit, de manière plus ou moins fine, une *partie* de la réalité physique, valable dans un cadre restreint
-  Induit des effets de bords : notion d'infini, "raccordement" des modèles (exp. solides/grains/fluides)
- **Seule constante** : toutes ces lois permettent de formuler l'évolution *infinitésimale* d'un système physique sous la forme de *systèmes différentiels*.

L'objectif devient alors de *résoudre* ces systèmes différentiels pour connaître le comportement général du système physique.

1.2. Le problème de Cauchy

Soit à résoudre une Équation Différentielle Ordinaire (EDO) avec conditions initiales :

$$\begin{cases} x(a) = x_0 \\ \dot{x}(t) = f(t, x(t)), \forall t \in [a, b] \end{cases} \quad \begin{array}{ll} x(t) \in \mathbb{R}^n & \text{grandeur physique dont on veut suivre l'évol. selon } t \\ x_0 \in \mathbb{R}^n & \text{condition initiale (état connu du système)} \\ [a, b] & \text{intervalle temporel d'étude} \\ f : [a, b] \times \mathbb{R}^n \rightarrow \mathbb{R}^n & \text{modèle descriptif infinitésimal (syst. + contraintes)} \end{array} \quad (13)$$

Hormis quelques cas simples (cf. tir balistique), la solution analytique est inaccessible. On va donc chercher à calculer *pas-à-pas* des valeurs approchées de la solution.

1.3. Processus approché

a) L'opérateur de dérivation (anticipée)

C'est l'expression des variations infinitésimales d'une grandeur x en fonction du paramètre t :

$$\dot{x}(t) = \frac{dx}{dt} = \lim_{dt \rightarrow 0} \frac{x(t + dt) - x(t)}{dt} \quad (14)$$

En pratique la condition limite ($dt \rightarrow 0$) est irréalisable \Rightarrow on fixe une valeur h^3 *suffisamment* petite. L'équation (14) devient alors :

$$\dot{x}(t) = \frac{x(t+h) - x(t)}{h} + \epsilon_0(t, h) \quad (\text{où } \epsilon_0(t, h) \text{ est l'erreur d'approximation}) \quad (15)$$

L'enjeu est alors d'évaluer l'erreur $\epsilon_0(t, h)$: si elle reste faible par rapport à $x(t)$, l'approximation est fiable.

³ h est appelé *pas d'échantillonnage temporel*

b Équation différentielle approchée

L'équation différentielle (13) s'écrit alors :

$$x(t+h) = x(t) + h.f(t, x(t)) + \epsilon_1(t, h) \quad \text{où } \epsilon_1(t, h) = h.\epsilon_0(t, h) \quad (16)$$

Si on néglige l'erreur d'approximation $\epsilon_1(t, h)$, on construit le processus approché $\tilde{x}(t)$ défini par :

$$\begin{cases} \tilde{x}(a) = x_0 \\ \tilde{x}(t+h) = \tilde{x}(t) + h.f(t, \tilde{x}(t)) \end{cases} \quad (17)$$

⇒ $\tilde{x}(t)$ est une valeur approchée de $x(t)$

⇒ f est évaluée en t avec cette valeur fautive : $f(t, \tilde{x}(t)) = f(t, x(t) + \epsilon_1(t, h)) = f(t, x(t)) + \epsilon_2(t, h)$

Le processus approché $\tilde{x}(t)$ ainsi construit contient donc deux sources d'erreur l'une (ϵ_1) résultant de l'approximation sur l'opérateur de dérivation, l'autre (ϵ_2) de l'effet de f sur ϵ_1 :

