Diferenças finitas, Euler e Runge - Kutta

J. A. Salvador

Introdução

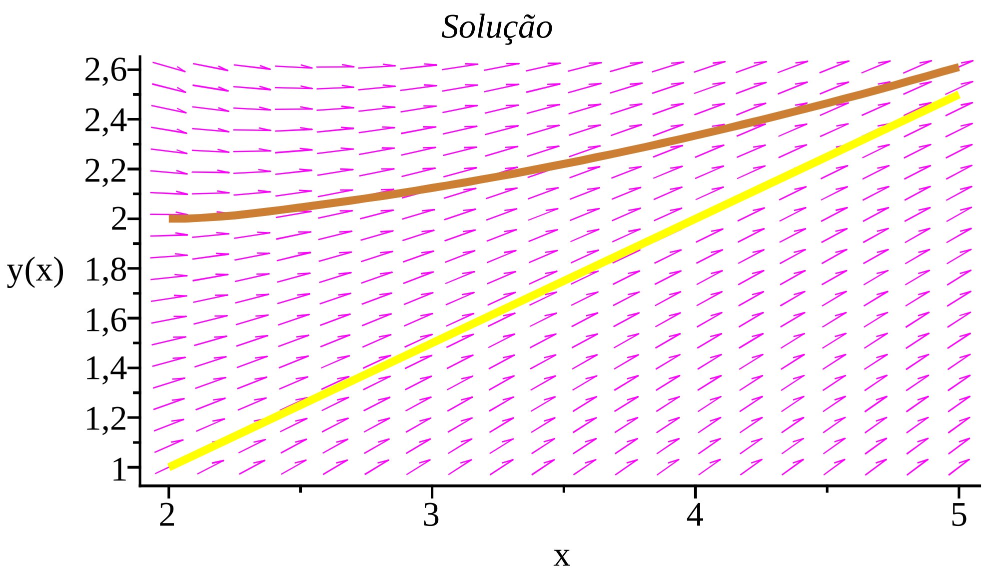
Muitas vezes os problemas reais são modelados por equações diferenciais com condições iniciais ou de contorno. Algumas vezes, a teoria nos garante a existência e unicidade da solução do problema e podemos encontrar soluções analíticas, entretanto, na maioria das vezes isto não é possível. Neste caso, devemos resolver numericamente.
Lembramos por exemplo que, dar uma equação diferencial ordinária de primeira ordem é dar um campo de vetores. Vejamos a eq1;

```
> restart; with(DEtools):
> eq1 := x*diff(y(x),x) = x - y(x);
```

$$eq1 := x\left(\frac{d}{dx}\,y(x)\right) = x - y(x) \qquad (1)$$

Dar a equação diferencial eq1 é dar o campo de vetores abaixo.

```
> phaseportrait( x*diff(y(x),x) = x - y(x), y(x), x=2..5, [[y(2)
=2], [y(2)=1]], title=`Solução`, colour=magenta, linecolor=
[gold,yellow]);
```



Observe as soluções, uma passando pelo ponto $[2, 2]$ e outra pelo ponto $[2, 1]$, como elas

tangenciam os campos vetoriais dados.
Analiticamente o problema de valor inicial constituído da eq1 e da condição inicial $y(2) = 2$ tem a solução analítica

```
> dsolve( {x*diff(y(x),x) = x - y(x), y(2) = 2}, y(x));
```

$$y(x) = \frac{1}{2}x + \frac{2}{x} \tag{2}$$

A resposta dada pelo Maple V é uma expressão e podemos transformá-la numa função $y = y(x)$.

```
> y := unapply( rhs(%), x): 'y(x)'= y(x);
```

$$y(x) = \frac{1}{2}x + \frac{2}{x} \tag{3}$$

E assim, calcular o seu valor nos diversos pontos de interesse:

