XI EMED - Encontro Mineiro de Equações Diferenciais
20 a 22 de Novembro de 2017
Poços de Caldas-MG

Implementação dos métodos de Runge-Kutta, Dormand-Prince e Adams-Bashforth-Moulton na linguaguem de programação Fortran para solução de equações diferenciais ordinárias

Viliam C. da Silveira, **Gervásio A. Degrazia**,
UFSM - Universidade Federal de Santa Maria, Departamento de Meteorologia,
Santa Maria - RS
E-mail: viliamcardoso@gmail.com, gervasiodegrazia@gmail.com.

1 Introdução

O objetivo desse trabalho é implementar na linguagem de programação Fortran [3] os métodos de Runge-Kutta [2], Dormand-Prince [4] e Adams-Bashforth-Moulton [1] para solução de equações diferenciais ordinárias (EDO) de ordem um. Esse trabalho é de grande importância, pois uma série de aplicações baseadas nessas implementações podem serem feitas, como por exemplo, desenvolvimento de modelos para previsão do tempo.

2 Metodologia

Vamos implementar o método clássico de Runge-Kutta de quarta ordem. Devemos seguir o seguinte procedimento

$$k_1 = f(x_i, y_i) \tag{1}$$

$$k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right) \tag{2}$$

$$k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_2\right) \tag{3}$$

$$k_4 = f(x_i + h, y_i + hk_3) \tag{4}$$

Assim, a solução no próximo passo é dada por

$$y_{i+1} = y_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{5}$$

O método de Dormand-Price é implementado da seguinte maneira

$$k_1 = hf(x_i, y_i) \tag{6}$$

$$k_2 = hf\left(x_i + \frac{1}{5}h, y_i + \frac{1}{5}k_1\right) \tag{7}$$

$$k_3 = hf\left(x_i + \frac{3}{10}h, y_i + \frac{3}{40}k_1 + \frac{9}{40}k_2\right) \tag{8}$$

$$k_4 = hf\left(x_i + \frac{4}{5}h, y_i + \frac{44}{45}k_1 - \frac{56}{15}k_2 + \frac{32}{9}k_3\right) \tag{9}$$

$$k_5 = hf\left(x_i + \frac{8}{9}h, y_i + \frac{19372}{6561}k_1 - \frac{25360}{2187}k_2 + \frac{64448}{6561}k_3 - \frac{212}{729}k_4\right) \tag{10}$$

$$k_6 = hf\left(x_i + h, y_i + \frac{9017}{3168}k_1 - \frac{355}{33}k_2 - \frac{46732}{5247}k_3 + \frac{49}{176}k_4 - \frac{5103}{18656}k_5\right) \tag{11}$$

$$k_7 = hf\left(x_i + h, y_i + \frac{35}{384}k_1 + \frac{500}{1113}k_3 + \frac{125}{192}k_4 - \frac{2187}{6784}k_5 + \frac{11}{84}k_6\right) \tag{12}$$

A solução no próximo passo é dada por

$$y_{i+1} = y_i + \frac{35}{384}k_1 + \frac{500}{1113}k_3 + \frac{125}{192}k_4 - \frac{2187}{6784}k_5 + \frac{11}{84}k_6 \tag{13}$$

O método de Adams-Bashforth-Moulton é um método preditor-corretor de passo múltiplo. O preditor explícito de passo 4 é dado por

$$y_{i+1} = y_i + \frac{h}{24}(55f(x_i, y_i) - 59f(x_{i-1}, y_{i-1}) + 37f(x_{i-2}, y_{i-2}) - 9f(x_{i-3}, y_{i-3})) \tag{14}$$

O corretor implícito é dado por

$$y_{i+1} = y_i + \frac{h}{24}(9f(x_{i+1}, y_{i+1}) + 19f(x_i, y_i) - 5f(x_{i-1}, y_{i-1}) + f(x_{i-2}, y_{i-2})) \tag{15}$$

Os valores $f(x_1, y_1)$, $f(x_2, y_2)$ e $f(x_3, y_3)$ são calculados pelo método de Dormand-Prince de passo simples.

