Metode Runge-Kutta

Radu T. Trîmbiţaş

18 ianuarie 2007

1 Probleme scalare, pas constant

Dorim să aproximăm soluţia problemei Cauchy

$$\begin{aligned} &y'(t) = f(t, y), a \leq t \leq b, \\ &y(a) = \alpha. \end{aligned}$$

pe o grilă uniformă de $(N + 1)$-puncte din $[a, b]$.

Dându-se un punct generic $x \in [a, b]$, $y \in \mathbb{R}^d$, definim un pas al metodei cu un pas prin

$$y_{next} = y + h\Phi(x, y; h), \quad h > 0. \tag{1}$$

La metodele Runge-Kutta se caută Φ de forma:

$$\begin{aligned} &\Phi(x, y; h) = \sum_{s=1}^{r} \alpha_s K_s \\ &K_1(x, y) = f(x, y) \\ &K_s(x, y) = f\left(x + \mu_s h, y + h\sum_{j=1}^{s-1} \lambda_{sj} K_j\right), \qquad s = 2, 3, \dots, r \end{aligned} \tag{2}$$

Este natural să impunem în (2) condiţiile

$$\mu_s = \sum_{j=1}^{s-1} \lambda_{sj}, \quad s = 2, 3, \dots, r, \quad \sum_{s=1}^{r} \alpha_s = 1, \tag{3}$$

unde primul set de condiţii este echivalent cu

$$K_s(x, y; h) = u'(x + \mu_s h) + O(h^2), \quad s \geq 2,$$

iar a doua este condiţia de consistenţă (adică $\Phi(x, y; h) = f(x, y)$).

Formula Runge-Kutta clasică de ordin $p = 4$ este:

$$\begin{aligned}
&\Phi(x, y; h) = \tfrac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)\\
&K_1(x, y; h) = f(x, y)\\
&K_2(x, y; h) = f\left(x + \tfrac{1}{2}h, y + \tfrac{1}{2}hK_1\right)\\
&K_3(x, y; h) = f\left(x + \tfrac{1}{2}h, y + \tfrac{1}{2}hK_2\right)\\
&K_4(x, y; h) = f(x + h, y + hK_3)
\end{aligned} \tag{4}$$

Metoda Runge-Kutta clasică de ordinul 4 pentru o grilă de $N + 1$ puncte echidistante este dată de algoritmul 1.

Algorithm 1 Metoda Runge-Kutta de ordinul 4

Intrare: Funcţia f, capetele a, b ale intervalului; întregul N; valoarea iniţială α.

Ieşire: $N + 1$ abscise t şi aproximantele w ale lui valorilor lui y în t.

$h := (b - a)/N$;
$t_0 := a$;
$w_0 := \alpha$;
for $i := 0$ **to** $N - 1$ **do**
  $K_1 := hf(t_i, w_i)$;
  $K_2 := hf(t_i + h/2, w_i + K_1/2)$;
  $K_3 := hf(t_i + h/2, w_i + K_2/2)$;
  $K_4 := hf(t_i + h, w_i + K_3)$;
  $w_{i+1} := w_i + \frac{1}{6}(K_1 + 2 * K_2 + 2 * K_3 + K_4)$;
  $t_{i+1} := t_i + h$;
end for

Exemplu. Utilizând metoda Runge-Kutta de ordinul 4 pentru a aproxima soluţia problemei Cauchy

$$\begin{aligned}
&y' = -y + t + 1, \quad t \in [0, 1]\\
&y(0) = 1,
\end{aligned}$$

cu $h = 0.1$, $N = 10$ şi $t_i = 0.1i$ se obţin rezultatele din tabelul de mai jos

t_i	Aproximante	Valori exacte	Eroarea
0.0	1	1	0
0.1	1.00483750000	1.00483741804	8.19640e-008
0.2	1.01873090141	1.01873075308	1.48328e-007
0.3	1.04081842200	1.04081822068	2.01319e-007
0.4	1.07032028892	1.07032004604	2.42882e-007
0.5	1.10653093442	1.10653065971	2.74711e-007
0.6	1.14881193438	1.14881163609	2.98282e-007
0.7	1.19658561867	1.19658530379	3.14880e-007
0.8	1.24932928973	1.24932896412	3.25617e-007
0.9	1.30656999120	1.30656965974	3.31459e-007
1.0	1.36787977441	1.36787944117	3.33241e-007

