

Chapitre 6: flot numérique et conditions d'ordre

Philippe Chartier

26 novembre 2013

On considère le problème de Cauchy sous forme autonome :

$$\begin{cases} \dot{y}(t) &= f(y(t)) \\ y(t_0) &= y_0 \end{cases}, \quad (0.1)$$

où f est une fonction définie sur \mathbb{R}^d et y_0 un point de \mathbb{R}^d . On suppose en outre que f est continue et **globalement** Lipschitzienne, de sorte que pour tout $y_0 \in \mathbb{R}^d$, le système (0.1) admet une solution **globale unique** sur \mathbb{R} . L'application flot exact $\varphi_t(y)$ est ainsi définie pour tout $(t, y) \in \mathbb{R} \times \mathbb{R}^d$. On suppose en outre dans tout ce chapitre que f est suffisamment régulière, par exemple ici que f **est indéfiniment différentiable**.

De très nombreux problèmes applicatifs sont mis en équation à l'aide d'un système d'équations différentielles de la forme (0.1) (voir par exemple le mouvement à deux corps dans le cours d'introduction). Au moment de l'approximation numérique, il est nécessaire (connaissant une valeur initiale) de pouvoir calculer une ou plusieurs valeurs $y(T_1), y(T_2) \dots$. L'objet de ce cours est l'étude des méthodes qui permettent d'opérer ce calcul (à l'aide d'un ordinateur). On va aussi répondre aux questions suivantes : existe-t-il des méthodes arbitrairement précises ? Quelles sont les conditions à imposer pour que la solution numérique approche la solution exacte à un certain ordre ?

1 Définition du flot numérique et de l'ordre local

L'exemple le plus élémentaire est la méthode d'Euler : on subdivise l'intervalle $[0, T]$ en $0 = t_0 < t_1 < \dots < t_N = T$, $t_i = t_0 + nh$ et, pour t voisin de t_n , on utilise les approximations

$$\dot{y}(t) \approx \frac{y(t_{n+1}) - y(t_n)}{h}$$

et

$$f(y(t)) \approx f(y(t_n))$$

En reportant dans l'équation différentielle, on aboutit à la méthode d'Euler :

$$y_{n+1} = y_n + hf(y_n)$$

où $y_n = y(t_n)$. En commençant avec y_0 pour $n = 0$, on calcule de proche en proche (récursivement) y_1, y_2, \dots jusqu'à y_N qui est sensé fournir une valeur approchée de $\varphi_T(y_0) = y(T)$. On verra que, lorsque $h \rightarrow 0$, on a $y_N \rightarrow y(T)$. Cela montre l'existence d'une méthode d'approximation numérique.

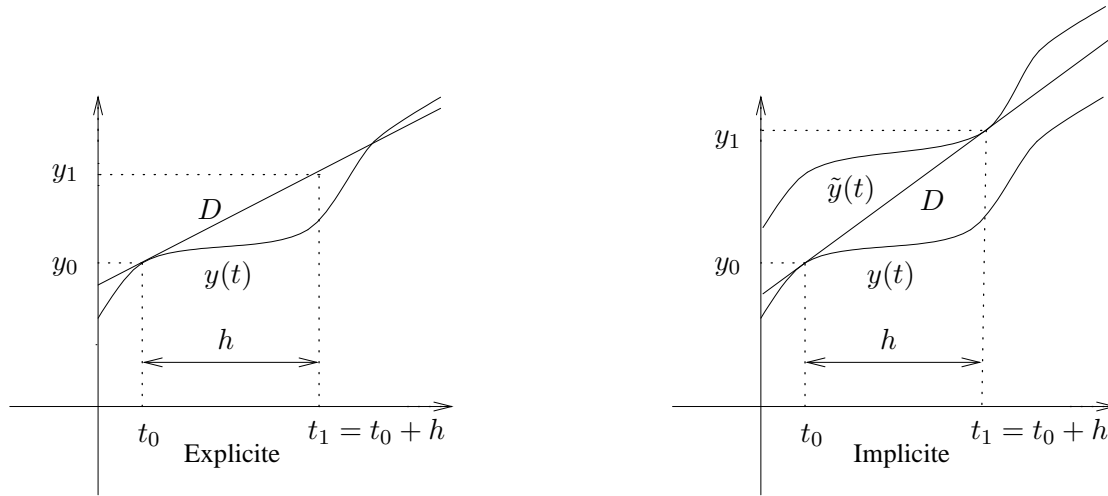


FIGURE 1 – Méthodes d'Euler

Définition 1.1 On appelle *flot numérique* une application ϕ_h définie sur $I \times \mathbb{R}^d$ à valeurs dans \mathbb{R}^d où I est un intervalle ouvert contenant 0, telle que, pour tout $h \in I$ et tout $y \in \mathbb{R}^d$, $\phi_h(y)$ soit une approximation de $\varphi_h(y)$.

Exemple 1.2 Pour la méthode d'Euler explicite définie ci-avant, on a

$$\phi_h(y) = y + hf(y)$$

Le flot numérique associé à la méthode d'Euler implicite (voir figure 1) est quant à lui donné par l'équation

$$\phi_h(y) = y + hf(\phi_h(y))$$

Définition 1.3 Soit ϕ_h un flot numérique. On dit que ϕ_h est d'ordre p si il existe une fonction ϵ de $I \times \mathbb{R}^d$ dans \mathbb{R}^d où I est un intervalle ouvert de \mathbb{R} contenant 0, **continue** et telle que

$$\phi_h(y) = \varphi_h(y) + h^{p+1}\epsilon(h, y) \quad (1.2)$$

1.1 Méthodes de Runge-Kutta

Les méthodes de Runge-Kutta sont une généralisation possible (comme le sont par ailleurs les méthodes multipas dont nous ne traiterons pas dans ce cours) de la méthode d'Euler explicite.

Définition 1.4 Soient $b \in \mathbb{R}^s$ et $A \in \mathcal{M}_s(\mathbb{R})$, respectivement un vecteur et une matrice de coefficients à valeurs réelles. Les relations suivantes

$$\begin{aligned} Y_i &= y_0 + h \sum_{j=1}^s a_{ij} f(Y_j), \quad i = 1, \dots, s, \\ y_1 &= y_0 + h \sum_{j=1}^s b_j f(Y_j), \end{aligned} \quad (1.3)$$

définissent un pas de la méthode de Runge-Kutta notée (A, b) . Les vecteurs Y_i sont les étapes internes de la méthode, tandis que y_1 désigne l'approximation après un pas.

