Métodos Numéricos para Equações Diferenciais

CCT - UDESC

Prof. Fernando Deeke Sasse

Prova 1

1. As equações abaixo são típicas em problemas de vibração.

$$\text{(a)}\ \ y'' + 0.5(y^2-1)y' + y = 0 \qquad y(0)=1 \qquad y'(0)=0$$
$$\text{(b)}\ \ y'' = y\cos 2x \qquad y(0)=0 \qquad y'(0)=1$$

Resolva estes problemas utilizando RK4 e RKF45, para x de 0 até 20, gerando tabelas e plotando os resultados.

Solução

(a) Inicialmente usaremos RK4

```
> restart;
> e1:=diff(y(x),x$2)+0.5*(y(x)^2-1)*diff(y(x),x)+y(x)=0;
```

$$e1 := \frac{d^2}{dx^2} y(x) + 0.5\,(y(x)^2-1)\left(\frac{d}{dx}y(x)\right) + y(x) = 0 \qquad \textbf{(1.1)}$$

```
> ic := [y(0)=1, D(y)(0)=0];
```

$$ic := [y(0)=1, D(y)(0)=0] \qquad \textbf{(1.2)}$$

```
> left := 0;
  right := 20.;
```

$$left := 0$$
$$right := 20. \qquad \textbf{(1.3)}$$

```
> numsteps := 100;
```

$$numsteps := 100 \qquad \textbf{(1.4)}$$

```
> h := (right-left)/numsteps;
```

$$h := 0.2000000000 \qquad \textbf{(1.5)}$$

```
> S1:=dsolve({e1,y(0)=1, D(y)(0)=0},numeric,method=classical
  [rk4], stepsize=h)  ;
```

$S1$:= **proc** ($x\_classical$) ... **end proc** **(1.6)**

```
> S2 := dsolve({e1, y(0) = 1, (D(y))(0) = 0}, numeric, method =
  rkf45, initstep = h);
```

$S2$:= **proc** (x_rkf45) ... **end proc** **(1.7)**

```
> op(2, S1(4.43)[2])
```

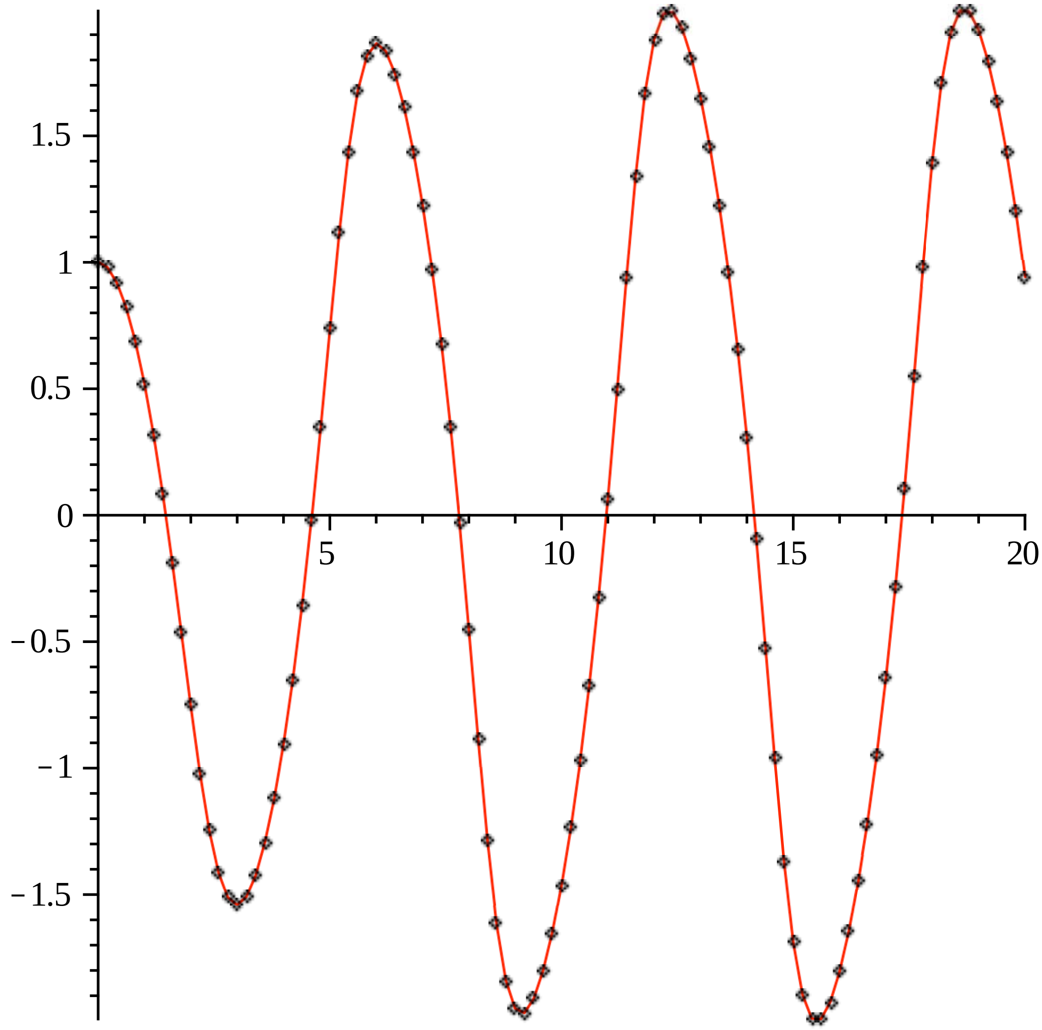
$$-0.313471637389154401 \qquad \textbf{(1.8)}$$

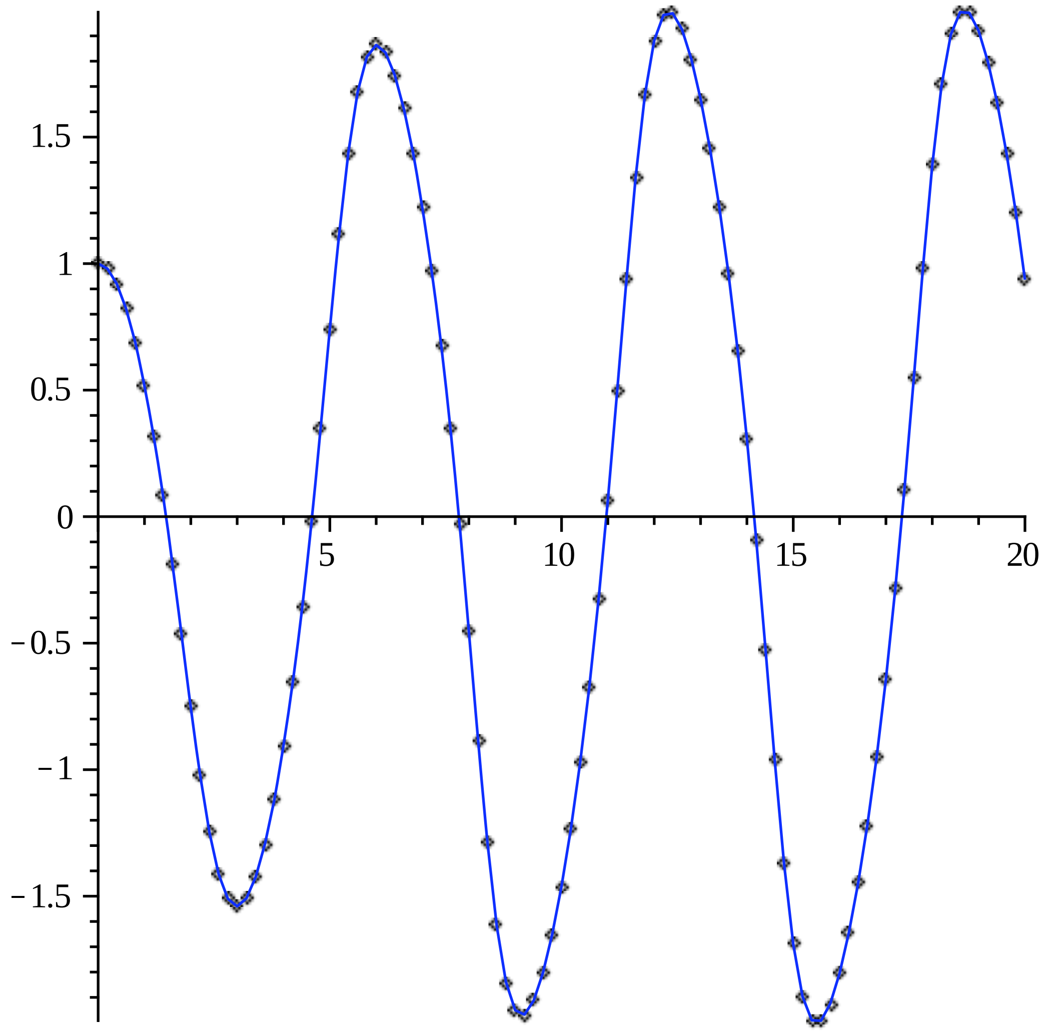
```
> op(2, S1(0)[2]);
```

$$1. \qquad \textbf{(1.9)}$$

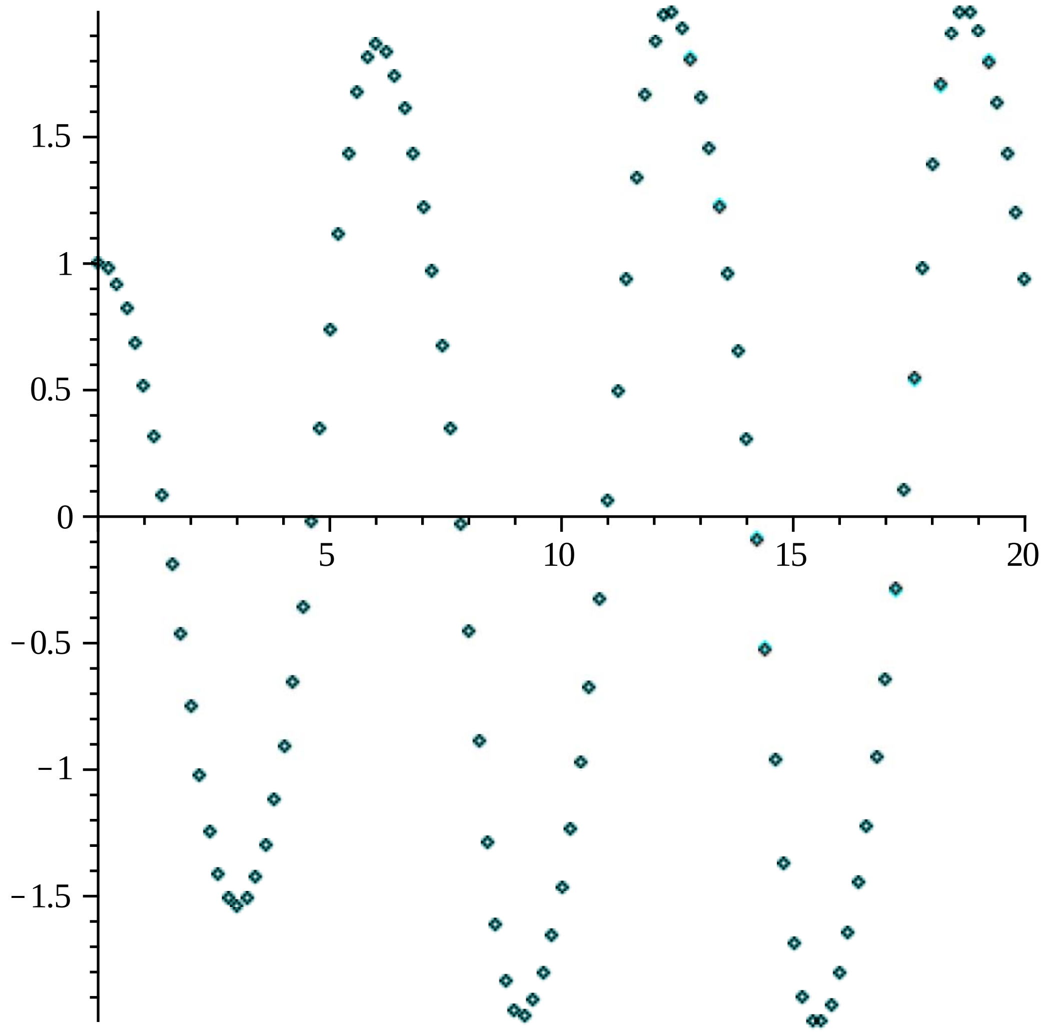
```
> LL1:=[seq([op(2, S1(i*h)[1]), op(2,S1(i*h)[2])],i=0..
  numsteps)]:
> LL2 := [seq([op(2, S2(i*h)[1]), op(2, S2(i*h)[2])], i = 0 ..
  numsteps)]:
> with(plots):
> p1 := pointplot(LL1,color=cyan):
```

```
> p2:=pointplot(LL1,connect=true,color=red):
> p3:=pointplot(LL2,color=black):
> p4:=pointplot(LL2,connect=true,color=blue):
> display([p2, p3]); display([p3, p4]);
```