Se obişnuieşte să se asocieze unei metode Runge-Kutta cu r stadii (2) tabloul

$$\begin{array}{c|cccc} \mu_1 & \lambda_{11} & \lambda_{12} & \dots & \lambda_{1r} \\ \mu_2 & \lambda_{21} & \lambda_{22} & \dots & \lambda_{2r} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ \mu_r & \lambda_{r1} & \lambda_{r2} & \dots & \lambda_{rr} \\ \hline & \alpha_1 & \alpha_2 & \dots & \alpha_r \end{array} \qquad \left(\text{în formă matricială } \begin{array}{c|c} \mu & \Lambda \\ \hline & \alpha^T \end{array}\right)$$

numit *tabelă Butcher*. Pentru metoda Runge-Kutta clasică de ordinul patru (4) tabela Butcher este:

$$\begin{array}{c|cccc} 0 & 0 & & & \\ \frac{1}{2} & \frac{1}{2} & 0 & & \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{2}{6} & \frac{2}{6} & \frac{1}{6} \end{array}$$

1.1 Probleme

1. Implementaţi metoda Runge-Kutta de ordinul 4.

2. Testaţi rutina dumneavoastră pe exemple ale căror soluţii pot fi exprimate prin cuadraturi şi reprezentaţi pe acelaşi grafic soluţia exactă.

3. Implementaţi următoarele metode: Euler, Euler modificată, Heun.

1.2 Probleme practice

Rezolvaţi problemele următoare. Comparaţi soluţia aproximativă cu cea exactă:

1.

$$\begin{aligned} &y' = x^2 - y, x \in [0,4], \\ &y(0) = 1. \end{aligned}$$

Soluţia exactă $y(x) = x^2 - 2x + 2 - e^x$.

2.

$$\begin{aligned} &y' = \frac{y^2}{1+x^2}; \\ &y(1) = -\frac{\pi}{4}. \end{aligned}$$

3.

$$\begin{aligned} &y' = \frac{1}{1+x^2} - y^2; \\ &y(0) = 0. \end{aligned}$$

Soluţia exactă:

$$y = \frac{x}{1+x^2}.$$

4.

$$\begin{aligned} &y' = -y^2, \qquad x \in [0,5] \\ &y(0) = 1. \end{aligned}$$

Soluţia exactă: $y(x) = 1/(1+x)$.

5.

$$y' = -y + 2\cos x, \quad x \in [0, 2\pi]$$
$$y(0) = 1.$$

Soluţia exactă: $y(x) = \cos x + \sin x$.

2 Sisteme de ecuaţii diferenţiale ordinare şi ecuaţii de ordin superior

Rezolvaţi următoarele EDO şi sisteme de EDO. Comparaţi soluţia exactă şi cea aproximativă. Găsiţi soluţiile şi cu rezolvitori MATLAB.

1.

$$u_1' = 3u_1 + 2u_2, \quad t \in [0,1], \quad u_1(0) = 0$$
$$u_2' = 4u_1 + u_2, \quad t \in [0,1], \quad u_2(0) = 1.$$

$h = 0.1$, soluţia exactă $u_1(t) = \frac{1}{3}(e^{5t} - e^{-t}), \quad u_2(t) = \frac{1}{3}(e^{5t} + 2e^{-t})$.

2.

$$u_1' = -4u_1 - 2u_2 + \cos t + 4\sin t,\ u_1(0) = 1,$$
$$u_2' = 3u_1 + u_2 - 3\sin t,\ u_2(0) = -1, t \in [0,2]$$

$h = 0.1$, soluţia exactă

$$u_1(t) = 2e^{-t} - 2e^{-2t} + \sin t,$$
$$u_2(t) = -3e^{-t} + 2e^{-2t}.$$

3.

$$u_1' = u_2, \quad u_1(0) = 3,$$
$$u_2' = -u_1 + 2e^{-t} + 1, \quad u_2(0) = 0,$$
$$u_3' = -u_1 + e^{-t} + 1, \quad u_3(0) = 1, \quad t \in [0,1].$$

$h = 0.1$, soluţia exactă

$$u_1(t) = \cos t + \sin t + 1,$$
$$u_2(t) = -\sin t + \cos t - e^{-t},$$
$$u_3(t) = -\sin t + \cos t.$$

4.

$$\begin{aligned} &t^2y'' - 2ty' + 2y = t^3 \ln t, \quad t \in [0,2] \\ &y(0) = 1, y'(0) = -1, \end{aligned}$$