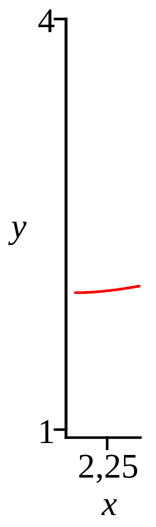
```
> 'y(2.1)' = y(2.1) ; 'y(2.3)' = y(2.3) ;
```

$$y(2.1) = 2.002380952$$

$$y(2.3) = 2.019565217 \tag{4}$$

e fazer o gráfico da solução

```
> plot(y(x), x=2..2.5, y=1..4, scaling=constrained, tickmarks=
  [1,2]);
```



Considere o problema de valor inicial constituído da equação diferencial $\frac{d}{dx}y(x) = f(x, y(x))$ e da condição inicial $y(x_0) = y_0$. Queremos saber o valor da solução do pvi num ponto x.
Como conhecemos o valor no ponto x_0, vamos escrever uma partição $x_0, x_1, x_2, \ldots, x_n$ e se dermos um passo

```
> h := x[i+1] - x[i];
```

$$h := x_{i+1} - x_i \tag{5}$$

podemos tentar calcular o valor da solução $y(x)$ num ponto x_i; $y(x_i)$
Se para calcular o valor de $y(x_i)$ usamos apenas o valor da solução num ponto anterior $y(x_{i-1})$ temos um método de passo simples ou de de passo 1.
No caso de um método de passo simples, $y(x_0) = y_0$ é uma aproximação inicial. Ele é autoinciante, e se para calcular o valor de $y(x_i)$ usamos mais valores da solução em outros pontos, temos um método de passo múltiplo.

Neste caso de método de passo múltiplo temos que buscar alguma estratégia para obter as aproximações iniciais.

Diferenças finitas e Euler

Vamos explorar os métodos numéricos para resolução de equações diferenciais ordinárias.
Tomemos como exemplo o movimento de uma bolinha de massa m lançada verticalmente para cima com velocidade inicial v_0. O movimento será ao longo de um eixo x vertical orientado para cima.
Em t = 0 a bolinha está em x = 0, com velocidade $v = v_0$. A aceleração da bolinha será constante e igual a $-g$.
Nosso problema é determinar a velocidade e a posição da bolinha como função do tempo em qualquer instante *t*.
Sabemos que a aceleração é a derivada da função velocidade em relação ao tempo, e se apenas a aceleração da gravidade está atuando numa partícula de massa unitária, temos

```
> diff(v(t), t) = -g;
```

$$\frac{d}{dt} v(t) = -g \tag{6}$$

cuja solução analítica do problema com a condição inicial $v(0) = v_0$ é

```
> dsolve( {diff(v(t), t) = -g, v(0) = v[0]}, v(t));
```

$$v(t) = -g\, t + v_0 \tag{7}$$

Pela definição de derivada, a equação acima é equivalente, no limite em que $h \to 0$, da seguinte equação:

```
> ((v(t+h)-v(t))/h) = -g;
```

$$\frac{v\left(t + x_{i+1} - x_i\right) - v(t)}{x_{i+1} - x_i} = -g \tag{8}$$

Para h finito esta equação é uma Equação de Diferenças Finitas. Rearranjando-a, obtemos:

```
> v(t+h) = v(t) - g * h;
```

$$v\left(t + x_{i+1} - x_i\right) = v(t) - g\left(x_{i+1} - x_i\right) \tag{9}$$

Esta equação nos permite calcular a velocidade da partícula num instante $t + h$, se conhecermos v (t). Aplicando-a sucessivamente aos instantes t = 0, h, 2h, 3h, ... poderemos determinar a velocidade em função do tempo para um tempo arbitrário.

```
> diff(x(t), t) = v(t);
```

$$\frac{d}{dt} x(t) = v(t) \tag{10}$$

Pela definição de derivada, a equação acima é equivalente, no limite em que $h \to 0$, da seguinte equação:

```
> ( x(t+h)-x(t))/h = v(t);
```

$$\frac{x\left(t + x_{i+1} - x_i\right) - x(t)}{x_{i+1} - x_i} = v(t) \tag{11}$$

Para *h* finito esta equação se chama Equação de Diferenças Finitas. Rearranjando-a, obtemos:

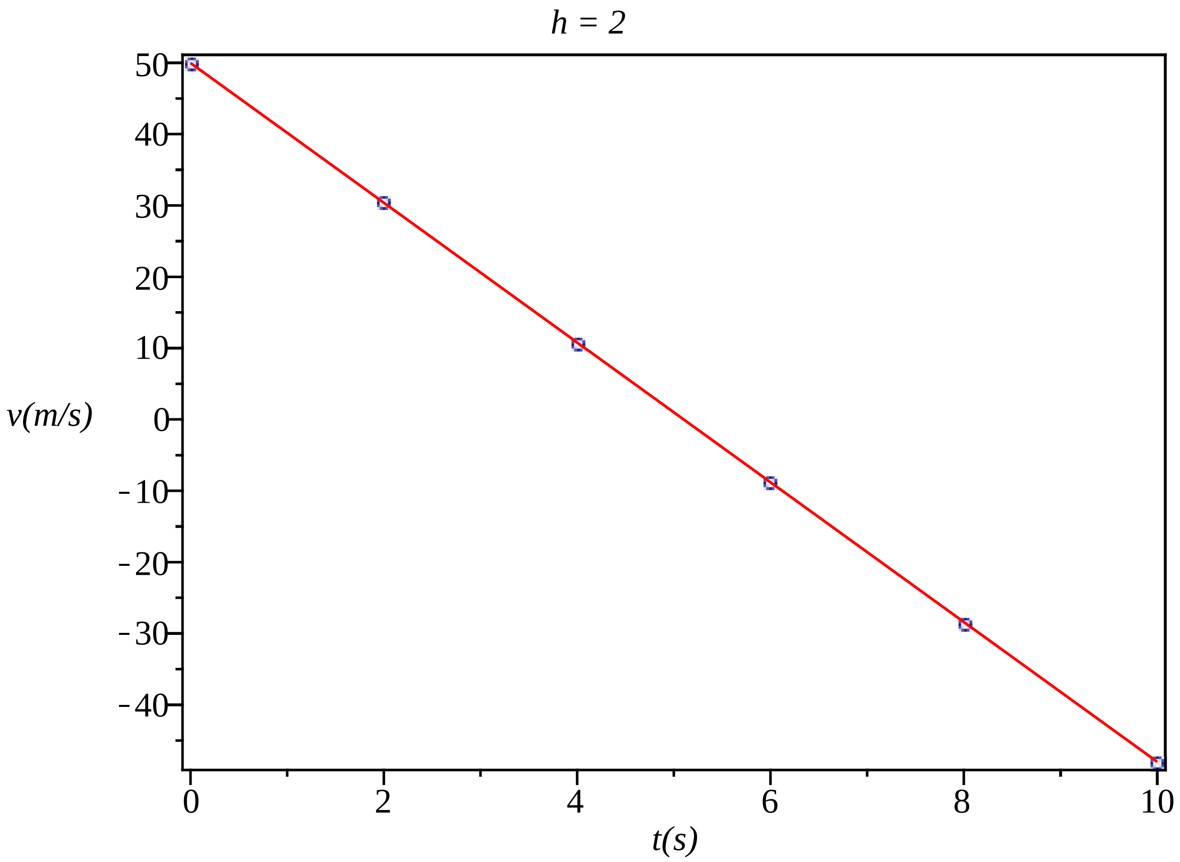
```
> x(t+h) = x(t) + v(t)*h;
```

$$x\left(t + x_{i+1} - x_i\right) = x(t) + v(t)\left(x_{i+1} - x_i\right) \tag{12}$$

Esta equação nos permite calcular a posição no instante $t+h$, se conhecermos $x(t)$. Aplicando-a sucessivamente aos instantes t = 0, h, 2h, 3h, ... poderemos determinar a posição em função do tempo:

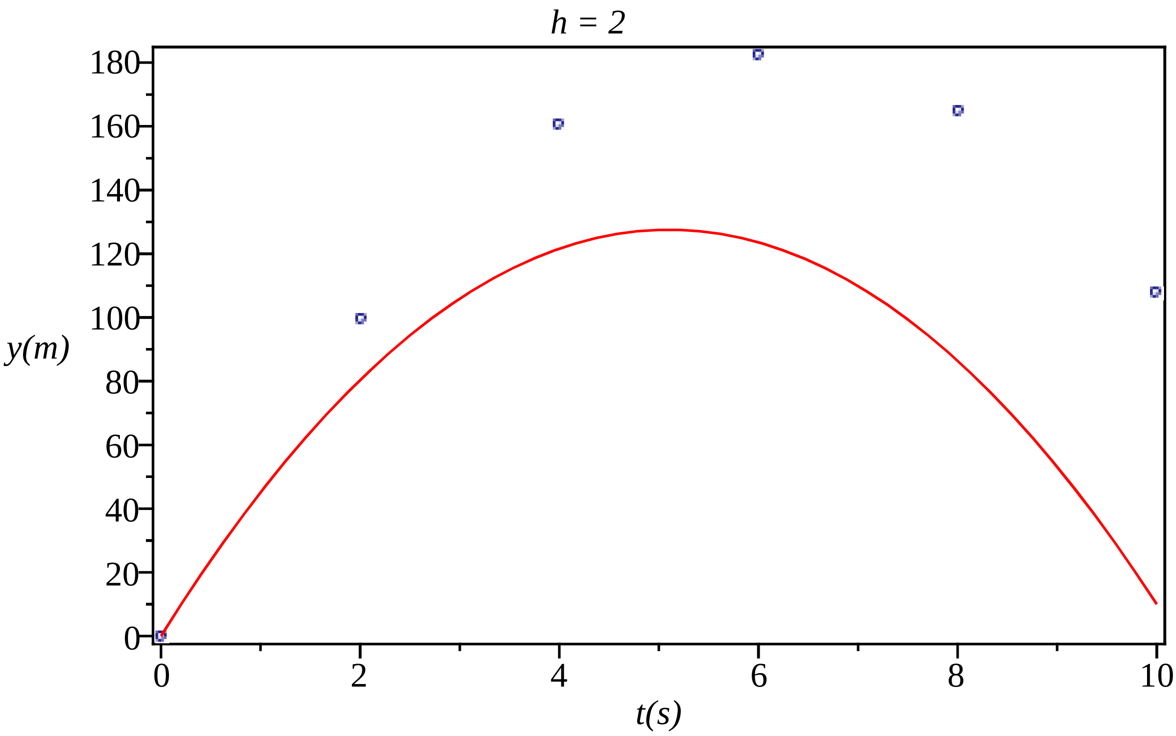
Passemos então ao cálculo da velocidade em função do tempo:

```
> restart: with(plots):
> tt := 10: h := 2: np := 1 + tt/h: t[0] := 0:
  v[0] := 50: g := 9.8:
  for i from 0 to (np-2) do
  t[i+1] := (i+1)*h:
  v[i+1] := v[i] - g*h:
  od:
> gr3 := pointplot([seq([t[i], v[i]], i=0..(np-1))], labels =
  [`t(s)`, `v(m/s)`], color=navy, symbol=circle, axes=BOXED):
  gr4 := plot(v[0]-g*t,t=0..10, color=red):
  display(gr3,gr4, title= `h = 2`);
```



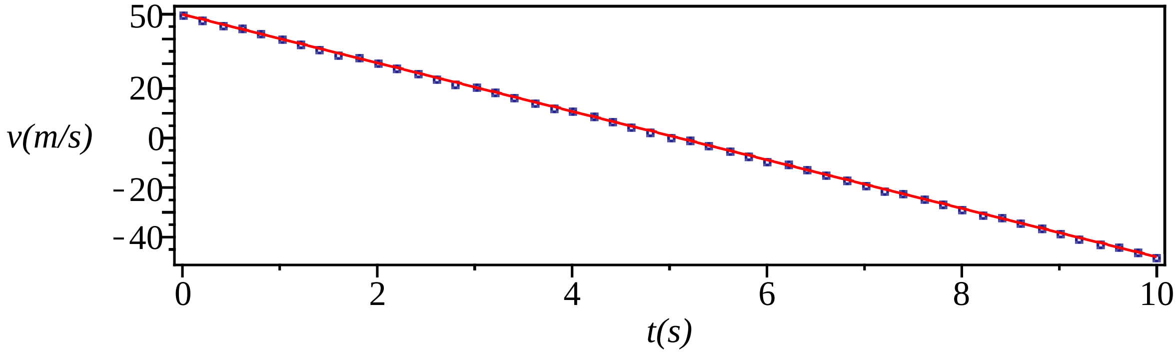
Agora que conhecemos a velocidade, podemos calcular a posição em função do tempo:

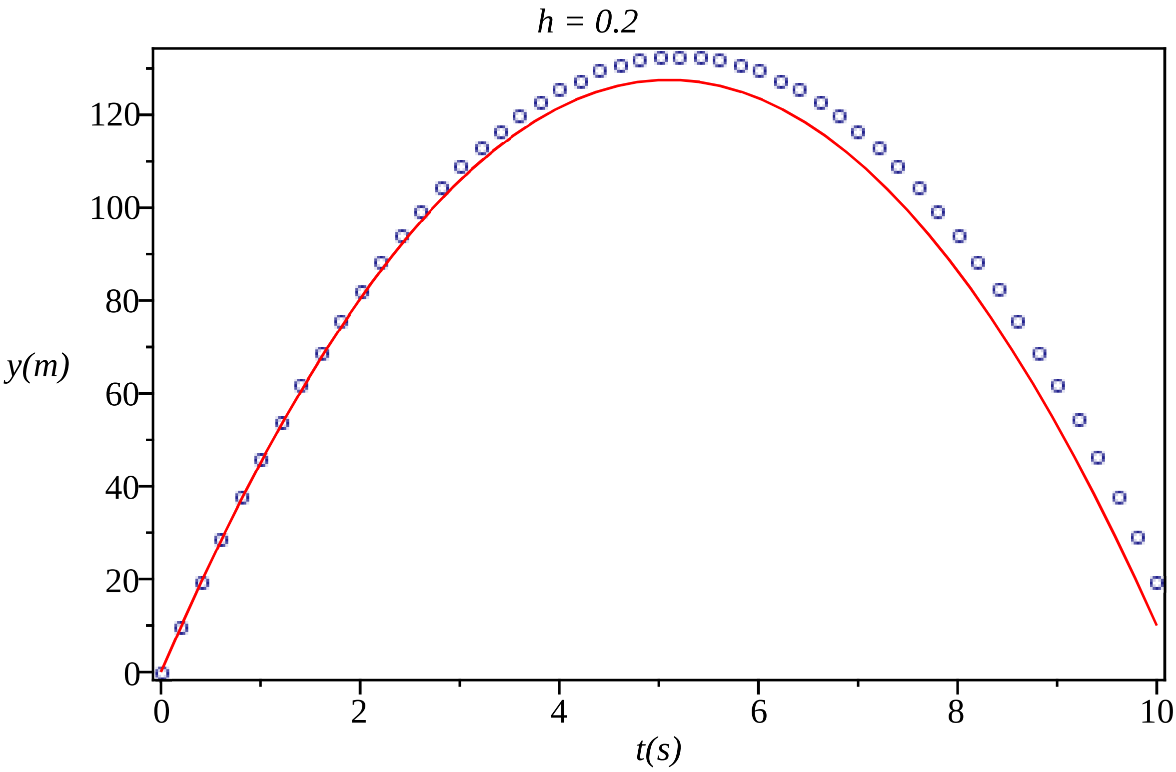
```
> y[0] := 0:
  for j from 0 to (np-2) do
  y[j+1] := y[j] + v[j]*h:
  od:
  gr1 := pointplot([seq([t[j], y[j]], j=0..(np-1))], labels =
  [`t(s)`, `y(m)`], color=navy, symbol=circle, axes=BOXED):
  gr2 := plot(v[0]*t-1/2*g*t^2, t=0..10, color=red):
  display(gr1, gr2, title= `h = 2`);
```