Remarque 1.5 Lorsque A est triangulaire inférieure stricte, la méthode est dite explicite. Il est en effet aisé de voir que les étapes internes Y_1, \dots, Y_s peuvent être calculées successivement dans cet ordre en fonction de quantités déjà évaluées. En toute généralité (A quelconque), la méthode est dite implicite, et nécessite la résolution d'un système non-linéaire (pour un champ f non-linéaire) par une méthode itérative (méthode du point-fixe ou variante de la méthode de Newton). Les méthodes de Runge-Kutta sont généralement représentées par leur tableau de Butcher (dans lequel on prend $c = A\mathbf{1}$ où $\mathbf{1} = (1, \dots, 1)^T \in \mathbb{R}^s$) :

$$\begin{array}{c|ccc} c_1 & a_{11} & \dots & a_{1s} \\ \vdots & \vdots & & \vdots \\ c_s & a_{s1} & \dots & a_{ss} \\ \hline & b_1 & \dots & b_s \end{array} \quad (1.4)$$

Exemple 1.6 Les méthodes d'Euler, respectivement explicite et implicite, sont représentables par les tableaux respectifs suivants :

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array} \quad \text{et} \quad \begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}$$

La méthode dite “de Runge” est représentable par le tableau :

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1/2 & 1/2 & 0 \\ \hline & 0 & 1 \end{array}$$

Exemple 1.7 Supposons que f soit scalaire et linéaire, de la forme $f(y) = \lambda y$, $\lambda \in \mathbb{C}$. Alors le flot numérique ϕ_h^{RK} associé à une méthode de Runge-Kutta (A, b) est donné pour h suffisamment petit par

$$\forall y \in \mathbb{R}^d, \phi_h^{RK}(y) = R(\lambda h)y$$

où

$$R(z) = 1 + zb^T(I - zA)^{-1}\mathbf{1}$$

La fonction R est appelée fonction de stabilité de la méthode de Runge-Kutta : lorsque $|R(z)| \leq 1$, la méthode est dite stable.

1.2 Méthodes de composition

Soit ϕ_h le flot numérique d'une méthode fixée, tel que

$$\phi_h(y) = y + hf(y) + \mathcal{O}(h^2)$$

Soient alors $\gamma_1, \dots, \gamma_s$ des nombres réels. La composition de ϕ_h pour les pas $\gamma_1 h, \dots, \gamma_s h$

$$\psi_h = \phi_{\gamma_s h} \circ \dots \circ \phi_{\gamma_1 h}. \quad (1.5)$$

est appelée **méthode de composition**. L'intérêt de ce genre de composition est qu'il est possible de choisir les coefficients $\gamma_1, \dots, \gamma_s$ de manière à ce que ψ_h soit plus précise que ϕ_h tout en conservant les propriétés géométriques de ϕ_h . Ainsi, si le flot numérique ϕ_h est symplectique, il en sera de même de ψ_h , si ϕ_h conserve le volume, alors ψ_h également, ...

Exemple 1.8 Supposons que ϕ_h soit le flot numérique associé à la méthode suivante (d'ordre 2)

$$\phi_h(y) = y + hf(y) + \frac{1}{2}h^2 f'(y)f(y) = \varphi_h(y) + \mathcal{O}(h^3)$$

Alors, la composition

$$\psi_h = \phi_{\gamma_3 h} \circ \phi_{\gamma_2 h} \circ \phi_{\gamma_1 h}$$

avec $\gamma_1 + \gamma_2 + \gamma_3 = 1$ et $\gamma_1^3 + \gamma_2^3 + \gamma_3^3 = 0$, vérifie

$$\psi_h(y) = \varphi_h(y) + \mathcal{O}(h^4)$$

et est donc asymptotiquement plus précise que la méthode d'Euler (elle est d'ordre au moins 3)

1.3 Méthodes de splitting

Les méthodes de splitting reposent sur une décomposition additive du champ de vecteur f en 2 (ou plus dans le cas général qu'on ne traitera pas ici) fonctions que l'on sait intégrer, soit exactement, soit numériquement :

$$f(y) = f_1(y) + f_2(y).$$

Supposons par exemple que les flots exacts φ_t^1 et φ_t^2 associées aux deux systèmes

$$\dot{y} = f_1(y) \text{ and } \dot{y} = f_2(y),$$

avec même condition initiale puissent être calculés. Alors, partant d'une valeur approchée y_0 de la solution, on calcule

$$\phi_h(y_0) = (\varphi_h^2 \circ \varphi_h^1)(y_0) \text{ ou } \phi_h^*(y_0) = (\varphi_h^1 \circ \varphi_h^2)(y_0).$$

On rappelle qu'en raison des propriétés du flot exact (loi de groupe pour les flots des systèmes autonomes), on a $\varphi_{-t}^1 = (\varphi_t^1)^{-1}$ et de même pour φ_t^2 de sorte que ϕ_t^* est l'adjoint de ϕ_t , i.e.

$$\phi_t^* = (\phi_{-t})^{-1}.$$

Les deux valeurs $\phi_h^*(y_0)$ et $\phi_h(y_0)$ sont des approximations d'ordre 1 de la solution exacte, comme on peut le voir à partir des premiers termes du développement en série de Taylor :

$$\begin{aligned}\phi_h(y_0) &= \varphi_h^1(\varphi_h^2(y_0)) = \varphi_h^1(y_0 + hf_2(y_0) + \mathcal{O}(h^2)) \\ &= y_0 + hf_2(y_0) + \mathcal{O}(h^2) + hf_1(y_0 + hf_2(y_0) + \mathcal{O}(h^2)) + \mathcal{O}(h^2) \\ &= y_0 + h(f_1(y_0) + f_2(y_0)) + \mathcal{O}(h^2) \\ &= y_0 + hf(y_0) + \mathcal{O}(h^2)\end{aligned}$$

Les formules pour ϕ_h et ϕ_h^* sont connues sous le nom de formules de *Lie-Trotter*. Une autre formule célèbre est la formule de *Strang* qui correspond à la composition suivante

$$\phi_h^S = \varphi_{h/2}^1 \circ \varphi_h^2 \circ \varphi_{h/2}^1$$

On vérifie facilement qu'elle est d'ordre 2.

Remarque 1.9 *Il est clairement possible de généraliser cette technique dans deux directions :*

– *en décomposant f en $N \geq 3$ parties*

$$f(y) = \sum_{n=1}^N f_n(y)$$

puis en considérant les méthodes de la forme

$$\phi_h = \varphi_h^1 \circ \varphi_h^2 \circ \dots \circ \varphi_h^N$$

– *en composant les flots numériques ϕ_h and ϕ_h^* (par exemple) pour obtenir des méthodes d'ordres plus élevés.*

2 Transport et accumulation des erreurs locales : erreur globale

L'erreur *globale*, c'est-à-dire l'erreur d'approximation de la solution exacte à l'extrémité T de l'intervalle d'intégration, résulte du transport et de l'accumulation en bout d'intervalle des erreurs locales. Dans ce paragraphe, nous estimons cette erreur globale pour des flots numériques quelconques d'ordres au moins 1.