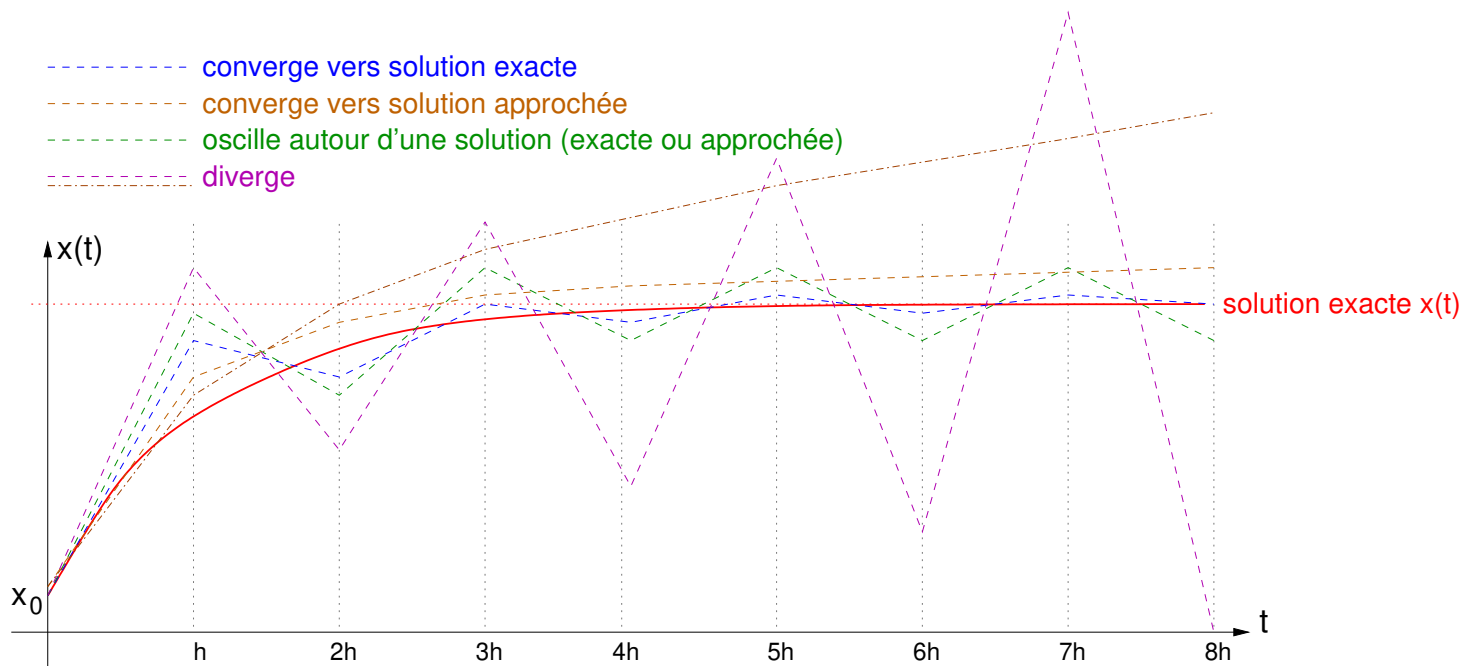
$$\begin{cases} \tilde{x}(a) = x_0 \\ \tilde{x}(t) = x(t) + (\epsilon_1(t, h) + \epsilon_2(t, h)) \end{cases} \quad (18)$$

Le comportement du processus numérique $\tilde{x}(t)$ dépend donc surtout du comportement de l'erreur d'approximation globale $\epsilon(t) = \epsilon_1(t, h) + \epsilon_2(t, h)$.

A cela pourra s'ajouter une troisième erreur due à l'approximation sur les calculs en virgule flottante sur machine (cf. norme IEEE pour le codage des réels), beaucoup plus difficile à évaluer et mal connue.

C Plusieurs comportements possibles

Selon que l'erreur globale converge, reste bornée ou diverge, le processus approché peut être stable (mais pas forcément exact) ou instable (divergent).



Les différentes méthodes de résolution numérique que l'on peut envisager se distinguent avant tout par la forme que prend l'erreur d'approximation globale.

Certaines seront **inconditionnellement stables**⁴ (pas de divergence quel que soit le pas temporel h) d'autres potentiellement **instables** (selon la valeur de h).