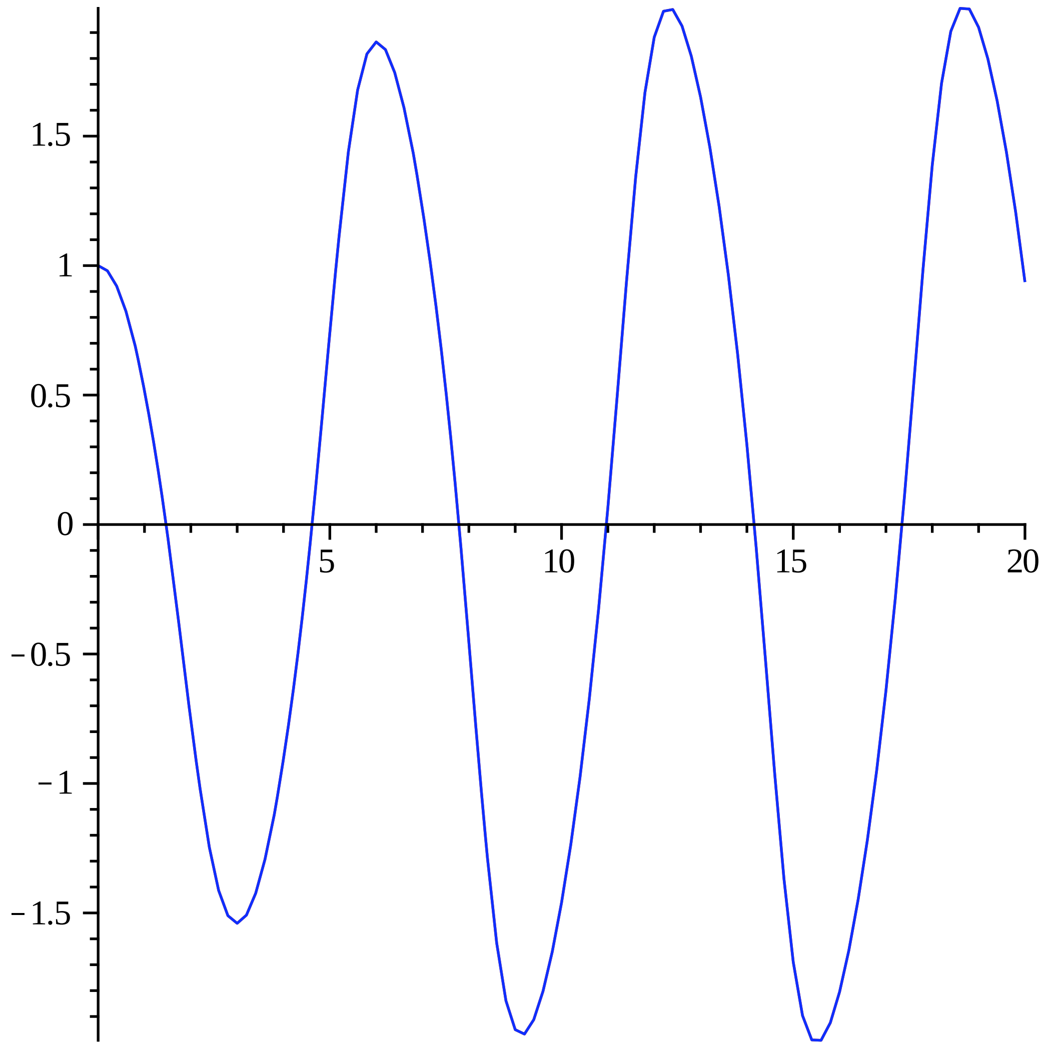





```
> display([p1, p3]);
```




```
> display([p2, p4]);
```




```
> e1;
```

$$\frac{d^2}{dx^2} y(x) + 0.5 \left(y(x)^2 - 1 \right) \left(\frac{d}{dx} y(x) \right) + y(x) = 0 \qquad \textbf{(1.10)}$$

A EDO pode ser decomposta em duas equações de primeira ordem acopladas definindo uma nova variável u = y' .

```
> Digits:=18:
> e2:=subs(diff(y(x),x)=u(x),e1);
```

$$e2 := \frac{d}{dx} u(x) + 0.5 \left(y(x)^2 - 1 \right) u(x) + y(x) = 0 \qquad \textbf{(1.11)}$$

Pelo método de Runge-Kutta de quarta ordem, para o intervalo [0,20], com 1000 divisões, O sistema é agora

$$y' = u, \quad y(0) = 1,$$

$$u' = -0.5 \left(y(x)^2 - 1 \right) u(x) - y(x), \quad u(0) = 0.$$

```
> h:=0.2; y0:=1.; x0:=0; u0:=0.;
```

$$h := 0.2$$

$$y0 := 1.$$

$$x0 := 0$$

(1.12)

$$u0 := 0. \qquad (1.12)$$

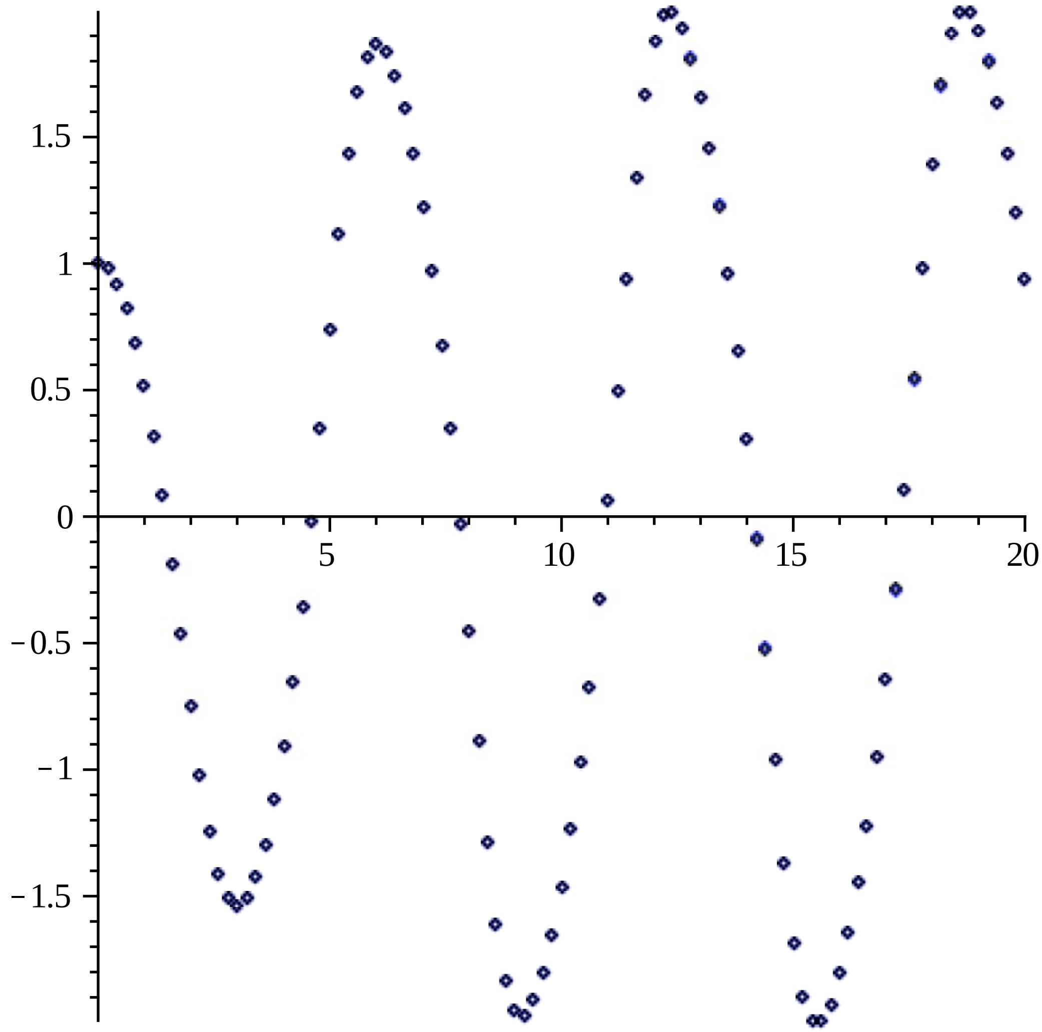
```
> f:=-(.5*(y^2-1))*u-y;
```

$$f := -0.5\,(y^2-1)\,u - y \qquad (1.13)$$

```
> F:=unapply(f,x,y,u);
```

$$F := (x, y, u) \rightarrow -0.5\,(y^2-1)\,u - y \qquad (1.14)$$

```
> drk4:=proc(x0,y0,u0,h,nit,F)
     local K1, K2,K3,K4, L1, L2, L3,L4,x,y,i,LL3,u;
     x[0] := x0; y[0] := y0; u[0] := u0;
     LL3:=[[x[0],y[0]]];
     for i from 1 to nit do
>       L1:=evalf(u[i-1]);
        K1:=evalf(F(x[i-1],y[i-1],u[i-1]));
>       L2:= evalf(u[i-1]+h*K1/2);
        K2:=evalf(F(x[i-1]+h/2,y[i-1]+h*L1/2,u[i-1]+h*K1/2));
>       L3:=evalf(u[i-1]+h*K2/2);
        K3:=evalf(F(x[i-1]+h/2,y[i-1]+h*L2/2,u[i-1]+h*K2/2));
>       L4:=evalf(u[i-1]+h*K3);
        K4:=evalf(F(x[i-1]+h,y[i-1]+h*L3,u[i-1]+h*K3));
        u[i]:=evalf(u[i-1]+(h/6)*(K1+2*K2+2*K3+K4));
        y[i]:=evalf(y[i-1]+(h/6)*(L1+2*L2+2*L3+L4));
>       x[i]:=evalf(x[0]+i*h);
>       LL3:=[op(LL3),[x[i],y[i]]];
>    od:
  end:
> LL3 := drk4(x0, y0, u0, h, 100, F):
> p4 := pointplot(LL3, color = blue):
> display([p4, p3]);
```




```
>
```

```
>
> drk4b:=proc(f,g,t0,tf,y0,u0,n)
    local k1,k2,k3,k4,l1,l2,l3,l4,i,k,l,u,y,t,L,h;
    L := [[t0, y0]];
    h := evalf((tf-t0)/n);
    t:=t0:y:=y0:u:=u0:
    for i to n do
      k1 := f(t, u, y);
      l1 := g(t, u, y);
      k2 := f(t+(1/2)*h, u+(1/2)*h*k1, y+(1/2)*h*l1);
      l2 := g(t+(1/2)*h, u+(1/2)*h*k1, y+(1/2)*h*l1);
      k3 := f(t+(1/2)*h, u+(1/2)*h*k2, y+(1/2)*h*l2);
      l3 := g(t+(1/2)*h, u+(1/2)*h*k2, y+(1/2)*h*l2);
      k4 := f(t+h, u+h*k3, y+h*l3);
      l4 := g(t+h, u+h*k3, y+h*l3);
      k := (k1+2*k2+2*k3+k4)*(1/6);
      l := (l1+2*l2+2*l3+l4)*(1/6);
      u := u+h*k;
      y := y+h*l;
      t := t+h;
      L := [op(L), [t, y]]
```

```
        end do;
    end:
```

> $g := (t, u, y) \rightarrow -0.5\left(y^2-1\right)u - y$:
> $f := (t, u, y) \rightarrow \mathrm{u}$:

> $t0 := 0$:
> $u0 := 0 : y0 := 1$:
> $tf := 20$:

> $n := 100$:

>

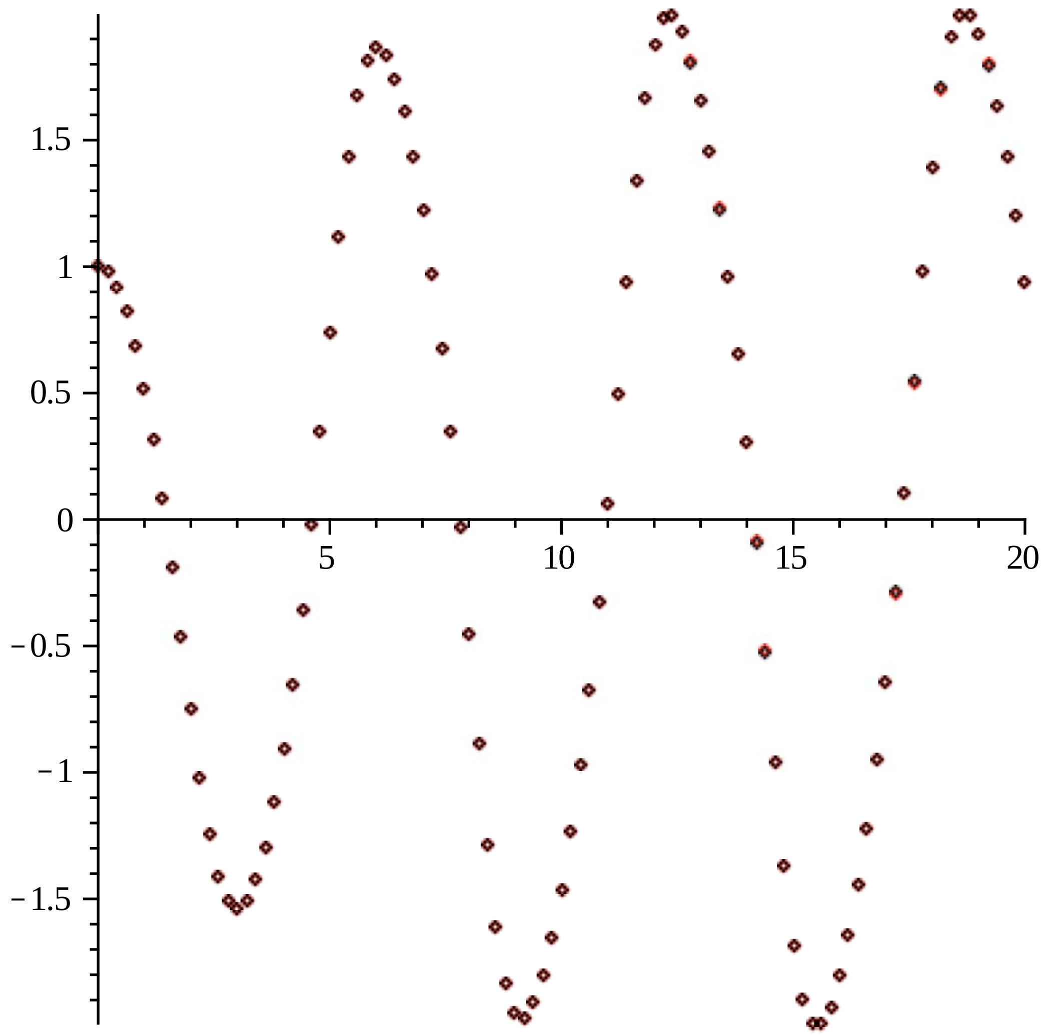
> $L := drk4b(g, f, t0, tf, y0, u0, n)$:

```
> with(plots):
```