Soit donc, pour un flot numérique ϕ_h , la séquence des approximations numériques

$$y_{k+1} = \phi_h(y_k), \quad k = 0, \dots, N-1 \tag{2.6}$$

associée à une subdivision à pas $h = T/N$ constant $t_0 = 0, t_1, \dots, t_N = T$ de l'intervalle d'intégration $[0, T]$. L'objectif est d'estimer la quantité

$$E = \varphi_{Nh}(y_0) - y_N. \quad (2.7)$$

Une majoration de $\|E\|$ peut alors être obtenue en remarquant que E est la somme des erreurs E_k (voir figure 2) et en écrivant donc :

$$\begin{aligned} E &= \varphi_{Nh}(y_0) - y_N \\ &= E_1 + \sum_{k=1}^{N-2} E_{k+1} + E_N \\ &= (\varphi_{Nh}(y_0) - \varphi_{(N-1)h}(y_1)) + \sum_{k=1}^{N-2} (\varphi_{(N-k)h}(y_k) - \varphi_{(N-k-1)h}(y_{k+1})) + (\varphi_h(y_{N-1}) - y_N) \end{aligned} \quad (2.8)$$

où chaque terme $E_{k+1} = (\varphi_{(N-k)h}(y_k) - \varphi_{(N-k-1)h}(y_{k+1}))$ représente la contribution de l'erreur locale $e_{k+1} = \varphi_h(y_k) - y_{k+1}$ à l'erreur globale.

Supposons maintenant que le flot numérique considéré ϕ_h est d'ordre p , de sorte que sur tout compact $K \subset \mathbb{R}^d$, il existe une constante $C > 0$ pour laquelle

$$\|e_k\| \leq Ch^{p+1} \quad (2.9)$$

On rappelle maintenant une version simplifiée du lemme de Gronwall :

Lemme 2.1 Pour $t_i \in [0, T]$, soient $\varphi_{t-t_i}(y)$ et $\varphi_{t-t_i}(\hat{y})$ les solutions exactes de (0.1) avec conditions initiales respectives y et \hat{y} en $t = t_i$. Soit en outre L la constante de Lipschitz de f sur \mathbb{R}^d . Alors, pour tout t dans $[t_i, T]$, on a

$$\|\varphi_{t-t_i}(y) - \varphi_{t-t_i}(\hat{y})\| \leq \|y - \hat{y}\| e^{L(T-t_i)} \quad (2.10)$$

Théorème 2.2 Sout $K_\rho \subset \mathbb{R}^n$ un voisinage compact de la solution exacte $\varphi_t(y_0)$, $t \in [0, T]$ de (0.1) de la forme

$$K_\rho = \{y \in \mathbb{R}^d; \exists t \in [0, T], \|y - \varphi_t(y_0)\| \leq \rho\}, \quad \rho > 0$$

Alors pour h suffisamment petit (i.e. N suffisamment grand), l'erreur globale est majorée par

$$\|E\| \leq h^p \frac{C}{L} (e^{LT} - 1) \quad (2.11)$$

où $h = T/N$.

Preuve. Le flot numérique ϕ_h étant d'ordre p , pour un $h_0 > 0$ dans I donné, on peut majorer $\epsilon(h, y)$ sur $[0, h_0] \times K_\rho$ par une constante $C > 0$. Supposons que la séquence des y_i soit toute entière contenue dans K_ρ . Alors, d'après le lemme de Gronwall, on a

$$\|E\| \leq Ch^p \sum_{k=1}^N h e^{L(T-t_i)} \leq Ch^p \int_0^T e^{L(T-t)} dt \leq h^p \frac{C}{L} (e^{LT} - 1)$$

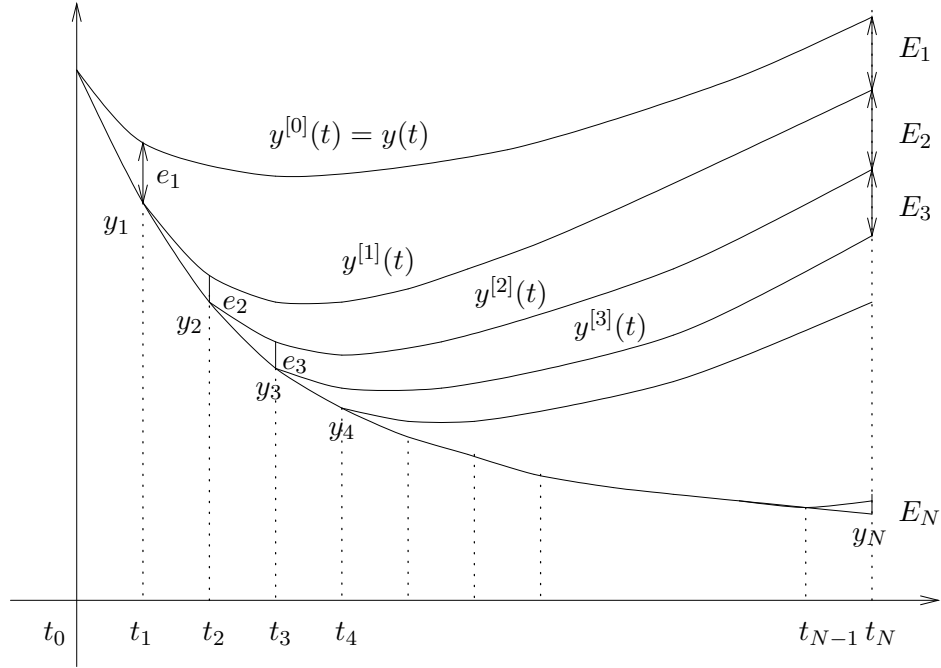


FIGURE 2 – Transport et accumulation des erreurs locales.

Maintenant, si h est tel que

$$h^p \frac{C}{L} (e^{L(T-t_0)} - 1) \leq \rho$$

alors on peut montrer par récurrence que la séquence des y_i ne sort pas du compact K_ρ et donc que l'estimation obtenue est valable. ■

Remarque 2.3 *Le théorème reste valable pour une subdivision de l'intervalle d'intégration à pas non constants. Il faut alors remplacer h par le maximum des $(t_{i+1} - t_i)$.*

3 Détermination des conditions d'ordre pour les méthodes de Runge-Kutta

Il est clair qu'une estimation du type (1.2) ne peut reposer que sur un développement en série de Taylor des deux flots φ_h et ϕ_h . On rappelle donc dans le paragraphe suivant, la forme que revêt le développement de Taylor d'une fonction à plusieurs variables.