⁴ ça ne veut pas dire qu'elle sont «meilleures»... - cf. plus tard

d Formulation de Taylor

Développement en série de Taylor pour une fonction $x(t)$ de classe C^∞ (infiniment dérivable) :

$$\forall(t, h) \quad x(t+h) = \sum_{k=0}^{\infty} \frac{x^{(k)}(t)}{k!} \cdot h^k = x(t) + h \cdot \dot{x}(t) + \frac{h^2}{2} \ddot{x}(t) + \frac{h^3}{6} \dddot{x}(t) + \dots \quad (19)$$

A l'ordre 1, le développement de Taylor s'écrit : $x(t+h) = x(t) + h \cdot \dot{x}(t) + o(h^2)$

L'EDO (13) s'écrit donc comme précédemment (cf. eq.(16)):

$$x(t+h) = x(t) + h \cdot f(t, x(t)) + o(h^2) \quad \text{avec cette fois} \quad o(h^2) = \left(\sum_{k=2}^{\infty} \frac{x^{(k)}(t)}{k!} \cdot h^k \right) \quad (20)$$

Définition : Méthode d'Ordre k

Une méthode d'intégration numérique sera dite **d'ordre k** si l'erreur d'approximation sur l'opérateur de dérivation est proportionnelle à h^{k+1} - ie : $\epsilon(h) = o(h^{k+1})$

Ne pas confondre avec l'ordre d'un système différentiel (à suivre...).

1.4. Formulation Intégrale

a Le problème de Cauchy

- On reprend l'eq. : (13), que l'on reformule par une intégration sur l'intervalle $[t, t + h]$

$$\dot{x}(t) = f(t, x(t)) \implies \int_t^{t+h} \dot{x}(t) dt = \int_t^{t+h} f(t, x(t)) dt \implies \frac{x(t+h) - x(t)}{h} = \int_t^{t+h} f(t, x(t)) dt$$

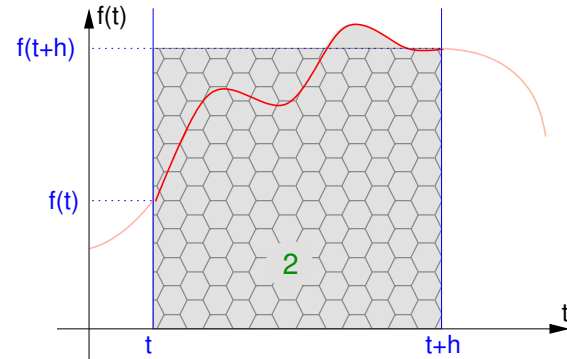
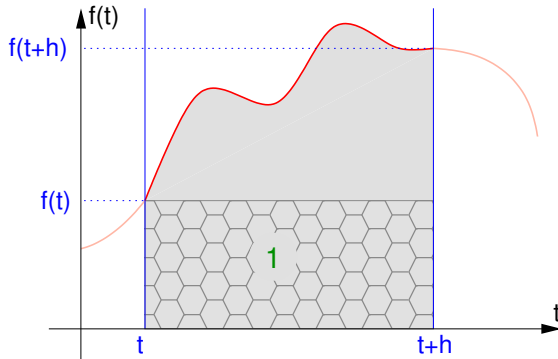
$$\implies \text{formulation intégrale de l'eq. (13) : } x(t+h) = x(t) + h \int_t^{t+h} f(t, x(t)) dt \quad (21)$$

- On obtient une approx. de $x(t+h)$ à partir d'une approx. de l'intégrale $\mathcal{A}(t, x(t)) = \int_t^{t+h} f(t, x(t)) dt$

b Approximation d'intégrale (méthode des rectangles)

(1) $\mathcal{A}(t, x(t)) = h \cdot f(t, x(t))$
 $\implies x(t+h) = x(t) + h \cdot f(t, x(t))$

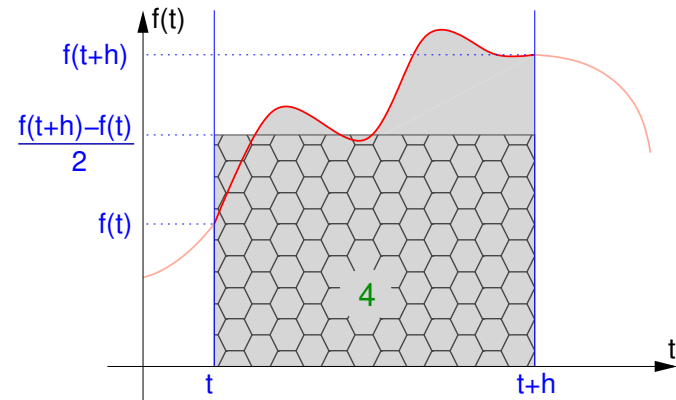
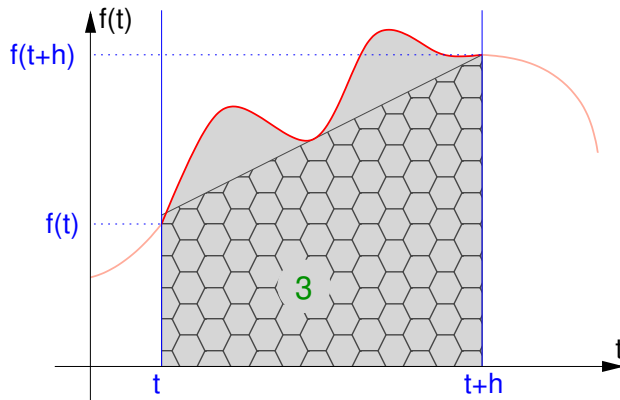
(2) $\mathcal{A}(t, x(t)) = h \cdot f(t+h, x(t+h))$
 $\implies x(t+h) = x(t) + h \cdot f(t+h, x(t+h))$