> L:

> $p5 := pointplot(L, color = red)$:

> $display([p5, p3])$;



>

(b)

> $g := (t, u, y) \rightarrow y \cdot \cos(2 \cdot t)$:
> $f := (t, u, y) \rightarrow \mathrm{u}$:

> $e4 := diff(y(t), t\$2) - y(t)\cdot\cos(2\cdot t)$

$$e4 := \frac{d^2}{dt^2}\, y(t) - y(t)\cos(2\,t) \qquad (1.15)$$

```
> S3 := dsolve({e4, y(0) = 0, (D(y))(0) = 1}, numeric, method =
  rkf45, initstep = h);
```

$S3$:= **proc** (x_rkf45) ... **end proc** (1.16)

```
>
> LL3:=[seq([op(2, S3(i*h)[1]), op(2,S3(i*h)[2])],i=0..n)]:
> p3:=pointplot(LL3,color=black):
```

> $y0 := 0 : u0 := 1 :$

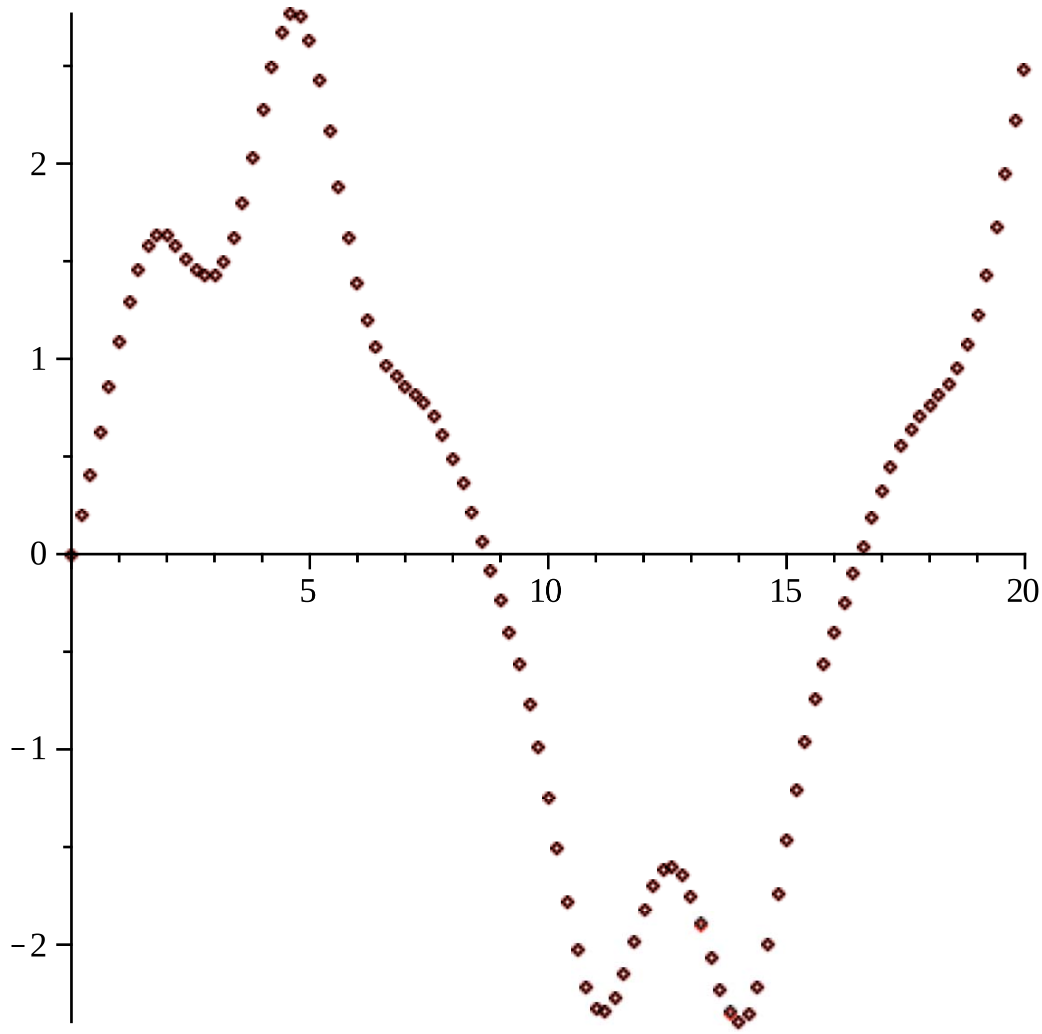
> $L := drk4b(g, f, t0, tf, y0, u0, n) :$

```
> with(plots):
```

> $L :$

> $p6 := pointplot(L, color = red) :$

> $display([p6, p3]);$



>

Novamente temos uma boa concordância entre os dois métodos.

2. Integre o PVI

$$y'' + y' - y^2 = 0 \qquad y(0) = 1 \qquad y'(0) = 0$$

de $x = 0$ *a* 3.5, usando método da série de Taylor de quarta ordem, RK4 e RKF45. Examine se o crescimento repentino em y próximo ao limite superior é real ou um artefato causado pela instabilidade. Sugestão: experimente com diferentes valores de h.

Solução

```
> restart;
> e1:=diff(y(x),x$2)+diff(y(x),x)-y(x)^2=0;
```

$$e1 := \frac{d^2}{dx^2} y(x) + \frac{d}{dx} y(x) - y(x)^2 = 0 \qquad (2.1)$$

```
> ic := [y(0)=1, D(y)(0)=0];
```

$$ic := [y(0) = 1, D(y)(0) = 0] \qquad (2.2)$$

```
> left := 0;
  right := 3.5;
```

$$left := 0$$

$$right := 3.5 \qquad (2.3)$$

```
> numsteps := 100;
```

$$numsteps := 100 \qquad (2.4)$$

```
> h := (right-left)/numsteps;
```

$$h := 0.03500000000 \qquad (2.5)$$

```
> S1:=dsolve({e1,y(0)=1, D(y)(0)=0},numeric,method=classical
  [rk4], stepsize=h)  ;
```

$S1$:= **proc** ($x\_classical$) ... **end proc** (2.6)

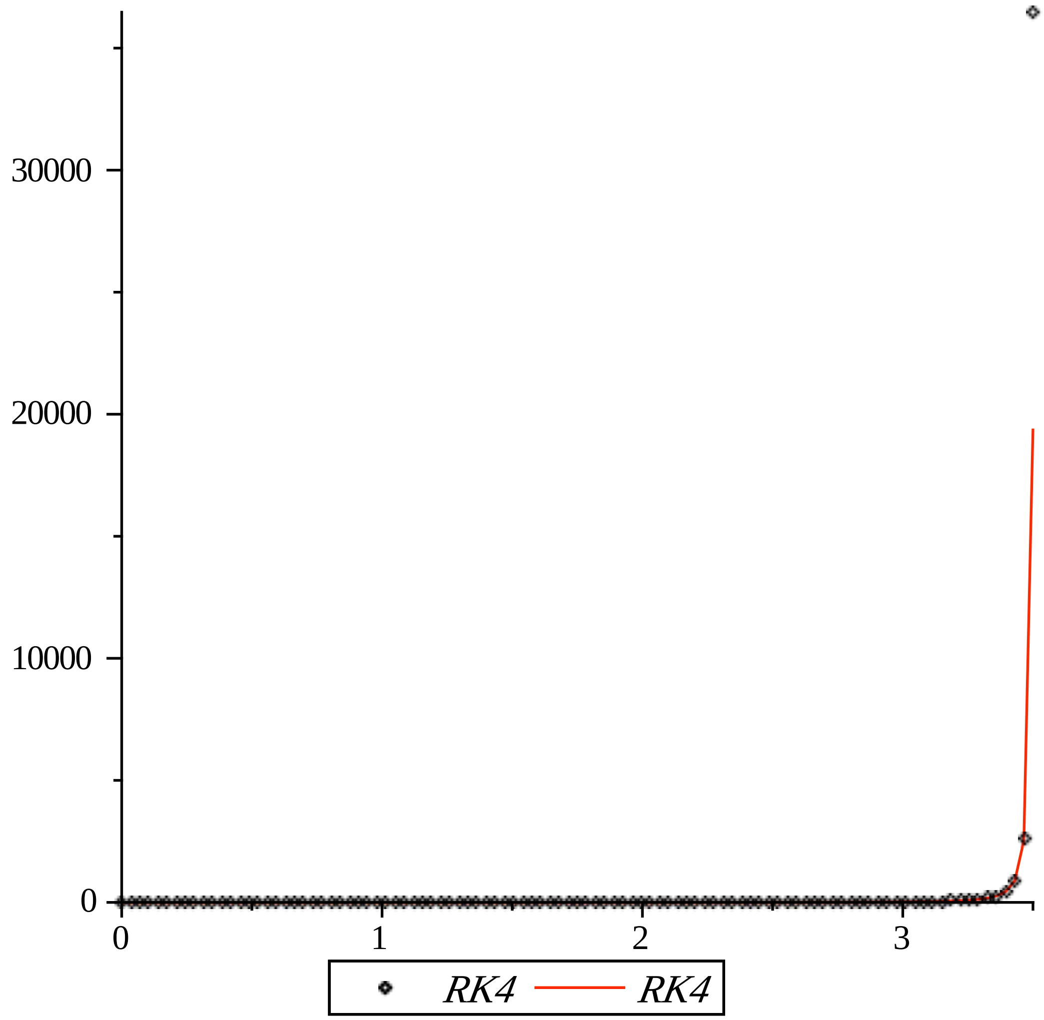
```
> S2 := dsolve({e1, y(0) = 1, (D(y))(0) = 0}, numeric, method =
  rkf45, initstep = h);
```

$S2$:= **proc** (x_rkf45) ... **end proc** (2.7)