3 Resultados

As Figuras 1 e 2 mostram as implementações na linguagem de programação fortran dos métodos descritos acima.

```fortran
allocate(VetX(m+1))
allocate(VetY(m+1))
!------------------------------------------------------------------
write(*,11)'Método de Runge-Kutta - ordem 4'
11 format(a33)
write(*,22)'i','x','y'
22 format(a5,a10,a10)
h = (b-a)/m ; xt = a ; yt = y0 ; VetX(1) = xt ; VetY(1) = yt
write(*,33)0,xt,yt
33 format(i5,f13.5,f10.5)
do i = 1,m
 x = xt ; y = yt ; call funcao(x,y,Fxy) ; k1 = Fxy
 x = xt + h/2 ; y = yt + h/2*k1 ; call funcao(x,y,Fxy) ; k2 = Fxy
 y = yt + h/2*k2 ; call funcao(x,y,Fxy) ; k3 = Fxy
 x = xt + h ; y = yt + h*k3 ; call funcao(x,y,Fxy) ; k4 = Fxy
 xt = a + i*h ; yt = yt + h/6*(k1+2*(k2+k3)+k4)
 write(*,44)i,xt,yt
 44 format(i5,f13.5,f10.5)
 VetX(i+1) = xt ; VetY(i+1) = yt
end do
```

```fortran
allocate(VetX(m+1))
allocate(VetY(m+1))
allocate(Erro(m+1))
!Chama a subrotina DOPRI--------------------------------------
h = (b-a)/m
call DOPRI(a,a+3*h,3,y0,VetX,VetY,Erro)
!------------------------------------------------------------
write(*,11)'Método de Adams-Bashforth-Moulton - ordem 4'
11 format(a45)
write(*,22)'i','x','y','Erro'
22 format(a5,a10,a10,a12)
do i = 1,4
 write(*,33)i-1,VetX(i),VetY(i),Erro(i)
 33 format(i5,f13.5,f10.5,e12.3e2)
end do
do i = 4,m
 x = VetX(i-3) ; y = VetY(i-3) ; call funcao(x,y,Fxy) ; f0 = Fxy
 x = VetX(i-2) ; y = VetY(i-2) ; call funcao(x,y,Fxy) ; f1 = Fxy
 x = VetX(i-1) ; y = VetY(i-1) ; call funcao(x,y,Fxy) ; f2 = Fxy
 x = VetX(i) ; y = VetY(i) ; call funcao(x,y,Fxy) ; f3 = Fxy
 Ypre = h*(55*f3 - 59*f2 + 37*f1 - 9*f0)/24 + VetY(i)
 VetY(i+1) = Ypre ; VetX(i+1) = a + i*h ; x = VetX(i+1)
 do j = 1,2
  y = VetY(i+1)
  call funcao(x,y,Fxy)
  f4 = Fxy
  Ycor = h*(9*f4 + 19*f3 -5*f2 + f1)/24 + VetY(i)
  VetY(i+1) = Ycor
 end do
 Erro2 = abs(Ycor-Ypre)*19./270
 write(*,44)i,VetX(i+1),VetY(i+1),Erro2
 44 format(i5,f13.5,f10.5,e12.3e2)
end do
```

Figura 1: Implementação do método de Runge-Kutta e Adams-Bashforth-Moulton para solução de problema de valor inicial.

Para mostrar o desempenho da implementação das fórmulas dadas acima (ver Tabela 1), vamos considerar o seguinte problema de valor inicial