c Approximation d'intégrale (méthode des trapèzes)

$$\mathcal{A}(t, x(t)) = h \cdot \frac{1}{2} \cdot (f(t+h, x(t+h)) - f(t, x(t)))$$

$$\Rightarrow x(t+h) = x(t) + \frac{h}{2} \cdot (f(t+h, x(t+h)) - f(t, x(t)))$$



d Approximation en 2 passes («midpoint method» revisitée)

(a) $y(t+h) = x(t) + h \cdot f(t, x(t))$

(b) $x(t+h) = x(t) + h \cdot f(t+h, y(t+h))$

Avantage :

on peut évaluer à chaque instant la «fiabilité» de la solution : $|x(t+h) - y(t+h)|$ doit rester «faible».

1.5. Problèmes d'ordre supérieur

Définition : Problème d'Ordre k

Une EDO (problème) est dite **d'ordre k** si elle met en jeu une fonction $x(t)$ et sa dérivée d'ordre k .

En général le problème est défini par une relation du type : $x^{(k)}(t) = \phi(t, x(t), \dot{x}(t), \dots, x^{(k-1)}(t))$

Dans la plupart des cas classiques, les syst. diff. en jeux sont d'ordre 2 (cf. équation de Newton). Ils se décomposent simplement en deux problèmes d'ordre 1 liés :

$$\left\{ \begin{array}{l} x(a) = x_0 \\ \dot{x}(a) = \dot{x}_0 \\ \ddot{x}(t) = \phi(t, x(t), \dot{x}(t)) \end{array} \right. \Rightarrow \left\{ \begin{array}{l} \text{(a)} \left\{ \begin{array}{l} \dot{x}(a) = \dot{x}_0 \\ \ddot{x}(t) = g(t, \dot{x}(t)) \end{array} \right. \\ \text{(b)} \left\{ \begin{array}{l} x(a) = x_0 \\ \dot{x}(t) = f(t, x(t)) \end{array} \right. \end{array} \right. \quad (22)$$

(a) produit un processus approché $\tilde{x}(t)$ au premier ordre

(b) travaille sur ce proc. approché ($\tilde{\dot{x}}(t) = f(t, x(t))$) pour produire une approx. au second ordre $\tilde{\tilde{x}}(t)$

Les problèmes de stabilité précédents restent les mêmes pour chacun des deux sous-systèmes (les erreurs d'approximation peuvent s'ajouter ou se compenser) mais rien n'oblige à adopter la même méthode numérique dans les deux cas (cf. méthode Leapfrog (p.31)).

II.2. Schémas d'intégration : les grands classiques

2.1. EDO et processus numérique

a Génération vs. échantillonnage (principe fondamental de la simulation)

- Une EDO est un *processus fonctionnel continu* : $\dot{x}(t) = f(t, x(t))$

☞ Echantillonnage :

- correspond à un *découpage* de la solution⁵ continue $x(t)$ sur un intervalle $[a, b]$:

- $x_0 = x(a)$
- $x_n = x(a + n \cdot \frac{b-a}{N})$ ou N est le nombre d'échantillons souhaité.

☞ en réalité $x_n = \lceil x(a + n \cdot \frac{b-a}{N}) \rceil$: valeur réelle quantifiée sur un format numérique réduit.