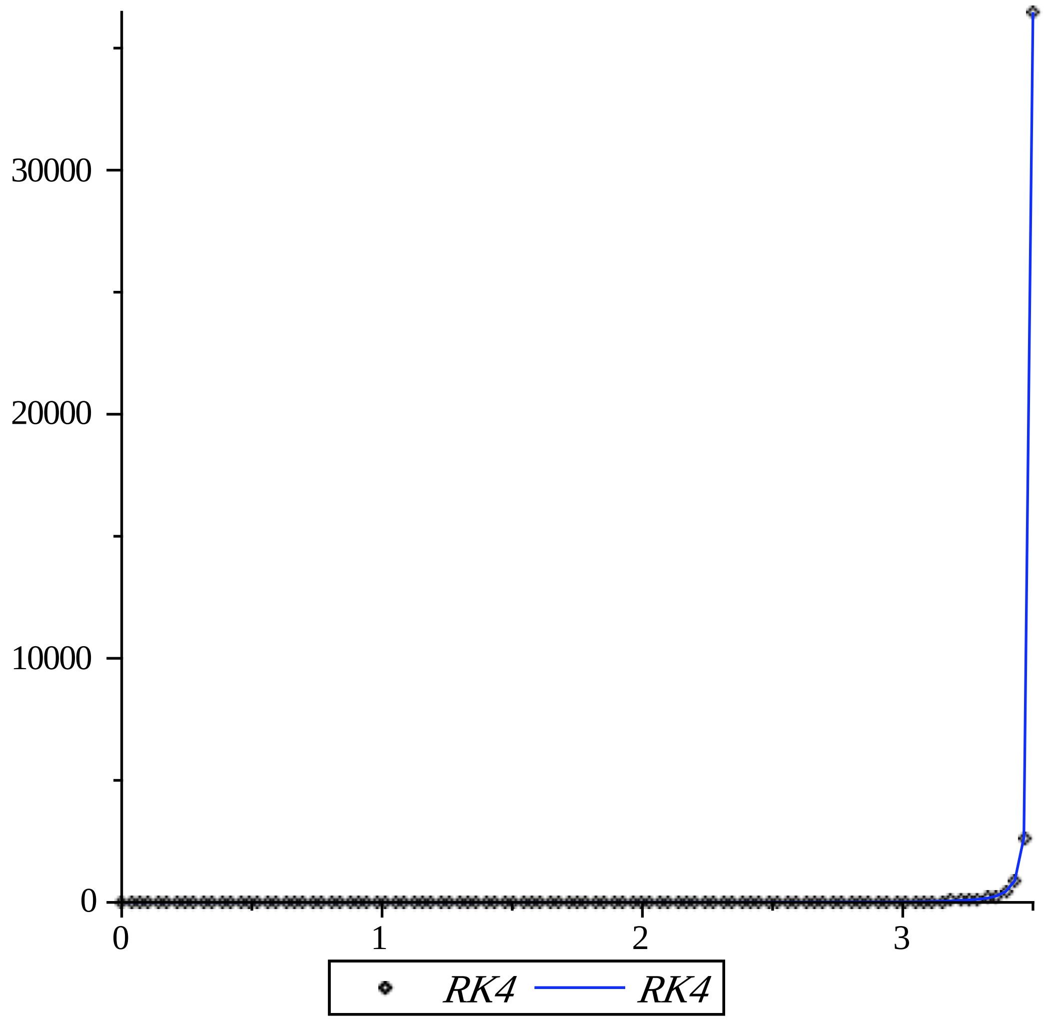
```
> op(2, S1(0)[2]);
```

$$1. \qquad (2.8)$$

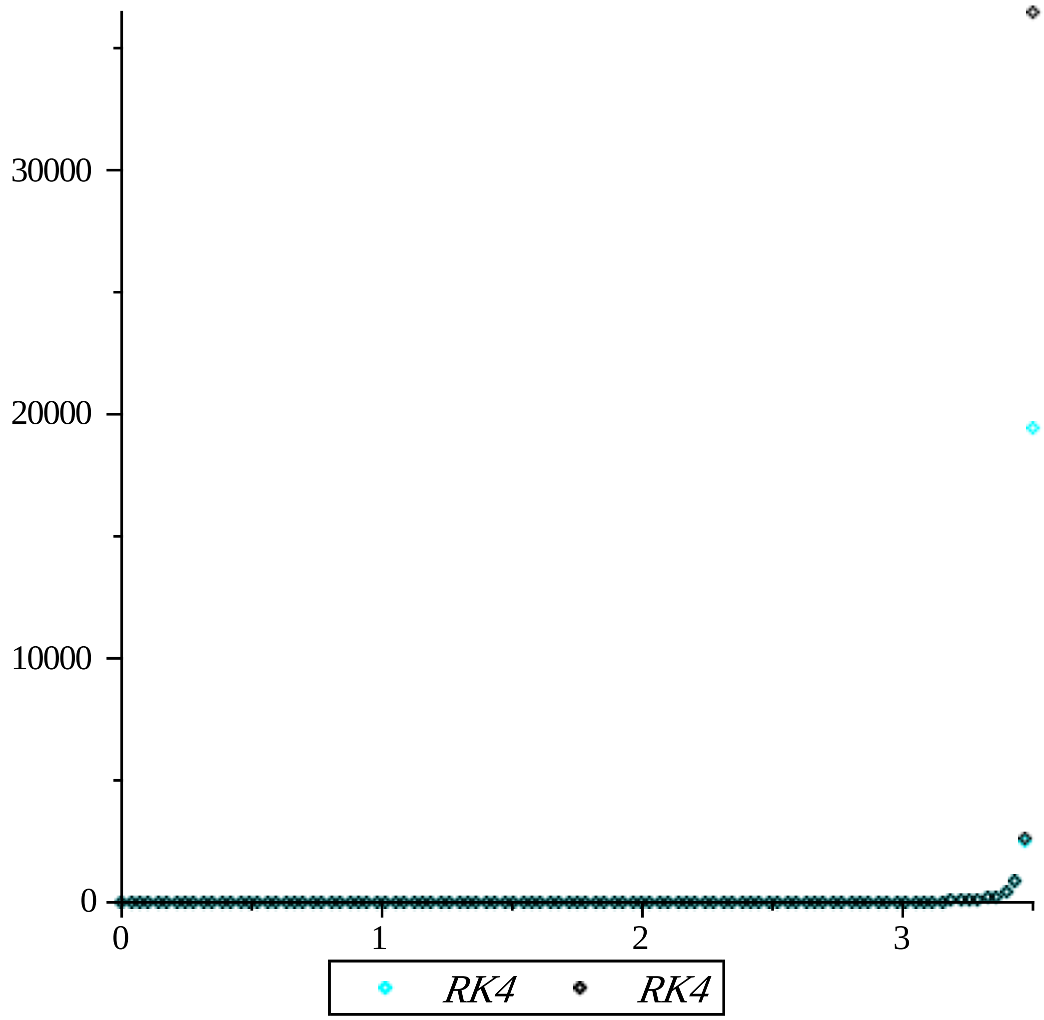
```
> LL1:=[seq([op(2, S1(i*h)[1]), op(2,S1(i*h)[2])],i=0..
  numsteps)]:
> LL2 := [seq([op(2, S2(i*h)[1]), op(2, S2(i*h)[2])], i = 0 ..
  numsteps)]:
> with(plots):
> p1 := pointplot(LL1,color=cyan,legend=RK4):
> p2:=pointplot(LL1,connect=true,color=red,legend=RK4):
> p3:=pointplot(LL2,color=black,legend=RK4):
> p4:=pointplot(LL2,connect=true,color=blue,legend=RK4):
> display([p2, p3]); display([p3, p4]);
```



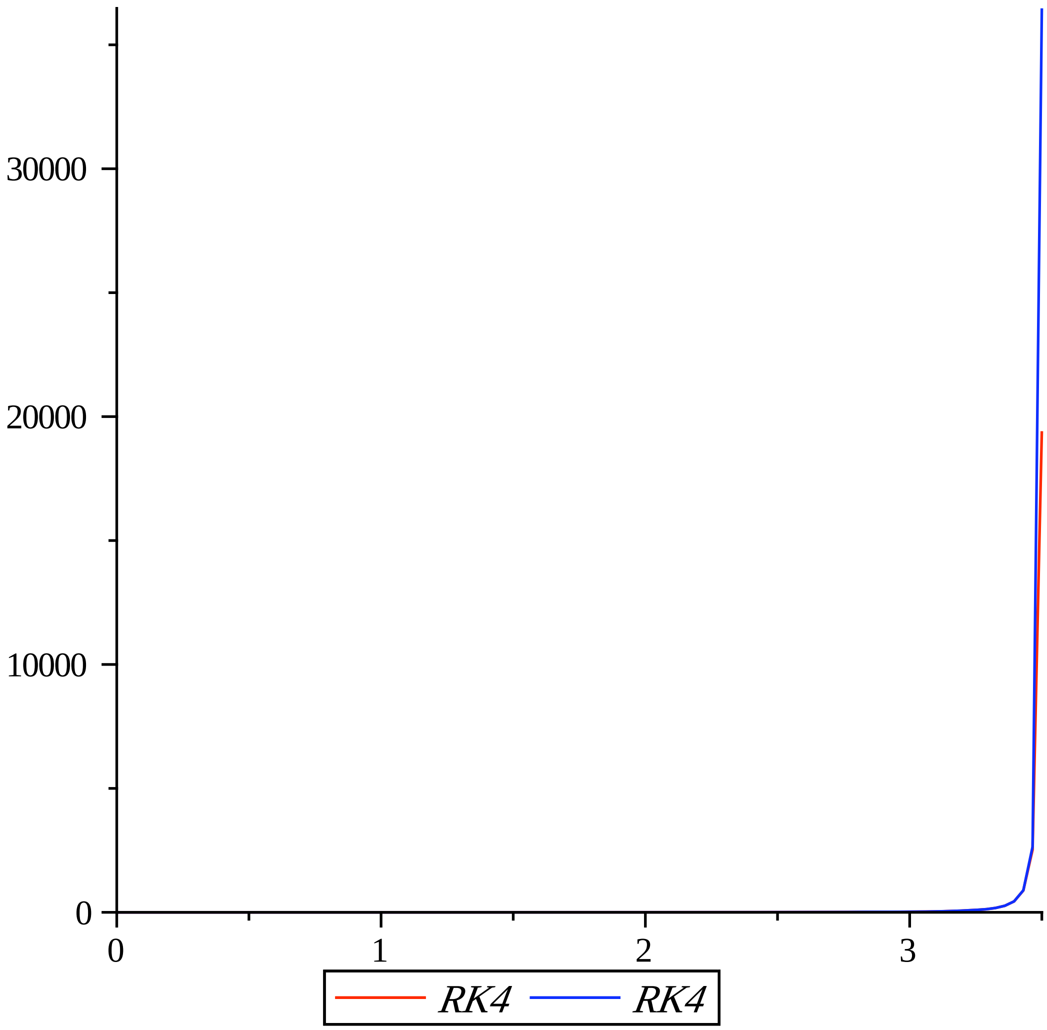
30000
20000
10000
0
0
1
2
3
RK4
RK4




```
> display([p1, p3]);
```




```
> display([p2, p4]);
```

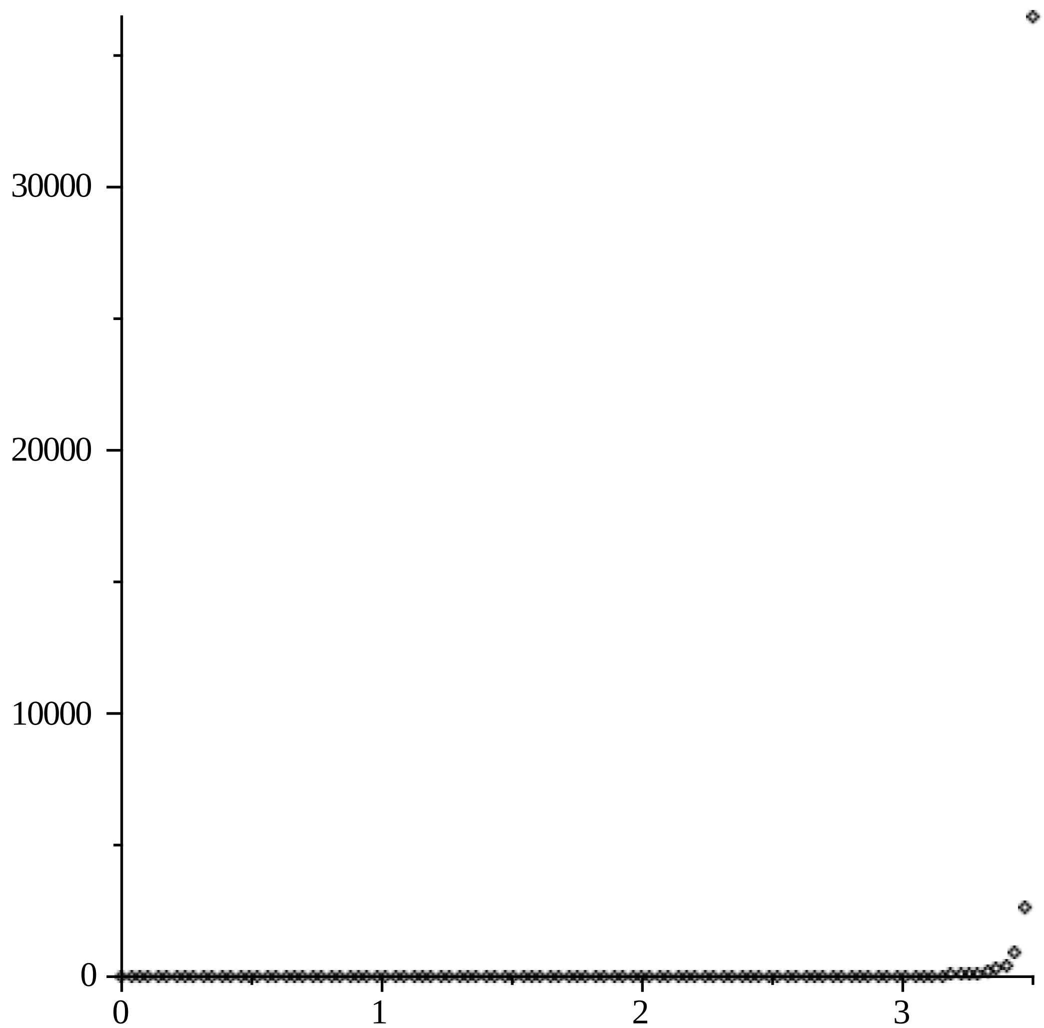


Utilizemos agora o método da série de Taylor, capaz de produzir soluções com grande acuracidade.

> $Digits := 20$:

```
> S_taylor:=dsolve({e1, y(0) = 1, (D(y))(0) = 0}, numeric,
  method = taylorseries, abserr = 1.*10^(-20), range = 0 ..
  3.5):
> L_taylor:=[seq([op(2, S_taylor(i*h)[1]), op(2,S_taylor(i*h)
  [2])],i=0..100)]:
> p3:=pointplot(L_taylor,color=black):
```

> $display(p3)$



>

Portanto, o súbito crescimento próximo a t=3.5 não é uma instabilidade do método numérico.

3. O flutuador cônico mostrado na Fig. 1 é livre para deslizar sobre uma barra vertical. Quando o flutuador é perturbado de sua posição de equilíbrio, ele sofre um movimento oscilatório descrito pela equação diferencial

$$\ddot{y} = g\left(1 - a y^3\right)$$

onde $a = 16\ m^{-3}$ (determinado pela densidade e dimensões do flutuador), e $g = \frac{9.80665\ m}{s^2}$. Se o flutuador é levantado a uma altura de $0.1\ m$ e solto, determine o período e a amplitude das ocilações. Resolva este problema usando (i) uma versão implementada por você mesmo de RK4, (ii) a versão embutida no seu sistema de RKF45. Compare os métodos e faça gráficos.