3.1 Rappel de calcul différentiel

Série de Taylor pour d variables : pour $y \in \mathbb{R}^d$ fixé et $\Delta \in \mathbb{R}^d$, on rappelle que toute fonction f de \mathbb{R}^d dans \mathbb{R}^d indéfiniment différentiable est développable en série de Taylor

$$f_i(y + \Delta) = f_i(y) + \sum_{q=1}^{\infty} \frac{1}{q!} \sum_{j_1=1}^d \sum_{j_2=1}^d \cdots \sum_{j_q=1}^d \frac{\partial^q f_i(y)}{\partial y_{j_1} \partial y_{j_2} \cdots \partial y_{j_q}} \Delta_{j_1} \cdots \Delta_{j_q} \quad (3.12)$$

Sous cette forme, les termes de la série sont difficiles à manipuler, et on leur préfère la représentation compacte suivante, basée sur les formes multilinéaires :

$$\begin{aligned} f^{(q)}(y) : \mathbb{R}^d \times \mathbb{R}^d \times \cdots \times \mathbb{R}^d &\rightarrow \mathbb{R}^d \\ (\Delta^{(1)}, \dots, \Delta^{(q)}) &\mapsto f^{(q)}(y) \left(\Delta^{(1)}, \dots, \Delta^{(q)} \right) \end{aligned}$$

où les composantes de ce vecteur sont définies par

$$\forall i = 1, \dots, d, \quad \left(f^{(q)}(y) \left(\Delta^{(1)}, \dots, \Delta^{(q)} \right) \right)_i = \sum_{j_1=1}^d \sum_{j_2=1}^d \cdots \sum_{j_q=1}^d \frac{\partial^q f_i(y)}{\partial y_{j_1} \partial y_{j_2} \cdots \partial y_{j_q}} \Delta_{j_1}^{(1)} \cdots \Delta_{j_q}^{(q)}$$

Il est important de noter les deux propriétés suivantes :

- l'application $f^{(q)}(y)$ est linéaire en chacun de ses arguments ;
- si σ est une permutation de $\{1, \dots, q\}$, alors

$$f^{(q)}(y) \left(\Delta^{(1)}, \dots, \Delta^{(q)} \right) = f^{(q)}(y) \left(\Delta^{(\sigma(1))}, \dots, \Delta^{(\sigma(q))} \right)$$

La série de Taylor de f peut donc aussi s'écrire

$$f(y + \Delta) = f(y) + \sum_{q=1}^{\infty} \frac{1}{q!} f^{(q)}(y) \left(\underbrace{\Delta, \dots, \Delta}_q \right)$$

Généralement, cette série ne converge pas. C'est la raison pour laquelle, on lui préfère la version tronquée avec reste.

Formule de Taylor avec reste intégral : pour $y \in \mathbb{R}^d$ fixé et $\Delta \in \mathbb{R}^d$, on rappelle que toute fonction f de \mathbb{R}^d dans \mathbb{R}^d indéfiniment différentiable est développable en série de Taylor avec reste

$$f(y + \Delta) = f(y) + \sum_{q=1}^N \frac{1}{q!} f^{(q)}(y) \left(\Delta, \dots, \Delta \right) + R_{N+1}(y, \Delta)$$

où

$$R_{N+1}(y, \Delta) = \int_0^1 \frac{(1-t)^N}{N!} f^{(N+1)}(y + t\Delta) \left(\Delta, \dots, \Delta \right) dt$$

3.2 Les premier termes

Avant de poursuivre de manière systématique, considérons par exemple -et en raison de sa simplicité- le flot numérique $\phi_h(y_0)$ associé à la méthode d'Euler implicite :

$$\phi_h(y_0) = y_0 + hf(\phi_h(y_0)).$$

En remplaçant dans le membre de droite $\phi_h(y_0)$ par $y_0 + \mathcal{O}(h)$ et en développant, puis en réitérant le processus avec l'approximation ainsi obtenue, on obtient successivement

$$\begin{aligned}\phi_h(y_0) &= y_0 + h \underbrace{f(y_0)}_{= \frac{d}{dt}\varphi_t(y_0)|_{t=0}} + \mathcal{O}(h^2), \\ \phi_h(y_0) &= y_0 + h \underbrace{f(y_0)}_{= \frac{d}{dt}\varphi_t(y_0)|_{t=0}} + h^2 \underbrace{f'(y_0)f(y_0)}_{= \frac{d^2}{dt^2}\varphi_t(y_0)|_{t=0}} + \mathcal{O}(h^3).\end{aligned}$$

Le développement de Taylor de $\phi_h(y_0)$ à l'ordre 2 ne fait donc apparaître que des dérivées de la solution exacte $\varphi_t(y_0)$ au point $t = 0$. Cependant, une itération supplémentaire permet d'obtenir le terme d'ordre 3

$$\begin{aligned}& h^3 \underbrace{\left(f'(y_0)f'(y_0)f(y_0) + \frac{1}{2}f''(y_0)(f(y_0), f(y_0)) \right)}_{\neq \frac{d^3}{dt^3}\varphi_t(y_0)|_{t=0} = f'(y_0)f'(y_0)f(y_0) + f''(y_0)(f(y_0), f(y_0))},\end{aligned}$$


qui lui, ne coïncide pas avec $\frac{d^3}{dt^3}\varphi_t(y_0)|_{t=0}$. Le développement de Taylor à tout ordre de $\phi_h(y_0)$ nécessite ainsi de distinguer chacune des dérivées de f . Il est bien sûr légitime d'omettre l'argument y_0 (ce que nous ferons désormais) dans $f(y_0)$, $f'(y_0)$, $f''(y_0)$, ..., ce qui allège l'écriture, mais ne suffit pas à rendre lisible l'ensemble de la série. Une complication supplémentaire tient à l'occurrence de coefficients devant chacun des termes (par exemple le coefficient $1/2$ devant $f''(f, f)$) qu'il est indispensable de pouvoir calculer. Pour toutes ces raisons, il a été très tôt proposé (Cayley 1857) de représenter les *différentielles élémentaires* de f par des arbres racinés, c'est-à-dire des graphes orientés (par convention vers le haut, la racine étant "au pied de l'arbre") et sans cycle. Dans les deux paragraphes suivants, nous construisons les développements "en arbre" des séries de Taylor de la solution exacte $\varphi_h(y_0)$, puis de la solution numérique $\phi_h(y_0)$ obtenue par une méthode de Runge-Kutta.