$h = 0.05$, soluţia exactă

$$y(t) = \frac{7}{4}t + \frac{t^3}{2}\ln t - \frac{3}{4}t^3.$$

5.

$$\begin{aligned} &y''' = -6y^4, \quad t \in [1, 1.9] \\ &y(1) = -1, \quad y'(1) = -1, \quad y''(1) = -2, \end{aligned}$$

$h = 0.05$; soluţia exactă

$$y(t) = \frac{1}{t-2}.$$

3 Controlul pasului

Pentru o descriere sintetică a metodelor Runge-Kutta cu pas variabil tabela Butcher se completează cu o linie suplimentară care serveşte la calculul lui Φ^* (şi deci a lui $r(x, y; h)$):

$$\begin{array}{c|ccccc} \mu_1 & \lambda_{11} & \lambda_{12} & \dots & \lambda_{1r} & \\ \mu_2 & \lambda_{21} & \lambda_{22} & \dots & \lambda_{2r} & \\ \vdots & \vdots & \vdots & \dots & \vdots & \\ \mu_r & \lambda_{r1} & \lambda_{r2} & \dots & \lambda_{rr} & \\ \hline & \alpha_1 & \alpha_2 & \dots & \alpha_r & \\ & \alpha_1^* & \alpha_2^* & & \alpha_r^* & \alpha_{r+1}^* \end{array}$$

Ca exemplu, tabela 1 este tabela Butcher pentru metoda Bogacki-Shampine. Ea stă la baza rezolvitorului `ode23` din MATLAB.

Un alt exemplu important este DORPRI5 sau RK5(4)7FM, o pereche cu ordinele 4-5 şi cu 7 stadii (tabela 2). Aceasta este o pereche foarte eficientă, ea stând la baza rezolvitorului `ode45` din MATLAB, dar şi a altor rezolvitori importanţi.

Algoritmul 2 încearcă să dea sugestii pentru implementarea unei metode Runge-Kutta cu pas variabil când se cunoaşte tabela Butcher. *ttol* este produsul dintre *tol* şi factorul de siguranţă (0.8 sau 0.9).

μ_j	λ_{ij}			
0	0			
$\frac{1}{2}$	$\frac{1}{2}$	0		
$\frac{3}{4}$	0	$\frac{3}{4}$	0	
1	$\frac{2}{9}$	$\frac{3}{9}$	$\frac{4}{9}$	0
α_i	$\frac{2}{9}$	$\frac{3}{9}$	$\frac{4}{9}$	0
α_i^*	$\frac{7}{24}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{8}$

Tabela 1: Tabela Butcher pentru metoda Bogacki-Shampine

μ_j	λ_{ij}						
0	0						
$\frac{1}{5}$	$\frac{1}{5}$	0					
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$	0				
$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$	0			
$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{25360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$	0		
1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$	0	
1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	0
α_i	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	0
α_i^*	$\frac{5179}{57600}$	0	$\frac{7571}{16695}$	$\frac{393}{640}$	$-\frac{92097}{339200}$	$\frac{187}{2100}$	$\frac{1}{40}$

Tabela 2: Perechea inclusă RK5(4)7FM (DORPRI5)

Algorithm 2 Fragment de pseudocod ce ilustrează implementarea unei metode RK cu pas variabil

```
done := false;
loop
  K_{:,1} := f(x, y);
  for i = 2  to s do
    w := y + hK_{:,1:i−1} λ^T_{i,1:i−1};
    K_{:,i} := f(x + μ_i h, w);
  end for
  δ := h max (|K(α* − α)^T|); {estimarea erorii}
  β := (δ/ttol)^{1/(1+p)}; {raport lung. pas}
  if δ < tol then
    {acceptare pas}
    y := y + h(Kα^T); {actualizare y}
    x := x + h;
    if done then
      EXIT {terminare şi ieşire}
    end if
    h := h/ max(β, 0.1); {predicţie pas următor}
    if x + h > x_end then
      h := x_end − x; {reducere pas la capăt}
      done := true;
    end if
  else
    {respingere pas}
    h := h/ min(β, 10); {reducere pas}
    if done then
      done := false;
    end if
  end if
end loop
```

3.1 Probleme

1. Implementaţi un mecanism de control al pasului pentru una din metodele descrise de tabelele Butcher precedente.

2. Testaţi rutina precedentă pentru o EDO scalară, un sistem şi o EDO de ordin superior.