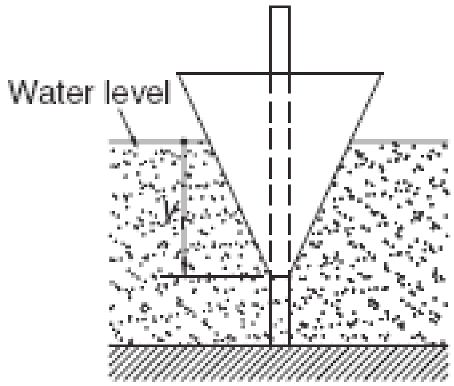


Fig. 1

Solução

```
> restart
> f:=gg*(1-a*y^3);
```

$$f := gg\left(1 - a\,y^3\right) \qquad \textbf{(3.1)}$$

```
> F:=unapply(f,t,y,u);
```

$$F := (t, y, u) \rightarrow gg\left(1 - a\,y^3\right) \qquad \textbf{(3.2)}$$

A posição de equilíbrio ocorre quando f = 0. Nesta situação o valor de y é dado por

$$> \; Y := evalf\left(\left(\frac{1}{16}\right)^{\left(\frac{1}{3}\right)}\right)$$

$$Y := 0.3968502630 \qquad \textbf{(3.3)}$$

Quando o flutuador é erguido em 0.1 m a gravidade passa a ser maior que o empuxo (quando o flutuador está totalmente erguido fora da água temos y = 0 e f = g.). O deslocamento inicial é então Y - 1 e

$$> \; gg := 9.90665;\ h := 0.01;\ a := 16;\ y0 := Y - 0.1;\ u0 := 0;\ t0 := 0;\ nit := 150$$

$$gg := 9.90665$$

$$h := 0.01$$

$$a := 16$$

$$y0 := 0.2968502630$$

$$u0 := 0$$

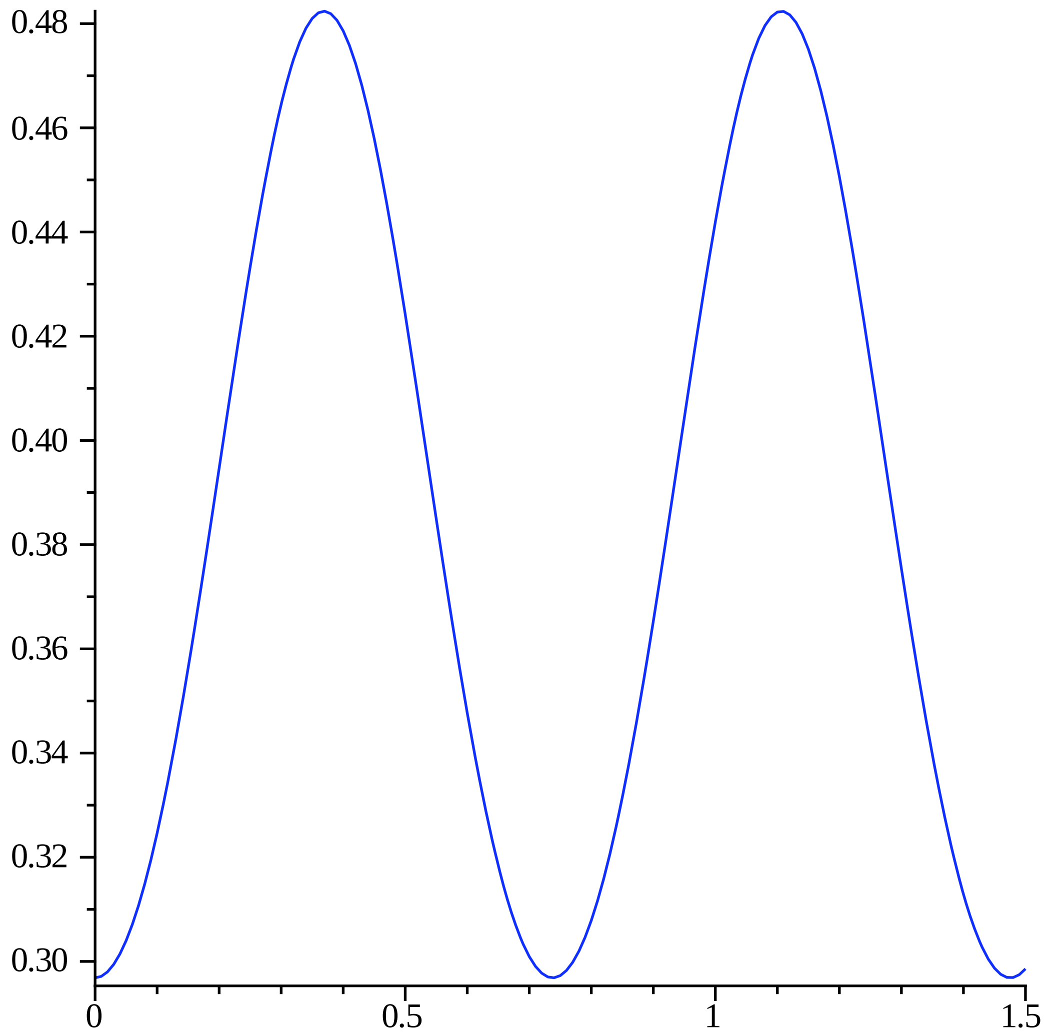
$$t0 := 0$$

$$nit := 150 \qquad \textbf{(3.4)}$$

Usando a implementação clássica de RK4 temos

```
> with(plots):
> drk4:=proc(x0,y0,u0,h,nit,F)
    local K1, K2,K3,K4, L1, L2, L3,L4,x,y,i,LL3,u;
    x[0] := x0; y[0] := y0; u[0] := u0;
    LL3:=[[x[0],y[0]]];
    for i from 1 to nit do
```

```
>       L1:=evalf(u[i-1]);
        K1:=evalf(F(x[i-1],y[i-1],u[i-1]));
>       L2:= evalf(u[i-1]+h*K1/2);
        K2:=evalf(F(x[i-1]+h/2,y[i-1]+h*L1/2,u[i-1]+h*K1/2));
>       L3:=evalf(u[i-1]+h*K2/2);
        K3:=evalf(F(x[i-1]+h/2,y[i-1]+h*L2/2,u[i-1]+h*K2/2));
>       L4:=evalf(u[i-1]+h*K3);
        K4:=evalf(F(x[i-1]+h,y[i-1]+h*L3,u[i-1]+h*K3));
        u[i]:=evalf(u[i-1]+(h/6)*(K1+2*K2+2*K3+K4));
        y[i]:=evalf(y[i-1]+(h/6)*(L1+2*L2+2*L3+L4));
>       x[i]:=evalf(x[0]+i*h);
>       LL3:=[op(LL3),[x[i],y[i]]];
>    od:
  end:
> LL3 := drk4(t0, y0, u0, h, nit, F):
> p4 := pointplot(LL3, color = blue,connect=true):
> display(p4);
```



Utilizaremos agora o comando embutido:

> $e2 := diff\left(y(t), t\$2\right) = gg \cdot \left(1 - a\, y(t)^3\right)$

$$e2 := \frac{d^2}{dt^2}\, y(t) = 9.90665 - 158.50640\, y(t)^3 \qquad \textbf{(3.5)}$$

```
> S2 := dsolve({e2, y(0) = y0, (D(y))(0) = u0}, numeric, method
```

```
= rkf45, initstep = 0.01);
```

$S2 := \textbf{proc}\,(x\_rkf45)\ \ldots\ \textbf{end proc}$ **(3.6)**

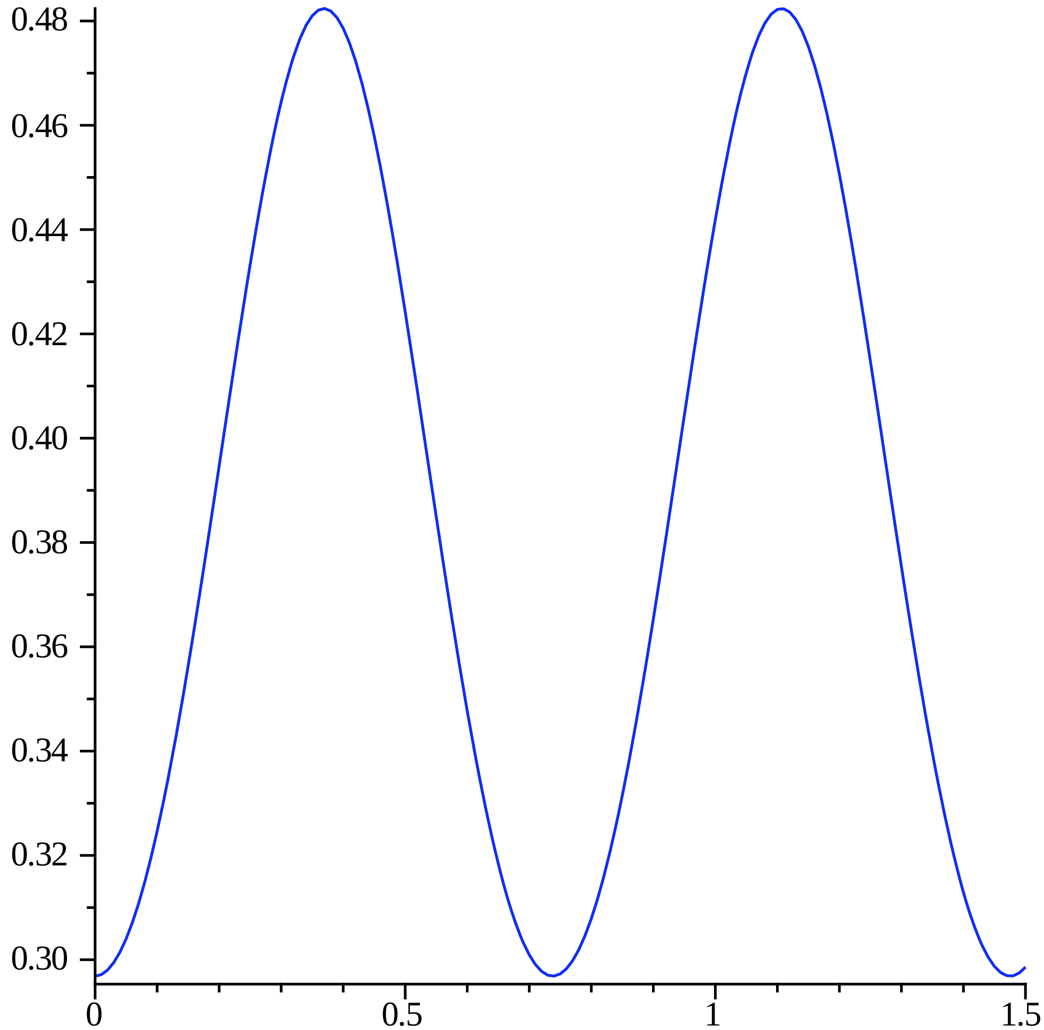
```
> op(2, S2(0)[2]);
```

0.296850263000000 **(3.7)**

```
> LL2 := [seq([op(2, S2(i*h)[1]), op(2, S2(i*h)[2])], i = 0 ..
150)]:
```

> $p5 := pointplot(LL2, connect = true) :$

> $display([p5, p4])$



Ou seja, os dois métodos praticamente coincidem. A amplitude do movimento é aproximadamente

```
> 0.48-y0;
```

0.1831497370 **(3.8)**

A frequencia é dada, em Hz, aproximadamente por

> $f := evalf\left(\frac{2 \cdot \pi}{0.74}\right);$

$f := 8.490790958$ **(3.9)**

>

4. Uma espaçonave é lançada a uma altitude $H = 772\ Km$ acima do nível do mar com uma velocidade $v_o = 6700$ m/s na direção mostrada na Fig. 2. As equações diferenciais descrevendo o movimento da espaçonave são dadas por

$$\ddot{r} = r\dot{\theta}^2 - \frac{GM_e}{r^2} \qquad \ddot{\theta} = -\frac{2\dot{r}\dot{\theta}}{r} \quad , \qquad (2.1)$$

onde r e θ são as coordenadas polares da espaçonave. As constantes envolvidas no movimento são dadas por

$G = 6.672 \times 10^{-11}\ \mathrm{m^3kg^{-1}s^{-2}}$

$M_e = 5.9742 \times 10^{24}\ \mathrm{kg}$ (massa da Terra)

$R_e = 6378.14\ \mathrm{km}$ (raio da Terra no nível do mar)

(a) Deduza fisicamente as eqs. (2.1)

(b) Escreva o sistema de EDOs em termos de um sistema de equações de primeira ordem e condições iniciais na forma

$$\dot{\mathbf{y}} = \mathbf{F}(t, \mathbf{y}),\ \mathbf{y}(0) = \mathbf{b}.$$

(c) Use RK4 e RKF45 para integrar as equações do tempo de lançamento até o instante em que ele atinge a Terra. Determine θ no tempo de impacto.