3.3 Dérivées de la solution exacte

La solution numérique ne pouvant être développée qu'en fonction des dérivées de f , il convient d'exprimer $\frac{d}{dt}\varphi_t(y_0)|_{t=0}$, $\frac{d^2}{dt^2}\varphi_t(y_0)|_{t=0}$, ..., suivant la même base. Pour les quatre premières dérivées de $\varphi_h(y_0)$, on obtient :

$$\frac{d}{dt}\varphi_t(y_0)\Big|_{t=0} = f$$

•

$$\left. \frac{d^2}{dt^2} \varphi_t(y_0) \right|_{t=0} = f' f$$


$$\left. \frac{d^3}{dt^3} \varphi_t(y_0) \right|_{t=0} = f''(f, f) + f' f' f$$



$$\begin{aligned} \left. \frac{d^4}{dt^4} \varphi_t(y_0) \right|_{t=0} &= f'''(f, f, f) + f''(f, f) + f''(f, f' f) + f''(f' f, f) + f' f''(f, f) + f' f' f' f \\ &= f'''(f, f, f) + 3 f''(f, f' f) + f' f''(f, f) + f' f' f' f \end{aligned}$$



A chaque élément différentiel $f, f' f, \dots$, a été associée la représentation graphique d'un arbre, dont l'interprétation peut maintenant être décrite en 3 points :

1. la racine de l'arbre, comme tous les noeuds, correspond à l'élément différentiel obtenu en dérivant f un certain nombre de fois (par rapport à y) ;
2. si un noeud possède k branches, alors la fonction f est dérivée k fois. C'est alors une fonction multilinéaire à k arguments ;
3. le i - ème argument de cette fonction est l'élément différentiel correspondant à la i - ème branche issue de ce noeud.





L'équivalence entre arbres et éléments différentiels peut maintenant être décrite formellement :

Définition 3.1 *L'ensemble \mathcal{T} des arbres est défini récursivement par :*

1. $\bullet \in \mathcal{T}$.
2. $\tau = [\tau_1, \dots, \tau_k] \in \mathcal{T}$ ssi $(\tau_1, \dots, \tau_k) \in \mathcal{T}^k$.

La notation $[\tau_1, \dots, \tau_k]$ désigne l'arbre obtenu en connectant les k branches τ_1, \dots, τ_k à une nouvelle racine commune. L'ordre des branches n'importe pas, pas plus que l'ordre des arguments de $f''(f' f, f)$, ou $f'''(f, f' f, f)$. Par exemple, on notera

$$[\bullet, \bullet, \bullet] = \text{tree with 3 branches} \quad [[\bullet], \bullet] = \text{tree with 2 branches, right one has 1 sub-branch} \quad [[\bullet, \bullet]] = \text{tree with 2 branches, left one has 1 sub-branch} \quad [[[\bullet]]] = \text{tree with 1 branch, which has 1 branch, which has 1 branch}$$

τ				
$F(\tau)$	f	$f'f$	$f''(f, f)$	$f'f'f$
$F_i(\tau)$	f_i	$\sum_{j=1}^d \frac{\partial f_i}{\partial y_j} f_j$	$\sum_{j=1, k=1}^d \frac{\partial^2 f_i}{\partial y_j \partial y_k} f_j f_k$	$\sum_{j=1, k=1}^d \frac{\partial f_i}{\partial y_j} \frac{\partial f_j}{\partial y_k} f_k$

Définition 3.2 A tout arbre τ de \mathcal{T} , on associe l'élément différentiel $F(\tau)$ défini récursivement par :

1. $F(\bullet)(y) = f(y)$,
2. $F([\tau_1, \dots, \tau_k])(y) = \frac{\partial^k f}{\partial y^k}(y) \left(F(\tau_1)(y), \dots, F(\tau_k)(y) \right)$.

Conformément à l'usage en vigueur dans ce paragraphe, nous omettons dans $F(\tau)(y)$ l'argument y lorsqu'aucune confusion n'est possible, et notons $F(\tau)$ en lieu et place de $F(\tau)(y)$. Les premiers termes du développement de Taylor de $\varphi_h(y_0)$ peuvent alors s'écrire sous la forme

$$\begin{aligned}
\varphi_h(y_0) &= y_0 + hf + \frac{h^2}{2!} (f'f) + \frac{h^3}{3!} (f''(f, f) + f'f'f) \\
&\quad + \frac{h^4}{4!} (f'''(f, f, f) + 3f''(f'f, f) + f'f''(f, f) + f'f'f'f) + \dots \\
&= y_0 + hF(\bullet) + \frac{h^2}{2!} F(\bullet\bullet) + \frac{h^3}{3!} (F(\bullet\vee\bullet) + F(\bullet\bullet\bullet)) \\
&\quad + \frac{h^4}{4!} (F(\bullet\vee\vee\bullet) + 3F(\bullet\vee\bullet\bullet) + F(\bullet\vee\bullet\bullet) + F(\bullet\bullet\bullet\bullet)) + \dots
\end{aligned}$$

Si l'ordre $|\tau|$ d'un arbre τ est défini comme son nombre de noeuds, on peut vérifier que les arbres d'ordre q apparaissent tous dans $\frac{d^q}{dt^q} \varphi_t(y_0) \big|_{t=0}$. Il reste finalement à déterminer le coefficient $\alpha(\tau)$ qui apparaît devant chaque arbre $\tau \in \mathcal{T}$. La tâche est facilitée par l'introduction du coefficient de normalisation σ suivant :

Définition 3.3 On définit la symétrie $\sigma(\tau)$ d'un arbre $\tau \in \mathcal{T}$ récursivement de la manière suivante :

1. $\sigma(\bullet) = 1$.
2. Si $(\tau_1, \dots, \tau_k) \in \mathcal{T}^k$ sont k arbres **distincts**, si r_1, r_2, \dots, r_k sont k entiers non nuls, et si

$$\tau = \underbrace{[\tau_1, \dots, \tau_1]}_{r_1}, \dots, \underbrace{[\tau_k, \dots, \tau_k]}_{r_k},$$

alors

$$\sigma(\tau) = \prod_{i=1}^k r_i! \sigma(\tau_i)^{r_i}.$$


















τ																	
$ \tau $	1	2	3	3	4	4	4	4	5	5	5	5	5	5	5	5	5
$\sigma(\tau)$	1	1	2	1	6	1	2	1	24	2	2	2	1	6	1	2	1

FIGURE 3 – Arbres d’ordres inférieurs à 5 et coefficients de symétrie associés.

Les arbres d’ordres 1 à 5 sont énumérés dans le tableau 3 ainsi que les valeurs de σ associées.