☞ Génération :

- Ici on va générer un *processus numérique discret* $x_{n+1} = f((x_k)_{n-p \leq k < n})$
- Le comportement numérique est donc celui d'une suite récurrente d'ordre⁶ p , donné par le schéma d'approximation de l'opérateur de dérivation.
- La correspondance doit être telle que *si $x_n \approx x(t)$ alors $x_{n+1} \approx x(t+h)$* :
 - $x_0 = x(a)$
 - $\forall n > 0 \quad x_n \approx x(a + n \cdot h)$ ou h est le pas de discrétisation.

☞ On cherche à générer une suite numérique censée rester proche d'un échantillonnage d'une solution continue que l'on ne connaît pas.

Il faut donc être en mesure d'évaluer la fiabilité de cette correspondance

⁵supposée connue

⁶encore une définition d'*ordre* !!

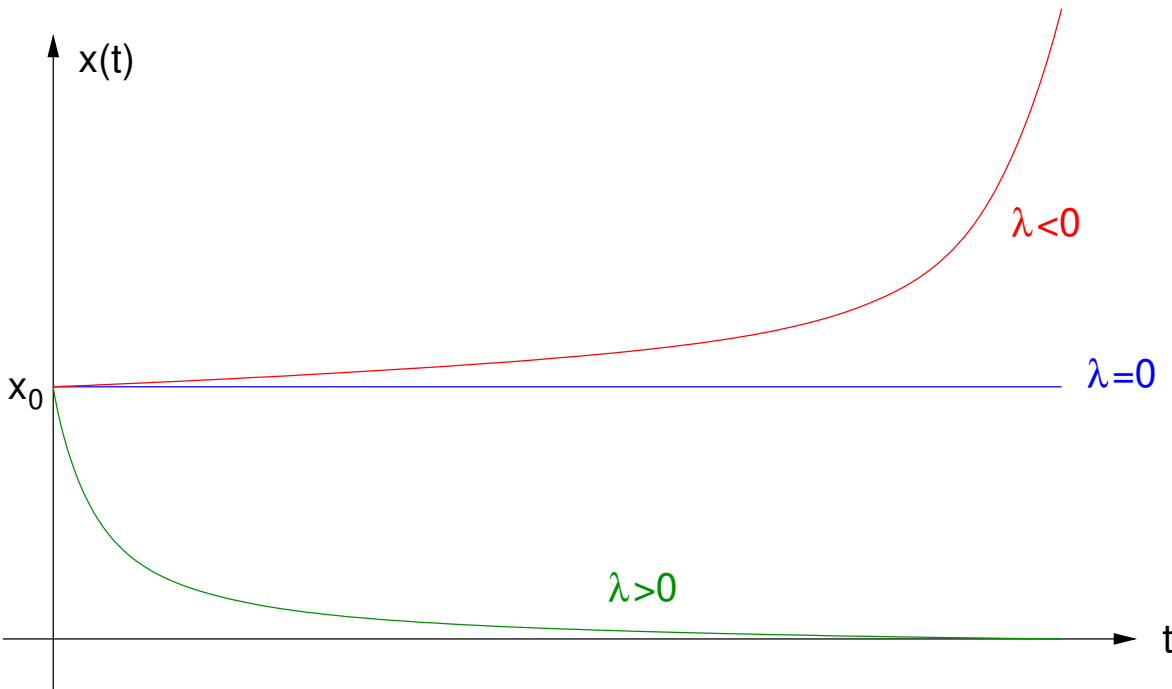
b L'équation de Dahlquist (test de stabilité)

Pour évaluer la stabilité d'une méthode d'intégration on utilise en général une fonction linéaire complexe définie par $f(t, x(t)) = -\lambda \cdot x(t)$ (où $\lambda \in \mathbb{C}$).