$$y' = 2xy \tag{16}$$

no intervalo $[1, 2]$, com 10 subintervalos e $y(1) = 1$. A solução exata da EDO acima é dada por $y(x) = e^{x^2-1}$.

```fortran
allocate(VetX(m+1))
allocate(VetY(m+1))
allocate(EG(m+1))
!-------------------------------------------------------------------------------------------
write(*,11)'Método de Dormand-Prince (DOPRI)'
11 format(a33)
write(*,22)'i','x','y','Erro'
22 format(a5,a10,a10,a12)
a21 = 1./5
a31 = 3./40;a32 = 9./40
a41 = 44./45;a42 = -56./15;a43 = 32./9
a51 = 19372./6561;a52 = -25360./2187;a53 = 64448./6561;a54 = -212./729
a61 = 9017./3168;a62 = -355./33;a63 = 46732./5247;a64 = 49./176;a65 = -5103./18656
a71 = 35./384;a73 = 500./1113;a74 = 125./192;a75 = -2187./6784;a76 = 11./84
c2 = 1./5;c3 = 3./10;c4 = 4./5;c5 = 8./9;c6 = 1.;c7 = 1.
e1 = 71./57600;e3 = -71./16695;e4 = 71./1920;e5 = -17253./339200;e6 = 22./525;e7 = -1./40
h = (b-a)/m ; xt = a ; yt = y0 ; VetX(1) = xt ; VetY(1) = yt ; EG(1) = 0
write(*,33)0,xt,yt
33 format(i5,f13.5,f10.5)
do i = 1,m
 x = xt ; y = yt ; call funcao(x,y,Fxy) ; k1 = h*Fxy
 x = xt + c2*h ; y = yt + a21*k1 ; call funcao(x,y,Fxy) ; k2 = h*Fxy
 x = xt + c3*h ; y = yt + a31*k1 + a32*k2 ; call funcao(x,y,Fxy) ; k3 = h*Fxy
 x = xt + c4*h ; y = yt + a41*k1 + a42*k2 + a43*k3 ; call funcao(x,y,Fxy) ; k4 = h*Fxy
 x = xt + c5*h ; y = yt + a51*k1 + a52*k2 + a53*k3 + a54*k4 ; call funcao(x,y,Fxy) ; k5 = h*Fxy
 x = xt + c6*h ; y = yt + a61*k1 + a62*k2 + a63*k3 + a64*k4 + a65*k5 ; call funcao(x,y,Fxy) ; k6 = h*Fxy
 x = xt + c7*h ; y = yt + a71*k1 + a73*k3 + a74*k4 + a75*k5 + a76*k6 ; call funcao(x,y,Fxy) ; k7 = h*Fxy
 xt = a + i*h ; yt = yt + a71*k1 + a73*k3 + a74*k4 + a75*k5 + a76*k6
 ErroGlobal = e1*k1 + e3*k3 + e4*k4 + e5*k5 + e6*k6 + e7*k7
 VetX(i+1) = xt ; VetY(i+1) = yt ; EG(i+1) = ErroGlobal
 write(*,44)i,xt,yt,ErroGlobal
 44 format(i5,f13.5,f10.5,e12.3e2)
end do
```

Figura 2: Implementação do método de Dormand-Prince para solução de problema de valor inicial.

i	x_i	y_i (Runge)	y_i (Dormand)	y_i (Adams)	$y(x_i)$ (solução exata)
0	1,0	1,00000	1,00000	1,00000	1,00000
1	1,1	1,23367	1,23368	1,23368	1,23368
2	1,2	1,55270	1,55271	1,55271	1,55271
3	1,3	1,99369	1,99372	1,99372	1,99372
4	1,4	2,61163	2,61170	2,61199	2,61170
5	1,5	3,49021	3,49034	3,49122	3,49034
6	1,6	4,75855	4,75882	4,76082	4,75882
7	1,7	6,61883	6,61937	6,62347	6,61937
8	1,8	9,39225	9,39334	9,40137	9,39333
9	1,9	13,59690	13,59907	13,61448	13,59905
10	2,0	20,08127	20,08558	20,11488	20,08554

Tabela 1: Resultados da implementação dos métodos para solução de EDO de ordem um

4 Conclusões

Os resultados mostram um bom desempenho das técnicas para solução de problema de valor inicial de equações diferenciais ordinárias. O método de Dormand-Prince foi o que apresentou melhor desempenho, ou seja, foi o que mais se aproximou da solução exata.

Referências

[1] F. F. C. Filho, *Algoritmos Numéricos*, 428 pages. LTC EDITORA, Belo Horizonte, 2007.

[2] I. Faragó, Numerical Methods for Ordinary Differential Equations Disponível em: *http://www.cs.elte.hu/~faragois/ODE_angol.pdf*.

[3] S. J. Chapman, *Fortran 95/2003 for scientists and engineers 3rd Edition*, 982 pages. McGraw-Hill Companies, United States of America, 2007.

[4] T. Kimura, On Dormand-Prince Method, September 24,2009 Disponível em: *http://depa.fquim.unam.mx/amyd/archivero/DormandPrince_19856.pdf*.

Palavras-chave: *EDO, Fortran*

Agradecimentos: *Agradecemos à CAPES pelo apoio financeiro.*