On alors le résultat suivant :

Théorème 3.4 *Le flot exact $\varphi_h(y_0)$ est développable en série formelle et on a :*

$$\varphi_t(y_0) = y_0 + \sum_{\tau \in \mathcal{T}} \frac{t^{|\tau|}}{\sigma(\tau)\gamma(\tau)} F(\tau)(y_0), \quad (3.13)$$

où le coefficient $\gamma(\tau)$ est défini récursivement de la manière suivante :

1. $\gamma(\bullet) = 1$,
2. $\gamma([\tau_1, \dots, \tau_k]) = |\tau| \prod_{j=1}^k \gamma(\tau_j)$.

Preuve. Il suffit de vérifier a posteriori que $\varphi_t(y_0)$ donné par le développement (3.13) est solution de l’équation intégrale satisfaite par $\varphi_t(y_0)$, à savoir

$$\varphi_t(y_0) = y_0 + \int_0^t f(\varphi_s(y_0)) ds$$

A cet effet, on calcule :

$$f(\varphi_s(y_0)) = f(y_0 + \Delta) = f(y_0) + \sum_{k=1}^{\infty} \frac{1}{k!} f^{(k)}\left(\underbrace{\Delta, \dots, \Delta}_{k \text{ fois}}\right) \quad (3.14)$$

où

$$\Delta = \sum_{\tau \in \mathcal{T}} \frac{h^{|\tau|}}{\sigma(\tau)\gamma(\tau)} F(\tau)(y_0).$$

En développant cette somme, on obtient donc

$$f(\varphi_s(y_0)) = f(y_0) + \sum_{k=1}^{\infty} \sum_{\tilde{\tau}_1, \dots, \tilde{\tau}_k} \frac{s^{|\tilde{\tau}_1|} \dots s^{|\tilde{\tau}_k|}}{k! \sigma(\tilde{\tau}_1) \dots \sigma(\tilde{\tau}_k) \gamma(\tilde{\tau}_1) \dots \gamma(\tilde{\tau}_k)} f^{(k)}\left(F(\tilde{\tau}_1), \dots, F(\tilde{\tau}_k)\right)$$

On observe alors que, si $\tau = [\tilde{\tau}_1, \dots, \tilde{\tau}_k]$, on a par définition de $F(\tau)$, $|\tau|$ et $\gamma(\tau)$, les relations suivantes

$$\begin{aligned} f^{(k)}\left(F(\tilde{\tau}_1), \dots, F(\tilde{\tau}_k)\right) &= F(\tau) \\ t^{|\tilde{\tau}_1|} \dots t^{|\tilde{\tau}_k|} &= t^{|\tau|-1} \\ \text{et } \gamma(\tilde{\tau}_1) \dots \gamma(\tilde{\tau}_k) &= \frac{\gamma(\tau)}{|\tau|} \end{aligned}$$

et en réécrivant $\tau = \underbrace{[\tau_1, \dots, \tau_1]_{r_1}}_{r_1}, \dots, \underbrace{[\tau_m, \dots, \tau_m]_{r_m}}_{r_m}$ où les τ_i sont désormais supposés distincts deux à deux et $\sum_{j=1}^m r_j = k$, il vient

$$\sigma(\tilde{\tau}_1) \dots \sigma(\tilde{\tau}_k) = \sigma(\tau_1)^{r_1} \dots \sigma(\tau_m)^{r_m} = \frac{\sigma(\tau)}{r_1! \dots r_m!}.$$

On peut donc remplacer la somme sur les k -uplets d'arbres $\tilde{\tau}_1, \dots, \tilde{\tau}_k$ par une somme sur les arbres $\tau = [\tilde{\tau}_1, \dots, \tilde{\tau}_k]$ à k branches : il faut cependant prendre garde au fait que les k -uplets $\tilde{\tau}_1, \dots, \tilde{\tau}_k$ sont ordonnés, alors que les branches de τ ne le sont pas, et donc diviser par $\frac{r_1! \dots r_m!}{k!}$. On a finalement

$$f(\varphi_s(y_0)) = \sum_{\tau \in \mathcal{T}} \frac{s^{|\tau|-1}}{\sigma(\tau)\gamma(\tau)} |\tau| F(\tau) \quad (3.15)$$

et donc

$$y_0 + \int_0^t f(\varphi_s(y_0)) \, ds = y_0 + \sum_{\tau \in \mathcal{T}} \frac{t^{|\tau|}}{\sigma(\tau)\gamma(\tau)} F(\tau)$$

■

3.4 B-séries et développement de la solution numérique

Bien qu'il soit possible de construire le développement en série de Taylor de $\phi_h(y_0)$ directement, la tâche est rendue plus aisée par l'introduction du concept de B-séries, dont nous verrons plus loin qu'il permet en outre de résoudre quelques questions profondes liées à la préservation des invariants.

Définition 3.5 Soit α une application de l'ensemble des arbres \mathcal{T} vers \mathbb{R} . On appelle *B-série* la série formelle suivante :

$$B(\alpha, y_0) = y_0 + \sum_{\tau \in \mathcal{T}} \frac{h^{|\tau|}}{\sigma(\tau)} \alpha(\tau) F(\tau)(y_0) \quad (3.16)$$

Remarque 3.6 Dans le cas général, une B-série n'est pas convergente : c'est déjà le cas lorsque le champ de vecteur f est linéaire et scalaire pour lequel la série est entière. Il faut donc à ce stade considérer la série comme formelle.

Lemme 3.7 Soit $B(\alpha, y_0)$ une B-série. Alors $y_0 + hf\left(B(\alpha, y_0)\right)$ est encore une B-série et on a la relation suivante

$$y_0 + hf\left(B(\alpha, y_0)\right) = B(\beta, y_0)$$

où β est défini récursivement par

1. $\beta(\bullet) = 1$;
2. si $\tau = [\tau_1, \dots, \tau_k]$ alors $\beta(\tau) = \alpha(\tau_1) \dots \alpha(\tau_k)$.