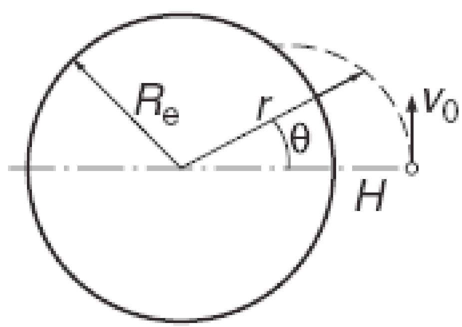


Fig. 2

Solução

> *restart*

> $e1 := diff(r(t), t) = u(t)$

$$e1 := \frac{d}{dt} r(t) = u(t) \qquad \textbf{(4.1)}$$

> $e2 := diff(u(t), t) = r(t) \cdot diff(\theta(t), t)^2 - \frac{G \cdot M}{r(t)^2}$

$$e2 := \frac{d}{dt} u(t) = r(t) \left(\frac{d}{dt} \theta(t)\right)^2 - \frac{G M}{r(t)^2} \qquad \textbf{(4.2)}$$

> $e3 := diff(\theta(t), t) = v(t);$

$$e3 := \frac{d}{dt} \theta(t) = v(t) \qquad \textbf{(4.3)}$$

> $e4 := diff(v(t), t) = -\frac{2 \cdot u(t) \cdot v(t)}{r(t)}$

$$e4 := \frac{d}{dt} v(t) = -\frac{2\, u(t)\, v(t)}{r(t)} \qquad (4.4)$$

> $e2b := subs(diff(\text{theta}(t), t) = v(t), e2)$

$$e2b := \frac{d}{dt} u(t) = r(t)\, v(t)^2 - \frac{G\, M}{r(t)^2} \qquad (4.5)$$

> $H := 772 \cdot 10^3;\ R := 6378.14 \cdot 10^3;\ G := 6.672 \cdot 10^{(-11)};\ M := 5.9742 \cdot 10^{24}$

$$H := 772000$$

$$R := 6.37814000\ 10^6$$

$$G := 6.672000000\ 10^{-11}$$

$$M := 5.974200000\ 10^{24} \qquad (4.6)$$

> $r0 := R + H;\ v0 := \frac{6700}{r0};\ \theta 0 := 0;\ u0 := 0$

$$r0 := 7.15014000\ 10^6$$

$$v0 := 0.0009370445894$$

$$\theta 0 := 0$$

$$u0 := 0 \qquad (4.7)$$

> $dsys := \{e1, e2b, e3, e4, r(0) = r0, \theta(0) = \theta 0, u(0) = u0, v(0) = v0\};$

$$dsys := \left\{ \frac{d}{dt} r(t) = u(t), \frac{d}{dt} \theta(t) = v(t), \frac{d}{dt} u(t) = r(t)\, v(t)^2 - \frac{3.985986240\ 10^{14}}{r(t)^2}, \frac{d}{dt} v(t) = -\frac{2\, u(t)\, v(t)}{r(t)}, r(0) = 7.15014000\ 10^6, \theta(0) = 0, u(0) = 0, v(0) = 0.0009370445894 \right\} \qquad (4.8)$$

>

> $dsn1 := dsolve(dsys, numeric);$

$$dsn1 := \textbf{proc}(x\_rkf45)\ \ldots\ \textbf{end proc} \qquad (4.9)$$

> R

$$6.37814000\ 10^6 \qquad (4.10)$$

> $dsn1(10)$

$$[t = 10., r(t) = 7.15006407892201282\ 10^6, \theta(t) = 0.00937051222568775245, u(t) = -15.1841582337949816, v(t) = 0.000937064489029502696] \qquad (4.11)$$

> $seq(op(2, dsn1(T)[2]), T = 900..1500, 15)$

$$6.55679722348242253\ 10^6, 6.53771276796694566\ 10^6, 6.51837795945649128\ 10^6, 6.49879683448141348\ 10^6, 6.47897356625737529\ 10^6, 6.45891243042481877\ 10^6, 6.43861786601037253\ 10^6, 6.41809447600742709\ 10^6, 6.39734702737612929\ 10^6, \qquad (4.12)$$

$6.37638045104338415\ 10^6$, $6.35519984190286789\ 10^6$, $6.33381045881500281\ 10^6$, $6.31221772460698709\ 10^6$, $6.29042725551312231\ 10^6$, $6.26844484256502241\ 10^6$, $6.24627643898989446\ 10^6$, $6.22392820707257278\ 10^6$, $6.20140651827148814\ 10^6$, $6.17871795321866404\ 10^6$, $6.15586930171972141\ 10^6$, $6.13286756275387667\ 10^6$, $6.10971994711716939\ 10^6$, $6.08643390980370808\ 10^6$, $6.06301710415876284\ 10^6$, $6.03947741904844530\ 10^6$, $6.01582299388784729\ 10^6$, $5.99206221864104271\ 10^6$, $5.96820373382109962\ 10^6$, $5.94425643049007002\ 10^6$, $5.92022945705092326\ 10^6$, $5.89613223598433845\ 10^6$, $5.87197443325497489\ 10^6$, $5.84776599268851429\ 10^6$, $5.82351713967804797\ 10^6$, $5.79923838118409179\ 10^6$, $5.77494050573458429\ 10^6$, $5.75063458701303601\ 10^6$, $5.72633199524374306\ 10^6$, $5.70204437497139536\ 10^6$, $5.67778366243938636\ 10^6$, $5.65356208812715579\ 10^6$

> $dsn1(1)[2]$

$$r(t) = 7.15013924078637734\ 10^6 \quad \textbf{(4.13)}$$

> $RR := t \rightarrow rhs(dsn1(t)[2])$

$$RR := t \rightarrow rhs\left(dsn1(t)_2\right) \quad \textbf{(4.14)}$$

> $tt := 900$

$$tt := 900 \quad \textbf{(4.15)}$$

> R

$$6.37814000\ 10^6 \quad \textbf{(4.16)}$$

```
> while RR(tt) > R do
    tt := tt + 15
  od:
```

> $TT := tt - 15;$

$$TT := 1020 \quad \textbf{(4.17)}$$

> $RR(\mathrm{TT})$

$$6.39734702737612929\ 10^6 \quad \textbf{(4.18)}$$

> $dsn1(TT)$

$$[t = 1020., r(t) = 6.39734702737612929\ 10^6, \theta(t) = 1.03158190633534486, u(t) = -1390.52226868961566, v(t) = 0.00117054865179889458] \quad \textbf{(4.19)}$$

> $tt := TT$

$$tt := 1020 \quad \textbf{(4.20)}$$

Refinamento:

```
> while RR(tt) > R do
    tt := tt + 0.006
  od:
```

> tt

$$1033.752 \quad \textbf{(4.21)}$$

Portanto, a partícula atinge a superfície em tt 1047.11 s, ou, em minutos,

> $\frac{tt}{60}$

$$17.22920000 \quad (4.22)$$

Neste instante, os parâmetros do movimento são dados por

> $dsn1(tt)$

$$\left[t = 1033.752,\ r(t) = 6.37813310440798569\ 10^6,\ \theta(t) = 1.04772772432230488,\ u(t) = -1403.77431316731463,\ v(t) = 0.00117761179162122458\right] \quad (4.23)$$

Portanto, o ângulo em radianos é

> $dsn1(tt)[3]$

$$\theta(t) = 1.04772772432230488 \quad (4.24)$$

ou, em graus,

> $evalf\left(\frac{rhs(\%)\cdot 180}{\pi}\right)$

$$60.03037664 \quad (4.25)$$

>

5. Resolva o problema de valor de contorno usando diferenças finitas:

$$\frac{d^2}{dx^2}y(x) + \frac{d}{dx}y(x) + y(x) = \sin(x),\quad y(0) = 1,\ y(2) = 1.$$

Solução

Esta é uma EDO de Sturm-Liouville, que tem a forma geral

```
> restart:
> with(plots):
> edo:= diff(y(x),`$`(x,2))+p(x)*diff(y(x),x)+q(x)*y(x) = r(x);
```

$$edo := \frac{d^2}{dx^2}y(x) + p(x)\left(\frac{d}{dx}y(x)\right) + q(x)\,y(x) = r(x) \quad (5.1)$$

Vamos considerar agora o problema particular $p = -1,\quad q = x1,\quad r = \sin x$

```
> p:=x->1;
```

$$p := x \rightarrow 1 \quad (5.2)$$

```
> q:=x->1;
```

$$q := x \rightarrow 1 \quad (5.3)$$

```
> r:=x->sin(x);
```

$$r := x \rightarrow \sin(x) \quad (5.4)$$

de modo que a EDO toma a forma

```
> edo;
```

$$\frac{d^2}{dx^2}y(x) + \frac{d}{dx}y(x) + y(x) = \sin(x) \quad (5.5)$$

Uma solução exata que satisfaz este problema de valor de contorno com y(0) = 1, y(2) =1 é dada por

> $S1 := dsolve(\mathrm{edo}, y(x))$

$$S1 := y(x) = e^{-\frac{1}{2}x} \sin\left(\frac{1}{2}\sqrt{3}\,x\right) \_C2 + e^{-\frac{1}{2}x} \cos\left(\frac{1}{2}\sqrt{3}\,x\right) \_C1 - \cos(x) \quad \textbf{(5.6)}$$

> $eq1 := evalf(subs(x=0, rhs(S1))) = 1$

$$eq1 := -1. + 1.\,\_C1 = 1 \quad \textbf{(5.7)}$$

> $eq2 := evalf(subs(x=2, rhs(S1))) = 1$

$$eq2 := 0.3631068106\,\_C2 - 0.05906544985\,\_C1 + 0.4161468365 = 1 \quad \textbf{(5.8)}$$

> $ss := solve(\{eq1, eq2\})$

$$ss := \{\_C1 = 2., \_C2 = 1.933271541\} \quad \textbf{(5.9)}$$

> $assign(ss)$

> $yex := evalf(rhs(S1))$

$$yex := 1.933271541\, e^{-0.5000000000\,x} \sin(0.8660254040\,x) + 2.\, e^{-0.5000000000\,x} \cos(0.8660254040\,x) - 1.\cos(x) \quad \textbf{(5.10)}$$

> $p1 := plot(yex, x = 0..2):$

Vamos comparar o resultado exato obtido acima com o numérico dado pelo sistema:

> $sol1 := dsolve(\{\mathrm{edo}, y(0) = 1, y(2) = 1\}, numeric)$

$$sol1 := \textbf{proc}\,(x\_bvp)\ \ldots\ \textbf{end proc} \quad \textbf{(5.11)}$$

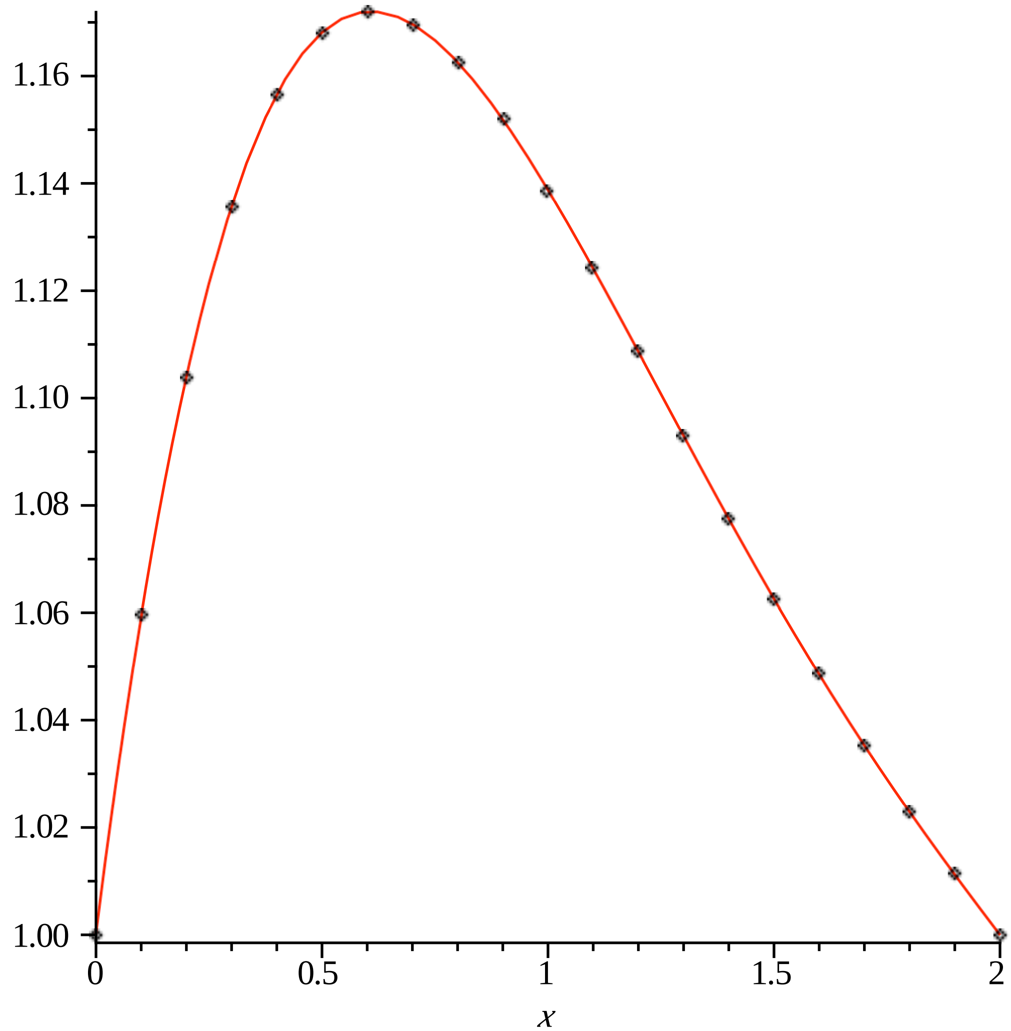
> $h1 := 0.1$

$$h1 := 0.1 \quad \textbf{(5.12)}$$

> $LL1 := [seq([op(2, sol1(i * h1)[1]), op(2, sol1(i * h1)[2])], i = 0..20)]:$

> $p2 := pointplot(LL1):$

> $display([p1, p2])$



Façamos agora uma implementação do algoritmo de diferenças finitas para aplicá-lo neste mesmo problema:

```
> h:=0.05;
```