On obtient le système diff. suivant dont la solution analytique est connue :

$$\begin{cases} x(a) = x_0 \\ \dot{x}(t) + \lambda \cdot x(t) = 0, \quad \forall t \in [a, b] \end{cases} \implies x(t) = x_0 e^{-\lambda t} \quad (23)$$

Solutions dans le cas réel ($\lambda \in \mathbb{R}$) :



2.2. Schéma de Euler Explicite (dérivation anticipée / évaluation en fin d'impulsion)

C'est le schéma d'intégration numérique le plus simple à mettre en œuvre, directement déduit des éq. (16) et (20), avec l'opérateur de dérivation anticipée

$$\dot{x}(t) = \frac{dx}{dt} = \lim_{dt \rightarrow 0} \frac{x(t+dt) - x(t)}{dt} \Rightarrow \dot{x}(t) = \frac{x(t+h) - x(t)}{h} \quad (24)$$

Il correspond à la création du processus numérique approché de l'éq. (17), ou à celui obtenu grâce à la première méthode des rectangles (1).

a Mise en œuvre

On définit ainsi le processus numérique $(x_k = x(a + k.h))_{0 \leq k < n}$ par :

$$\begin{cases} x_0 = x_0 \\ x_{k+1} = x_k + h.f(t_k, x_k) \end{cases} \quad (25)$$

b Stabilité

La méthode Euler Explicite appliquée à l'équation de Dahlquist fournit une solution complète sous la forme d'une suite géométrique de raison $(1 - \lambda h)$:

$$x_{k+1} = x_k - h.\lambda.x_k = (1 - h.\lambda).x_k \implies \forall k \in \mathbb{N}, x_k = x_0.(1 - \lambda h)^k \quad (26)$$

Cette suite numérique reste bornée (donc stable) si et seulement si ($|1 - \lambda h| \leq 1$)

- hypothèse : $\lambda \in \mathbb{R}^{*+}$ i.e $\lambda > 0$ ($x(t)$ converge vers 0)
- solution : $\{x_k = x_0 \cdot (1 - \lambda h)^k\}_{k \in \mathbb{N}}$
- stabilité : $\Leftrightarrow |1 - \lambda h| \leq 1$

$$|1 - \lambda h| < 1 \Leftrightarrow -1 < 1 - \lambda h < +1 \Leftrightarrow h < \frac{2}{\lambda} \quad (27)$$

- La méthode numérique **Euler Explicite** est stable (converge vers la solution analytique) si et seulement si le pas d'échantillonnage est tel que $h < \frac{2}{\lambda}$ (pour l'équation de Dahlquist)
- Cette méthode est dite **conditionnellement stable**.
- On peut toujours stabiliser **Euler Explicite** en réduisant le pas temporel, mais c'est au prix de calculs supplémentaires.
- Il faut essayer de travailler au plus près de la valeur limite : économique mais dangereux^a)....
- Trouver la limite de stabilité est en général **très** difficile, même pour un problème linéaire
- Dans le cas non linéaire, cette limite devient variable

^acf. la suite : réseaux masses/ressorts

2.3. Schéma de Euler Implicite (évaluation en début d'impulsion)

C'est une variante de la méthode précédente avec l'expression de l'opérateur de dérivation retardée:

$$\dot{x}(t) = \frac{dx}{dt} = \lim_{dt \rightarrow 0} \frac{x(t) - x(t - dt)}{dt} \Rightarrow \dot{x}(t) = \frac{x(t) - x(t - h)}{h} \quad (28)$$

a mise en œuvre

L'équation différentielle (13): $(\dot{x}(t) = f(t, x(t)))$ devient alors (cf. eq.(20)) :

$$x(t + h) = x(t) + h \cdot f(t + h, x(t + h)) + o(h^2) \quad (29)$$

et on définit comme précédemment le processus numérique approché $(x_k)_{0 \leq k < n}$ par :

$$\begin{cases} x_0 = x_0 \\ x_{k+1} = x_k + h \cdot f(t_{k+1}, x_{k+1}) \end{cases} \quad (30)$$

La formulation devient beaucoup plus complexe à traiter du fait que la fonction f est évaluée en (x_{k+1}) ... que l'on est justement en train de calculer...

La valeur de (x_{k+1}) est utilisée **implicitement** dans le calcul.

Il faudra donc *extraire* x_{k+1} de l'expression de $f(t_{k+1}, x_{k+1})$, soit, pour schématiser, *factoriser* cette expression sous la forme $f(t_{k+1}, x_{k+1}) = x_{k+1} \cdot \Phi(t_{k+1})$ où $\Phi(t_{k+1})$ ne dépend plus de x_{k+1} .