Preuve. La preuve est quasiment identique à celle du théorème 3.4. En prenant

$$\Delta = B(\alpha, y_0) - y_0$$

dans la formule (3.14), on obtient l'équivalent de la formule (3.15) où l'on a essentiellement remplacé $1/\gamma$ par α :

$$f\left(B(\alpha, y_0)\right) = f(y_0) + \sum_{\tau=[\tau_1, \dots, \tau_k] \in \mathcal{T}} \frac{h^{|\tau|-1}}{\sigma(\tau)} \alpha(\tau_1) \dots \alpha(\tau_k) F(\tau)$$

de sorte que

$$y_0 + hf\left(B(\alpha, y_0)\right) = y_0 + \sum_{\tau \in \mathcal{T}} \frac{h^{|\tau|}}{\sigma(\tau)} \beta(\tau) F(\tau).$$

■

Afin de déterminer le développement en série du flot numérique $\phi_h(y_0)$ associé à une méthode de Runge-Kutta (A, b) , nous supposons que chaque étape interne Y_i , pour $i = 1, \dots, s$, peut être représentée par une B-série de coefficients α_i , pour $i = 1, \dots, s$. Pour chaque arbre τ , on note $\alpha(\tau)$ le vecteur de \mathbb{R}^s de composantes $\alpha_i(\tau)$, $i = 1, \dots, s$. Les équations liant les Y_i conduisent alors aux relations :

$$Y_i = y_0 + \sum_{j=1}^s a_{i,j} hf\left(B(\alpha_j, y_0)\right) = y_0 + \sum_{j=1}^s a_{i,j} (B(\beta_j, y_0) - y_0) = B\left(\sum_{j=1}^s a_{i,j} \beta_j, y_0\right).$$

En notant \diamond le produit (commutatif et associatif) de vecteurs composante par composante défini comme

$$\forall (u, v) \in \mathbb{R}^s \times \mathbb{R}^s, (u \diamond v) = (u_i v_i)_{i=1, \dots, s}$$

on obtient ainsi la formule récursive suivante :

$$\begin{aligned} \alpha(\bullet) &= A\mathbf{1}, \\ \alpha([\tau_1, \dots, \tau_k]) &= A\left(\alpha(\tau_1) \diamond \alpha(\tau_2) \diamond \dots \diamond \alpha(\tau_k)\right). \end{aligned}$$

Il est alors aisé de prouver le théorème suivant :

Théorème 3.8 Soit une méthode de Runge-Kutta de coefficients (A, b) où $A \in \mathcal{M}_s(\mathbb{R})$ et $b \in \mathbb{R}^s$. Le flot numérique Φ_h associé à cette méthode est développable en B-série et on a

$$\phi_h(y_0) = y_0 + \sum_{\tau \in \mathcal{T}} \frac{h^{|\tau|}}{\sigma(\tau)} \left(b^T \Phi(\tau) \right) F(\tau)(y_0)$$

où Φ est la fonction de \mathcal{T} à valeurs dans \mathbb{R}^s définie récursivement par

1. $\Phi(\bullet) = \mathbf{1}$;
2. si $\tau = [\tau_1, \dots, \tau_k]$ alors $\Phi(\tau) = A \left(\Phi(\tau_1) \diamond \Phi(\tau_2) \diamond \dots \diamond \Phi(\tau_k) \right)$.

Théorème 3.9 Soit une méthode de Runge-Kutta de coefficients (A, b) où $A \in \mathcal{M}_s(\mathbb{R})$ et $b \in \mathbb{R}^s$. Le flot numérique associé ϕ_h est d'ordre local $p + 1$ (c'est-à-dire d'ordre global p) si et seulement si

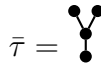
$$\forall \tau \in \mathcal{T}, |\tau| \leq p, b^T \Phi(\tau) = \frac{1}{\gamma(\tau)}. \quad (3.17)$$

Preuve. Le caractère suffisant des conditions est clair : si deux B-séries coïncident pour tous les arbres d'ordres plus petits ou égaux à p , alors leur différence est bornée par un terme de la forme $h^{p+1} R(t)$ où $0 \leq t \leq h$ et où $R(t)$ peut être obtenu en considérant les développements en série de Taylor avec reste intégral des solutions, exacte et numérique.

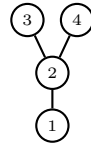
Le caractère nécessaire des conditions repose sur la possibilité de choisir f arbitrairement, de telle sorte que la famille $(F(\tau))_{\tau \in \mathcal{T}}$ constitue une base de l'espace vectoriel des séries formelles de la forme $B(\alpha, \cdot) - Id$. Etant donné un arbre $\bar{\tau}$ d'ordre q quelconque, nous allons donc construire un champ f tel que

$$F(\bar{\tau})(y_0) \neq 0 \text{ et } F(\tau)(y_0) = 0 \text{ pour tout } \tau \neq \bar{\tau} \text{ tel que } |\tau| = q.$$

L'idée consiste à attribuer **une et une seule fois** q indices $1, \dots, q$ aux q noeuds de l'arbre $\bar{\tau}$ d'une manière arbitrairement fixée. Pour l'arbre



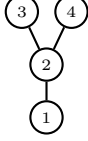
on aura par exemple



On construit alors le champ f de la manière suivante : pour chaque noeud de $\bar{\tau}$, soit i son indice et j_1, \dots, j_k les k indices des racines de ses branches. On pose

$$f_i(y) = y_{j_1} \cdots y_{j_k} \text{ si } k \geq 1 \text{ et } f_i(y) = 1 \text{ si } k = 0.$$

En parcourant les q noeuds de $\bar{\tau}$, on définit ainsi les q composantes de $f \in \mathcal{C}^\infty(\mathbb{R}^q; \mathbb{R}^q)$. Pour l'arbre

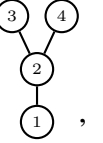


indiqué par exemple, on obtient ainsi :

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} = \begin{pmatrix} y_2 \\ y_3 y_4 \\ 1 \\ 1 \end{pmatrix}$$

Il est alors facile de vérifier que

$$F(\bar{\tau}) = \begin{pmatrix} \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix}$$



et que $F(\tau) = 0$ pour tout arbre $\tau \neq \bar{\tau}$ d'ordre supérieur ou égal à q . Par exemple, pour $\bar{\tau} =$ la première composante de $F(\bar{\tau})(y)$ s'écrit

$$\sum_{j,k,l} \frac{\partial f_1}{\partial y_j} \frac{\partial^2 f_j}{\partial y_k \partial y_l} f_k f_l = \frac{\partial f_1}{\partial y_2} \frac{\partial^2 f_2}{\partial y_3 \partial y_4} f_3 f_4 = 1$$

car tous les autres termes de la somme sont nuls. De même, les autres composantes de $F(\bar{\tau})(y)$ sont nulles. ■