$$h := 0.05 \tag{5.13}$$

```
> x0:=0; xn:=2;
```

$$x0 := 0$$
$$xn := 2 \tag{5.14}$$

```
> n:=trunc((xn-x0)/h);
```

$$n := 40 \tag{5.15}$$

```
> y[0]:=1; y[n]:=1;
```

$$y_0 := 1$$
$$y_{40} := 1 \tag{5.16}$$

```
> x[0]:=x0;x[n]:=xn;
```

$$x_0 := 0$$
$$x_{40} := 2 \tag{5.17}$$

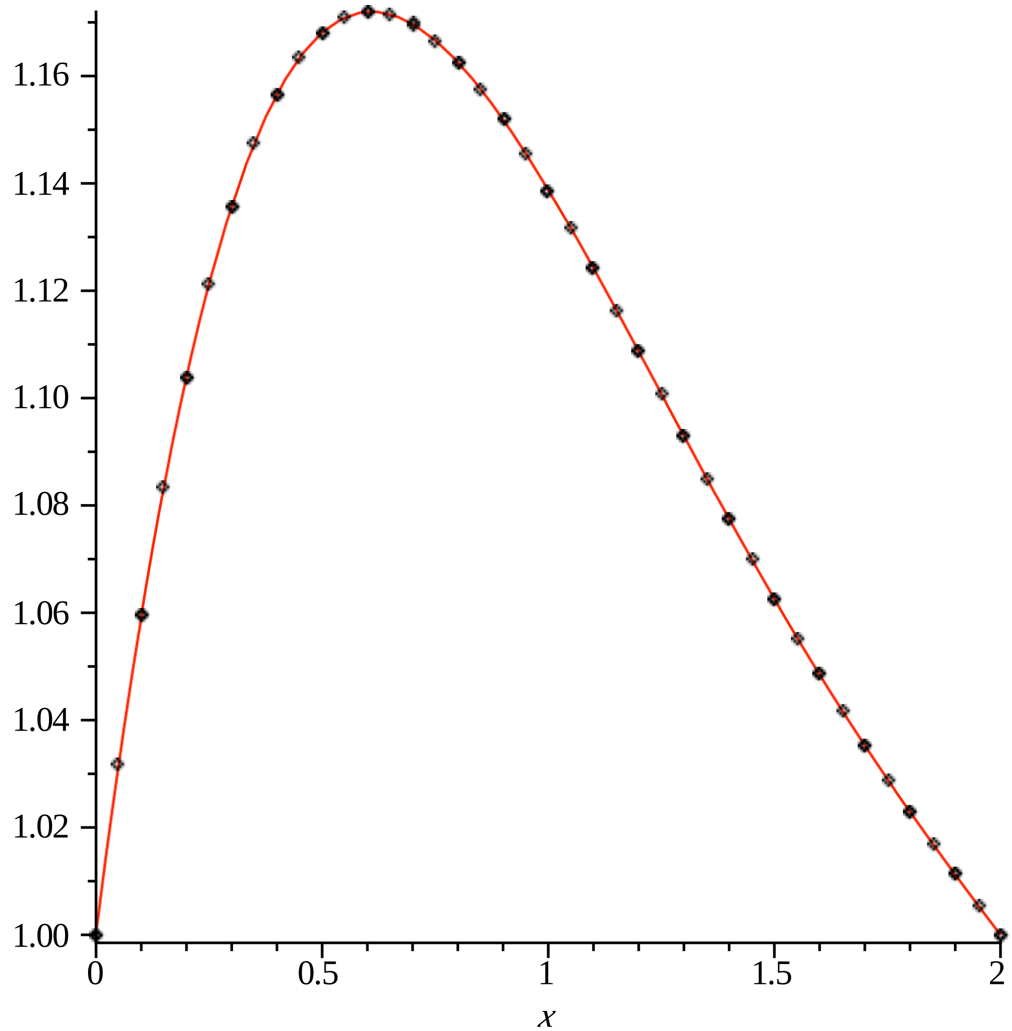
```
> for i to n-1 do
     x[i]:=x[i-1]+h;
```

```
    eq[i]:=(1-h*p(x[i])/2)*y[i-1]+
    (-2+q(x[i])*h^2)*y[i]+(1+h*p(x[i])/2)*y[i+1] = h^2*r(x[i])
  ;
  od:
> sol:=solve({seq(eq[i],i=1..n-1)},{seq(y[i],i=1..n-1)}):
> assign(sol);
> LL:=[seq([x[k],y[k]],k=0..n)];
```

LL := [[0, 1], [0.05, 1.031677874], [0.10, 1.059416099], [0.15, 1.083460793], [0.20, 1.104054472], [0.25, 1.121435325], [0.30, 1.135836548], [0.35, 1.147485720], [0.40, 1.156604229], [0.45, 1.163406749], [0.50, 1.168100752], [0.55, 1.170886084], [0.60, 1.171954574], [0.65, 1.171489693], [0.70, 1.169666261], [0.75, 1.166650195], [0.80, 1.162598300], [0.85, 1.157658102], [0.90, 1.151967724], [0.95, 1.145655801], [1.00, 1.138841435], [1.05, 1.131634183], [1.10, 1.124134088], [1.15, 1.116431736], [1.20, 1.108608359], [1.25, 1.100735953], [1.30, 1.092877441], [1.35, 1.085086859], [1.40, 1.077409565], [1.45, 1.069882481], [1.50, 1.062534353], [1.55, 1.055386038], [1.60, 1.048450806], [1.65, 1.041734665], [1.70, 1.035236703], [1.75, 1.028949442], [1.80, 1.022859209], [1.85, 1.016946520], [1.90, 1.011186472], [1.95, 1.005549142], [2, 1]] **(5.18)**

Comparemos com a solução exata (linha contínua) e com a solução numérica fornecida pelo sistema:

```
> p3:=pointplot(LL):
> display([p1,p2,p3]);
```



Note que existe boa concordância entre os resultados numéricos fornecidos pelo sistema e aquele resultante do programa implementado.

> $F := unapply(yex, x)$

$$F := x \rightarrow 1.933271541\, e^{-0.5000000000\, x} \sin(0.8660254040\, x) + 2.\, e^{-0.5000000000\, x} \cos(0.8660254040\, x) - 1.\, \cos(x) \quad \textbf{(5.19)}$$

O erro deste último resultado é dado por

```
> erro:=[seq(abs(evalf((y[j]-F(x[j]))*100/F(x[j]))),j=1..n-1)];
```

erro := [0.001552155857, 0.002777644910, 0.003730684873, 0.004454417042, **(5.20)**
0.004982880692, 0.005344276342, 0.005561860018, 0.005654803657,
0.005640391988, 0.005533048052, 0.005345895464, 0.005090473474,
0.004777478970, 0.004416412016, 0.004016875200, 0.003587180195,
0.003135825532, 0.002670285004, 0.002198177982, 0.001726609069,
0.001262526406, 0.0008124541635, 0.0003824699884, 0.00002137814923,
0.0003938258686, 0.0007296281834, 0.001024331467, 0.001273686967,
0.001474252336, 0.001622885895, 0.001717162076, 0.001755225603,
0.001735728564, 0.001657950822, 0.001521820321, 0.001327828848,

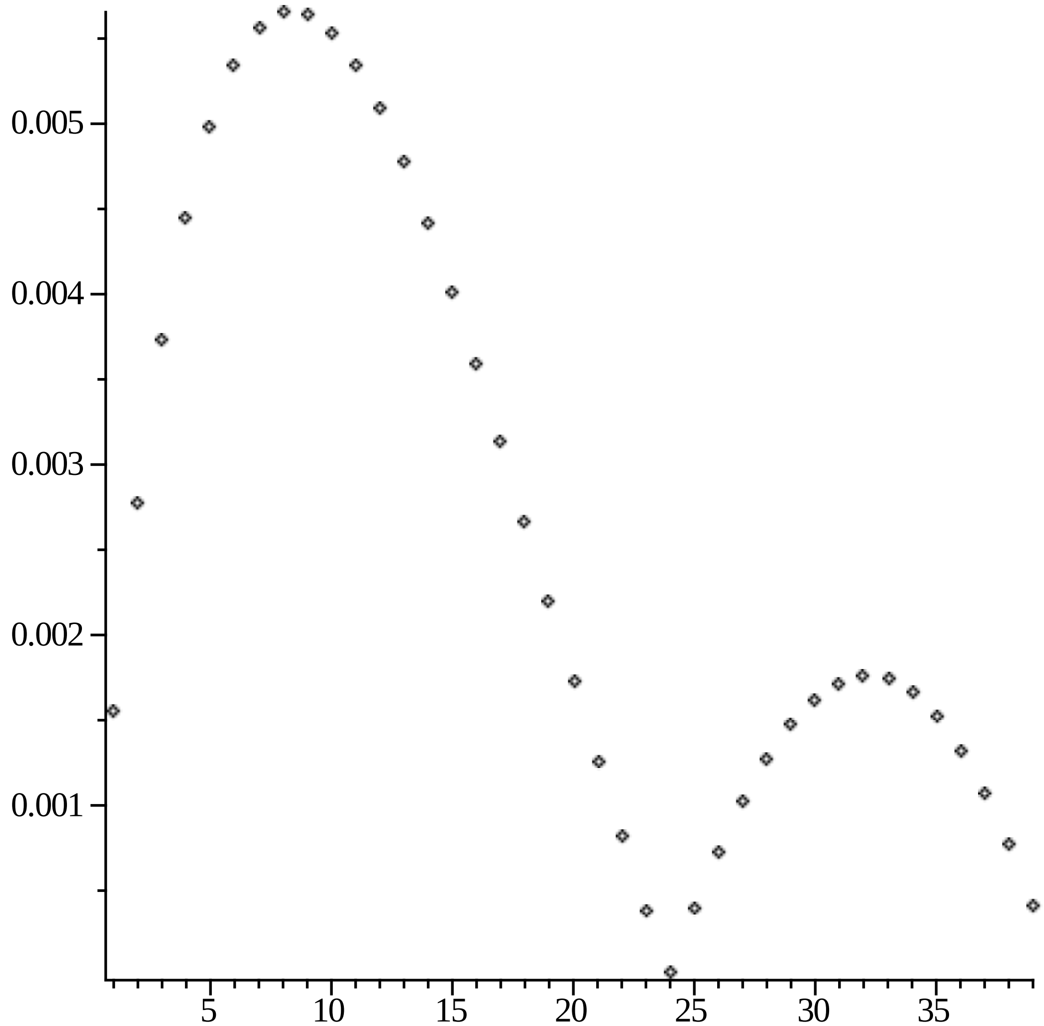
0.001077036188, 0.0007706728835, 0.0004108186121]

Façamos um gráfico do erro percentual:

```
> LL2:=[seq([i,erro[i]],i=1..n-1)];
```

LL2 := [[1, 0.001552155857], [2, 0.002777644910], [3, 0.003730684873], [4, 0.004454417042], [5, 0.004982880692], [6, 0.005344276342], [7, 0.005561860018], [8, 0.005654803657], [9, 0.005640391988], [10, 0.005533048052], [11, 0.005345895464], [12, 0.005090473474], [13, 0.004777478970], [14, 0.004416412016], [15, 0.004016875200], [16, 0.003587180195], [17, 0.003135825532], [18, 0.002670285004], [19, 0.002198177982], [20, 0.001726609069], [21, 0.001262526406], [22, 0.0008124541635], [23, 0.0003824699884], [24, 0.00002137814923], [25, 0.0003938258686], [26, 0.0007296281834], [27, 0.001024331467], [28, 0.001273686967], [29, 0.001474252336], [30, 0.001622885895], [31, 0.001717162076], [32, 0.001755225603], [33, 0.001735728564], [34, 0.001657950822], [35, 0.001521820321], [36, 0.001327828848], [37, 0.001077036188], [38, 0.0007706728835], [39, 0.0004108186121]] **(5.21)**

> *pointplot* (*LL2*)



6. Resolva o problema de valor de contorno:

$$y'' = -3\,y\,y', \quad y(0) = 0, \quad y'(2) = 1$$

Solução

Temos agora um problema não-linear, de modo que podemos esperar maiores dificuldades com o método de diferenças finitas, pois o sistema a ser resolvido será agora não-linear.

```
> restart
> edo := diff(y(x), `$`(x, 2)) =-3·diff(y(x), x)·y(x)
```

$$edo := \frac{d^2}{dx^2}\,y(x) = -3\left(\frac{d}{dx}\,y(x)\right)y(x) \tag{6.1}$$

```
> F := subs({y(x) = y, diff(y(x), x) = u}, rhs(edo))
```

$$F := -3\,u\,y \tag{6.2}$$

```
> f := unapply(F, x, y, u)
```

$$f := (x, y, u) \rightarrow -3\,u\,y \tag{6.3}$$

```
> x0 := 0.; xf := 2.; y0 := 0; yf := 1.
```

$$x0 := 0.$$
$$xf := 2.$$
$$y0 := 0$$
$$yf := 1. \quad (6.4)$$

> $N := 40$

$$N := 40 \quad (6.5)$$

> $h := \frac{(xf - x0)}{N}$

$$h := 0.05000000000 \quad (6.6)$$

> $eq := i \rightarrow Y[i+1] - 2 \cdot Y[i] + Y[i-1] = h^2 \cdot f\left(x[i], Y[i], \frac{(Y[i+1] - Y[i-1])}{2 \cdot h}\right)$

$$eq := i \rightarrow Y_{i+1} - 2\, Y_i + Y_{i-1} = h^2\, f\left(x_i, Y_i, \frac{1}{2}\, \frac{Y_{i+1} - Y_{i-1}}{h}\right) \quad (6.7)$$

> $Y[0] := y0$

$$Y_0 := 0 \quad (6.8)$$

> $Y[N] := yf$

$$Y_{40} := 1. \quad (6.9)$$

> $x[0] := x0;$

$$x_0 := 0. \quad (6.10)$$

> **for** i **to** $N - 1$ **do**
> $x[i] := x[i-1] + h:$
> **od**:

>

> $vars := [\, seq(Y[i], i = 1..N-1)\,]$

$$vars := [Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7, Y_8, Y_9, Y_{10}, Y_{11}, Y_{12}, Y_{13}, Y_{14}, Y_{15}, Y_{16}, Y_{17}, Y_{18}, Y_{19}, Y_{20}, Y_{21}, Y_{22}, Y_{23}, Y_{24}, Y_{25}, Y_{26}, Y_{27}, Y_{28}, Y_{29}, Y_{30}, Y_{31}, Y_{32}, Y_{33}, Y_{34}, Y_{35}, Y_{36}, Y_{37}, Y_{38}, Y_{39}] \quad (6.11)$$