Alors, en notant $\Psi(t_{k+1}) = 1 - h \cdot \Phi(t_{k+1})$, l'eq.(30) s'écrit $x_{k+1}(1 - h \cdot \Phi(t_{k+1})) = x_k$ soit $x_{k+1} = \Psi^{-1}(t_{k+1}) \cdot x_k$

b Stabilité

On utilise encore l'équation de Dahlquist ($\dot{x}(t) + \lambda x(t) = 0$, $\lambda \in \mathbb{C}$)

La méthode **Euler Implicite** fournit encore une solution complète sous la forme d'une suite géom. :

$$x_{k+1} = x_k - h \cdot \lambda \cdot x_{k+1} \implies x_{k+1} = \frac{1}{1 + \lambda h} x_k \implies \forall k \in \mathbb{N}, x_k = x_0 \cdot \left(\frac{1}{1 + \lambda h} \right)^k \quad (31)$$

- Cette suite numérique reste bornée (stable) si et seulement si $\left| \frac{1}{1 + \lambda h} \right| \leq 1$
- Dans l'hypothèse ($\lambda \in \mathbb{R}^{++}$), cette condition est **toujours** vérifiée.

➤ La méthode numérique **Euler Implicite** est donc **inconditionnellement stable**.

✚ Le principal problème est d'*extraire* (x_{k+1}) de l'expression implicite $f(t_{k+1}, x_{k+1})$
Hormis quelques cas très simples, cela revient à inverser un système (donc des matrices, pas toujours constantes \Rightarrow méthodes itératives coûteuses, potentiellement instables).

➤ D'un point de vue théorique, **Euler Implicite** est meilleure que **Euler Explicite** puisqu'elle reste stable et assure une convergence quel que soit le pas d'échantillonnage.
En outre la convergence est beaucoup plus rapide qu'avec une méthode **explicite** puisque le processus numérique *anticipe* le comportement de la solution réelle.

✚ D'un point de vue pratique, sa mise en œuvre effective peut s'avérer difficile et le gain obtenu sur le pas d'échantillonnage est compensé par le surcoût de calcul de l'inversion du système.

- Enfin, une méthode **Implicite** introduit en général des «effets de bords» qui peuvent apparaître comme bénéfiques ou nuisibles selon ce que l'on cherche à simuler (cf.(p.42) : oscillateur élémentaire / (p.56) : modèle modal)

🔗 Le choix d'une méthode ou d'une autre dépend avant tout de la situation.

2.4. Le schéma Leapfrog : une solution intermédiaire

Aussi connu sous le nom de **schéma de Störmer-Verlet** (entre autres), le **Leapfrog** (lit. «saute-mouton») ne fonctionne que pour les systèmes différentiels d'ordre 2 (majorité des cas en simulation dynamique).

a Principe et mise en œuvre

Il consiste à utiliser des schémas différents pour les 1^o et 2^o intégrations :

- Evaluation de l'accélération (bilan dyn.) en fin d'impulsion : *schéma explicite*

$$\ddot{x}(t) = (\dot{x}(t+h) - \dot{x}(t)) / h \Rightarrow \dot{x}(t+h) = \dot{x}(t) + h\ddot{x}(t) \Rightarrow v_{n+1} = v_n + h.a_n$$

- Evaluation de la vitesse en début d'impulsion : *schéma implicite*

$$\dot{x}(t) = (x(t) - x(t-h)) / h \Rightarrow x(t+h) = x(t) + h\dot{x}(t+h) \Rightarrow x_{n+1} = x_n + h.v_{n+1}$$

On obtient les processus numériques auto-évolutifs générant les états pas-à-pas :

impuls. n	calculs			impuls. (n+1)
état connu	bilan dyn.	1 ^o intégration	2 ^o intégration	état connu
x_n	$a_n = \phi(x_n, v_n)$	$v_{n+1} = v_n + h.a_n$	$x_{n+1} = x_n + h.v_{n+1}$	x_{n+1}
v_n				v_{n+1}

Bien que **conditionnellement stable** (cf. *plus tard*) comme **Euler Explicite**, elle s'avère plus stable que celle-ci.

b Version condensée

En travaillant avec la position retardée (x_{n-1}) plutôt que la vitesse ($v_n = (x_n - x_{n-1})/h$) on obtient une forme condensée.