Exemple 3.10 *A titre d'illustration, on cherche à construire une méthode de Runge-Kutta d'ordre 4. Soit donc (A, b) , une méthode à s étapes. On rappelle que*

$$A \in \mathcal{M}_s(\mathbb{R}), \quad b \in \mathbb{R}^s$$

et on note $c = A\mathbf{1}$. Les conditions d'ordre sont obtenues en construisant les arbres d'ordre inférieur ou égal à 4 (voir Tableau 1). Il est clair que $s = 1$ ne permet pas de les satisfaire toutes. Pour $s = 2$, il est aisé de vérifier que $c_1 = c_2$ ne convient pas. On suppose donc que c_1 et c_2 sont distincts de sorte que $(\mathbf{1}, c)$ forme une base de \mathbb{R}^2 . Il existe donc δ_1 et δ_2 , tels que

$$c^{\odot 2} = \delta_1 \mathbf{1} + \delta_2 c.$$

Or, $b^T[\mathbf{1}, c, c^2, c^3] = (1, 1/2, 1/3, 1/4)$, donc

$$\begin{cases} \delta_1 + \frac{1}{2}\delta_2 = \frac{1}{3} \\ \delta_1 \delta_2 + \frac{1}{2}(\delta_1 + \delta_2^2) = \frac{1}{4} \end{cases},$$

et on obtient $\delta_1 = -1/6$ et $\delta_2 = 1$. c_1 et c_2 sont donc les racines de $x^2 - x + 1/6 = 0$. On prend par exemple $c_1 = \frac{1}{2} - \frac{\sqrt{3}}{6}$ et $c_2 = \frac{1}{2} + \frac{\sqrt{3}}{6}$. Le système $b^T[e, c] = (1, 1/2)$ donne alors $b_1 = b_2 = 1/2$. De même, il existe μ_1 et μ_2 , tels que

$$Ac = \mu_1 e + \mu_2 c.$$

Or, $b^T[Ac, A^2c] = (1/6, 1/24)$, donc

$$\begin{cases} \mu_1 + \frac{1}{2}\mu_2 = \frac{1}{6} \\ \mu_1\mu_2 + \frac{1}{2}(\mu_1 + \mu_2^2) = \frac{1}{24} \end{cases},$$

et on obtient $\mu_1 = -1/12$ et $\mu_2 = 1/2$. Ce qui signifie en particulier que

$$Ac = \frac{1}{2}c^{\diamond 2}.$$

Réciproquement, si $Ac = \frac{1}{2}c^{\diamond 2}$, alors on peut vérifier que

$$\begin{aligned} b^T Ac &= \frac{1}{2}b^T c^{\diamond 2} = \frac{1}{6}, \\ b^T Ac^2 &= b^T A(c - \frac{1}{2}\mathbf{1}) = b^T Ac - \frac{1}{6}b^T A\mathbf{1} = \frac{1}{12}, \\ b^T A^2c &= \frac{1}{2}b^T Ac^{\diamond 2} = \frac{1}{24}, \\ b^T(c \diamond Ac) &= \frac{1}{2}b^T(c \diamond c^2) = \frac{1}{2}b^T c^{\diamond 3} = \frac{1}{6}. \end{aligned}$$

Il n'existe donc qu'une seule méthode d'ordre 4 avec $s = 2$. C'est la méthode de Gauss :

$$\begin{array}{c|cc} \frac{1}{2} - \frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{2} + \frac{\sqrt{3}}{6} & \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Exemple 3.11 Cherchons désormais à construire une méthode explicite d'ordre 3. Il est aisé de constater que $s = 2$ est insuffisant. On prend donc $s = 3$. En supposant comme précédemment que $A\mathbf{1} = c$, A et c sont de la forme

$$A = \begin{bmatrix} 0 & 0 & 0 \\ c_2 & 0 & 0 \\ a_{31} & c_3 - a_{31} & 0 \end{bmatrix} \text{ et } c = \begin{bmatrix} 0 \\ c_2 \\ c_3 \end{bmatrix}.$$

Des conditions $b^T e = 1$, $b^T Ae = 1/2$ et $b^T (Ae)^2 = 1/3$ on tire b_1 , b_2 et b_3 . a_{31} est alors obtenu à partir de l'équation restante $b^T A^2\mathbf{1} = 1/6$. On obtient alors la méthode suivante

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ c_2 & c_2 & 0 & 0 \\ c_3 & \frac{c_3(3c_2^2 - 3c_2 + c_3)}{c_2(3c_2 - 2)} & \frac{c_3(c_2 - c_3)}{c_2(3c_2 - 2)} & 0 \\ \hline & \frac{1}{6} \frac{-3c_2 + 6c_2c_3 + 2 - 3c_3}{c_2c_3} & \frac{-1}{6} \frac{3c_3 - 2}{c_2(c_2 - c_3)} & \frac{1}{6} \frac{3c_2 - 2}{c_2(c_2 - c_3)} \end{array}$$

On peut par exemple éliminer les paramètres libres restants en imposant les deux conditions d'ordre 4, $b^T(Ae)^3 = 1/4$ et $b^T A(Ae)^2 = 1/12$, ce qui donne finalement la méthode

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 \\ 1 & -1 & 2 & 0 \\ \hline & 1/6 & 2/3 & 1/6 \end{array}.$$

Ordre	τ	$\gamma(\tau)$	$\Phi(\tau)$	Condition
1		1	e	$b^T e = 1$
2		2	$c = Ae$	$b^T c = \frac{1}{2}$
3		3	$c^{\diamond 2} = c \diamond c$	$b^T c^{\diamond 2} = \frac{1}{3}$
		6	Ac	$b^T Ac = \frac{1}{6}$
4		4	$c^{\diamond 3}$	$b^T c^{\diamond 2} = \frac{1}{3}$
		8	$c \diamond Ac$	$b^T (c \diamond Ac) = \frac{1}{8}$
		12	$Ac^{\diamond 2}$	$b^T Ac^{\diamond 2} = \frac{1}{12}$
		24	$A^2 c$	$b^T A^2 c = \frac{1}{24}$
5		5	$c^{\diamond 4}$	$b^T c^{\diamond 4} = \frac{1}{5}$
		10	$c^{\diamond 2} \diamond (Ac)$	$b^T (c^{\diamond 2} \diamond (Ac)) = \frac{1}{10}$
		20	$(Ac) \diamond (Ac)$	$b^T ((Ac) \diamond (Ac)) = \frac{1}{20}$
		15	$c \diamond (Ac^{\diamond 2})$	$b^T c \diamond (Ac^{\diamond 2}) = \frac{1}{15}$
		30	$c \diamond (A^2 c)$	$b^T c \diamond (A^2 c) = \frac{1}{30}$
		20	$A(c^{\diamond 3})$	$b^T A(c^{\diamond 3}) = \frac{1}{20}$
		40	$A(c \diamond (Ac))$	$b^T A(c \diamond (Ac)) = \frac{1}{40}$
		60	$A^2 c^{\diamond 2}$	$b^T A^2 c^{\diamond 2} = \frac{1}{60}$
		120	$A^3 c$	$b^T A^3 c = \frac{1}{120}$

TABLE 1 – Les 17 conditions d'ordre 5.