> $eqns := [\, seq(eq(k), k = 1..N-1)\,]:$

> $S1 := fsolve(eqns);$

$$S1 := \{Y_1 = 0.07571781434, Y_2 = 0.1505805067, Y_3 = 0.2237711530, Y_4 = 0.2945456558, Y_5 = 0.3622607944, Y_6 = 0.4263936802, Y_7 = 0.4865517987, Y_8 = 0.5424739860, Y_9 = 0.5940236231, Y_{10} = 0.6411759145, Y_{11} = 0.6840013400, Y_{12} = 0.7226472764, Y_{13} = 0.7573194765, Y_{14} = 0.7882646764, Y_{15} = 0.8157551671, Y_{16} = 0.8400757701, Y_{17} = 0.8615133478, Y_{18} = 0.8803487474, Y_{19} = 0.8968509415, Y_{20} = 0.9112730490, Y_{21} = 0.9238498997, Y_{22} = 0.9347968180, Y_{23} = 0.9443093306, Y_{24} = 0.9525635465, Y_{25} = 0.9597170035, Y_{26} = 0.9659098125, Y_{27} = 0.9712659765, Y_{28} = 0.9758947857, Y_{29} = 0.9798922218, Y_{30} = 0.9833423241, Y_{31} = 0.9863184851, Y_{32} = 0.9888846581, Y_{33} \quad (6.12)$$

$= 0.9910964657, Y_{34} = 0.9930022067, Y_{35} = 0.9946437623, Y_{36} = 0.9960574052, Y_{37} = 0.9972745193, Y_{38} = 0.9983222342, Y_{39} = 0.9992239845\}$

> assign(S1)

> with(plots):

> L := [seq([i·h, Y[i]], i = 0..N)]

$L := [[0., 0], [0.05000000000, 0.07571781434], [0.1000000000, 0.1505805067], [0.1500000000, 0.2237711530], [0.2000000000, 0.2945456558], [0.2500000000, 0.3622607944], [0.3000000000, 0.4263936802], [0.3500000000, 0.4865517987], [0.4000000000, 0.5424739860], [0.4500000000, 0.5940236231], [0.5000000000, 0.6411759145], [0.5500000000, 0.6840013400], [0.6000000000, 0.7226472764], [0.6500000000, 0.7573194765], [0.7000000000, 0.7882646764], [0.7500000000, 0.8157551671], [0.8000000000, 0.8400757701], [0.8500000000, 0.8615133478], [0.9000000000, 0.8803487474], [0.9500000000, 0.8968509415], [1.000000000, 0.9112730490], [1.050000000, 0.9238498997], [1.100000000, 0.9347968180], [1.150000000, 0.9443093306], [1.200000000, 0.9525635465], [1.250000000, 0.9597170035], [1.300000000, 0.9659098125], [1.350000000, 0.9712659765], [1.400000000, 0.9758947857], [1.450000000, 0.9798922218], [1.500000000, 0.9833423241], [1.550000000, 0.9863184851], [1.600000000, 0.9888846581], [1.650000000, 0.9910964657], [1.700000000, 0.9930022067], [1.750000000, 0.9946437623], [1.800000000, 0.9960574052], [1.850000000, 0.9972745193], [1.900000000, 0.9983222342], [1.950000000, 0.9992239845], [2.000000000, 1.]]$ **(6.13)**

> p0 := pointplot(L)

$$p0 := PLOT(...) \qquad (6.14)$$

A solução exata é dada por

> S1 := dsolve(edo, y(x))

$$S1 := y(x) = \frac{1}{3}\,\frac{\tanh\left(\frac{1}{2}\,\frac{(x+\_C3)\sqrt{6}}{\_C1}\right)\sqrt{6}}{\_C1} \qquad (6.15)$$

Para determinar as constantes aplicamos as condições iniciais:

> eq1 := subs(x = 0, rhs(S1)) = 0

$$eq1 := \frac{1}{3}\,\frac{\tanh\left(\frac{1}{2}\,\frac{\_C2\sqrt{6}}{\_C1}\right)\sqrt{6}}{\_C1} = 0 \qquad (6.16)$$

> eq2 := subs(x = 2, rhs(S1)) = 1

$$eq2 := \frac{1}{3}\,\frac{\tanh\left(\frac{1}{2}\,\frac{(2+\_C2)\sqrt{6}}{\_C1}\right)\sqrt{6}}{\_C1} = 1 \qquad (6.17)$$

```
> gg:=fsolve({eq1,eq2},{_C1,_C2});
```

$$gg := \{\_C1 = 0.8125738061, \_C2 = 0.\} \qquad (6.18)$$

> $assign(gg)$

> $yex := rhs(S1)$

$$yex := 0.4102191467\tanh\left(0.6153287200\,x\sqrt{6}\right)\sqrt{6} \quad \textbf{(6.19)}$$

> $yex := unapply(yex, x)$

$$yex := x \rightarrow 0.4102191467\tanh\left(0.6153287200\,x\sqrt{6}\right)\sqrt{6} \quad \textbf{(6.20)}$$

> $p1 := plot(rhs(S1), x = 0..2):$

A solução numérica fornecida pelo sistema é:

> $sol1 := dsolve(\{edo, y(0) = 0, y(2) = 1\}, numeric)$

$$sol1 := \textbf{proc}\,(x\_bvp)\ \ldots\ \textbf{end proc} \quad \textbf{(6.21)}$$

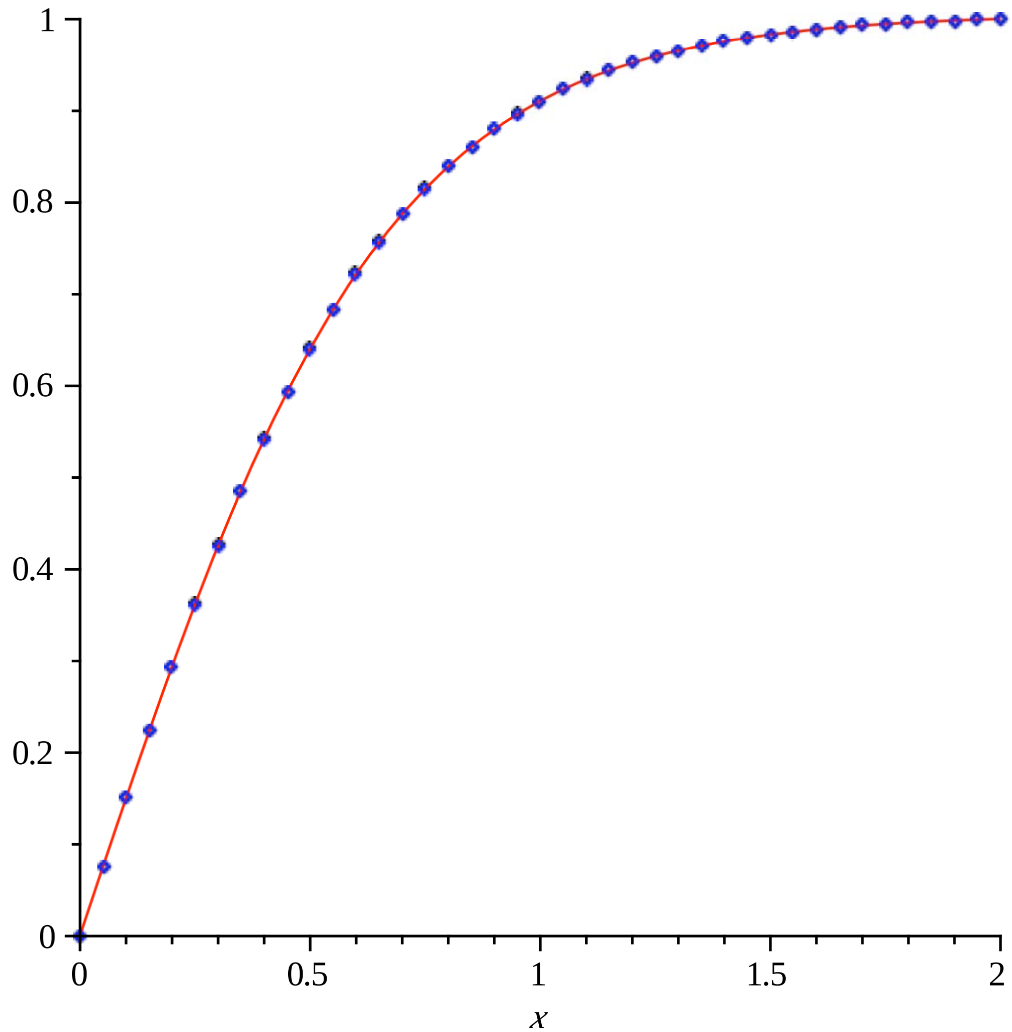
> $LL1 := [seq([op(2, sol1(i*h)[1]), op(2, sol1(i*h)[2])], i = 0..N)]:$

> $p2 := pointplot(LL1, color = blue)$

$$p2 := PLOT(\ldots) \quad \textbf{(6.22)}$$

Comparemos os três resultados:

> $display([p0, p1, p2])$



Os erros percentuais cometidos pelos dois métodos são:

> $erro1 := [seq(\text{abs}(evalf((Y[j] - yex(x[j]))*100/yex(x[j]))), j = 1..N-1)]:$

> $erro2 := [seq(\text{abs}(evalf((Y[j] - op(2, sol1(j*h)[2]))*100/op(2, sol1(j*h)[2]))), j = 1..N-1)]:$

> $LL3 := [seq([i, erro1[i]], i = 1..N-1)]:$

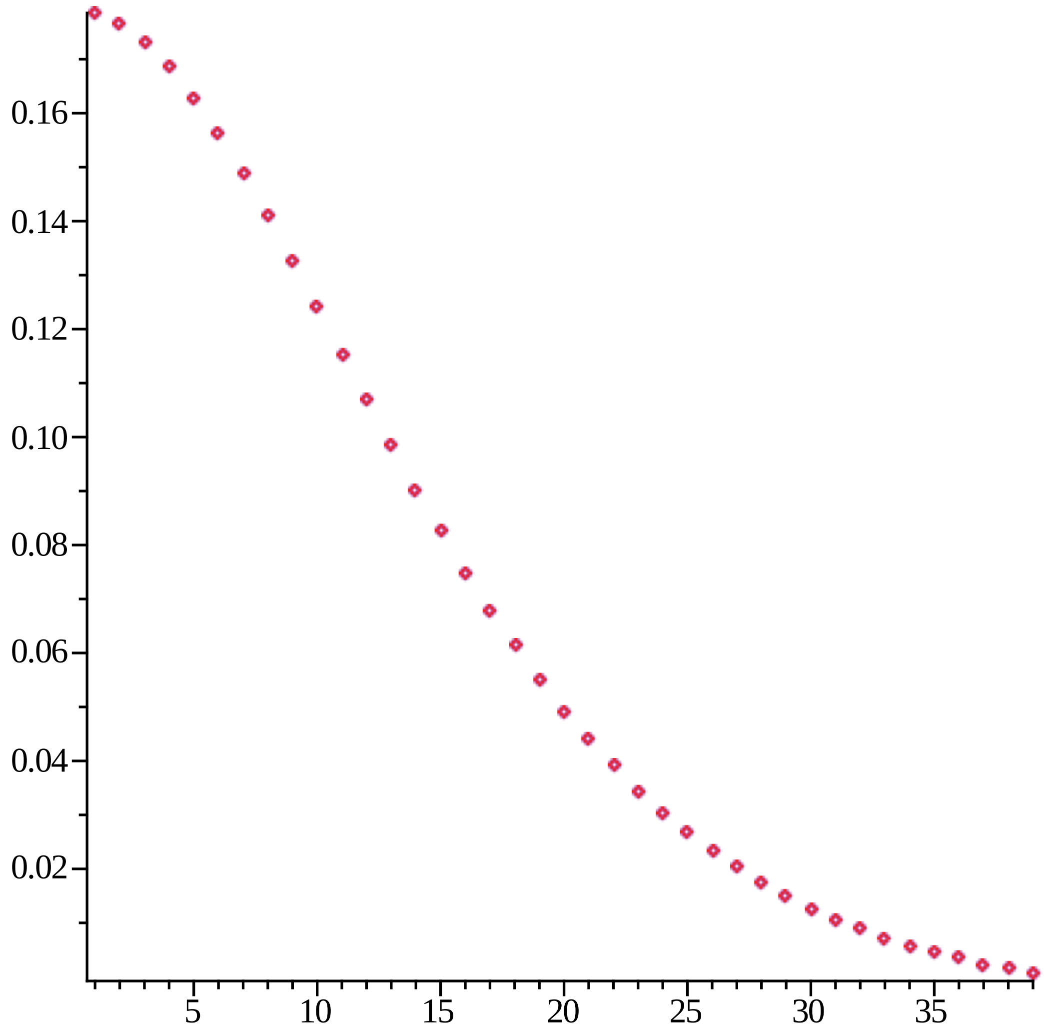
> $LL4 := [seq([i, erro2[i]], i = 1..N-1)]:$

> $p3 := pointplot(LL3, color = blue); p4 := pointplot(LL4, color = red)$

$$p3 := PLOT(...)$$

$$p4 := PLOT(...) \quad \textbf{(6.23)}$$

> $display([p3, p4])$



Ou seja, o erro produzido pelo código implementado é praticamente igual àquele cometido pelo método embutido do sistema, embora o primeiro seja mais lento. Note que o erro diminui rapidamente à medida que x aumenta.