L'EDO du second ordre $\ddot{x}(t) = f(t, x(t), \dot{x}(t))$ fournit un processus numérique **explicite** (i.e. aucun terme d'ordre $(n+1)$ à droite) du second ordre:

$$\begin{cases} \ddot{x}_n = (\dot{x}_{n+1} - \dot{x}_n)/h = \Phi(x_n, \dot{x}_n) \\ \dot{x}_n = (x_n - x_{n-1})/h \end{cases} \Rightarrow x_{n+1} = 2x_n - x_{n-1} + h^2 \cdot \Phi(x_n, x_{n-1}) \quad (32)$$

La boucle de simulation s'écrit alors très simplement, en 2 étapes :

$$\left| \begin{array}{l} \text{conditions initiales : } (x_0, v_0) \\ \Rightarrow x_1 = x_0 + h * v_0 \end{array} \right. \quad \forall n > 1 \quad \left| \begin{array}{l} 1. \quad a_n = \Phi(x_n, x_{n-1}) \quad \text{bilan dynamique au pas } n \\ 2. \quad x_{n+1} = 2x_n - x_{n-1} + h^2 \cdot a_n \quad \text{mise à jour des position au pas } (n+1) \\ \text{GOTO 1.} \end{array} \right. \quad (33)$$

Dans l'exemple du tir balistique avec frottement cinétique (de coeff. α) et gravité $\vec{G} \Big|_{-g}^0$:

- bilan dyn. : $\vec{a}_n = \frac{1}{m} \sum \vec{F}_n = \frac{1}{m} (m \cdot \vec{G} - \alpha \cdot \vec{v}_n) = \vec{G} - \frac{\alpha}{h \cdot m} (\vec{x}_n - \vec{x}_{n-1})$
- intégrations : $\vec{x}_{n+1} = (2 - \frac{\alpha \cdot h}{m}) \cdot \vec{x}_n + (\frac{\alpha \cdot h}{m} - 1) \cdot \vec{x}_{n-1} + h^2 \cdot \vec{G}$

2.5. Schémas d'ordre supérieur *RK4 et quelques autres*

a Runge-Kutta d'ordre 4 *RK4*

Méthode basée sur une approximation à l'ordre 4 du développement de Taylor de la dérivée :

$$\dot{x}(t) = f(t, x(t)) \Rightarrow x(t+h) = x(t) + h.f(t, x(t)) + \frac{h^2}{2} \frac{df(t, x(t))}{dt} + \frac{h^3}{6} \frac{d^2f(t, x(t))}{dt^2} + \frac{h^4}{24} \frac{d^3f(t, x(t))}{dt^3} + o(h^4)$$

b Mise en œuvre

La valeur *intégrée* $x(t+h)$ nécessite quelques calculs de «pentes» intermédiaires :

- $k_1 = h.f(t, x(t))$: pente au début de l'intervalle
- $k_2 = h.f(t + \frac{h}{2}, x(t) + \frac{k_1}{2})$: pente «avant» au milieu de l'intervalle
- $k_3 = h.f(t + \frac{h}{2}, x(t) + \frac{k_2}{2})$: pente «arrière» au milieu de l'intervalle
- $k_4 = h.f(t+h, x(t) + k_3)$: pente à la fin de l'intervalle

et finalement : $x(t+h) = x(t) + \frac{1}{6} (k_1 + 2k_2 + 2k_3 + k_4)$

c Et quelques autres

- Crank-Nicholson (2): $x(t+h) = x(t) + \frac{h}{2} (f(t, x(t)) + f(t+h, x(t+h)))$
- Adams-Bashford (2): $x(t+h) = x(t) + h \left(\frac{3}{2}f(t, x(t)) - \frac{1}{2}f(t-h, x(t-h)) \right)$
- Adams-Bashford (3): $x(t+h) = x(t) + h \left(\frac{23}{12}f(t, x(t)) - \frac{16}{12}f(t-h, x(t-h)) + \frac{5}{12}f(t-2h, x(t-2h)) \right)$
- Adams-Moulton (3): $x(t+h) = x(t) + h \left(\frac{5}{12}f(t+h, x(t+h)) + \frac{8}{12}f(t, x(t)) - \frac{1}{12}f(t-h, x(t-h)) \right)$
- etc...