Refazemos agora os cálculos para um valor de h bem menor,

```
> with(plots):
> tt := 10: h := 0.2:  np := 1 + tt/h:
  t[0] := 0: v[0] := 50: g := 9.8:
  for i from 0 to (np-2) do
                   t[i+1] := (i+1)*h:
                  v[i+1] := v[i] - g*h:
  od:
> gr3 := pointplot([seq([t[i], v[i]], i=0..(np-1))], labels =
  [`t(s)`, `v(m/s)`], color=navy, symbol=circle, axes=BOXED):
  gr4 := plot(v[0]-g*t,t=0..10, color=red):
  y[0] := 0:
  for j from 0 to (np-2) do
  y[j+1] := y[j] + v[j]*h:
  od:
  gr1 := pointplot([seq([t[j], y[j]], j=0..(np-1))], labels =
  [`t(s)`, `y(m)`], color=navy, symbol=circle, axes=BOXED):
  gr2 := plot(v[0]*t-1/2*g*t^2, t=0..10, color=red):
  display(gr3, gr4);
   display(gr1, gr2, title= `h = 0.2`);
```





Método de Euler ou Runge - Kutta de Ordem 1

```
> restart; with(DEtools): with(plots):
> Digits:=8:
> x:='x': y:='y':
> f:= (x,y)-> y(x)^2 +1;
```

$$f := (x, y) \to y(x)^2 + 1 \tag{13}$$

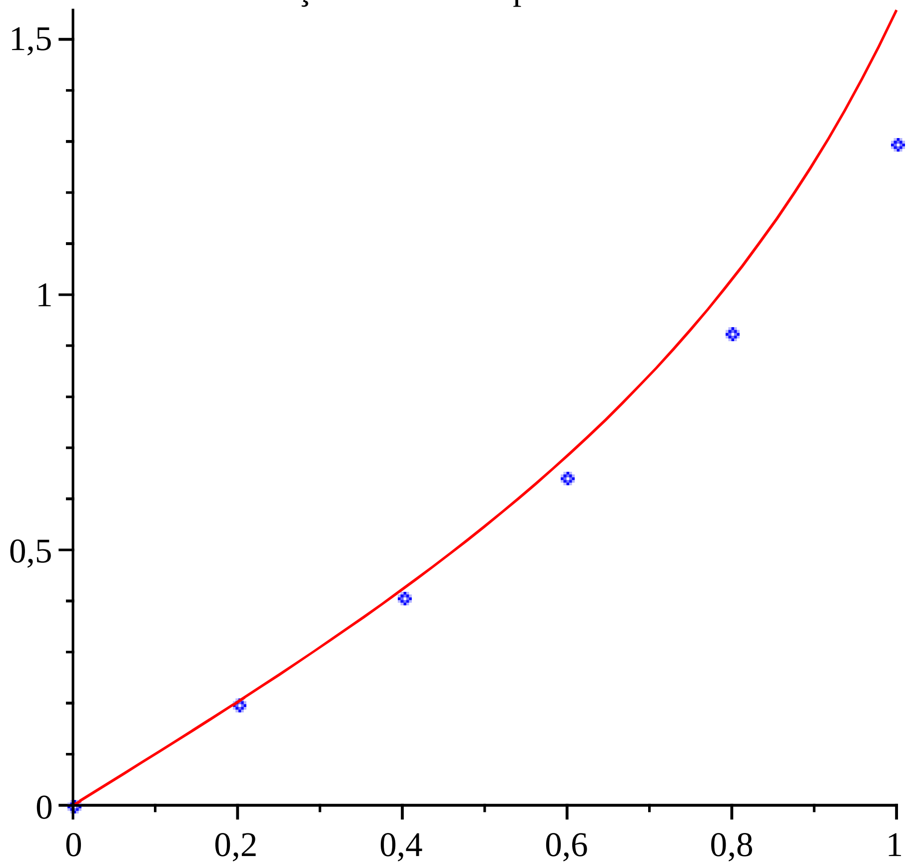
```
> h:=0.2:  N:=5:  x(0):=0:  y(0):=0:        # [a, b] = [0, 1]:

> for n from 0 to N do
>    y(n+1):= y(n) + h*f(x(n),y(n));
>    x(n+1):= x(n)+h;
> od:
> seq([x(n),y(n), tan(x(n)), tan(x(n))-y(n)],n=0..N);
```

[0, 0, 0, 0], [0.2, 0.2, 0.20271004, 0.00271004], [0.4, 0.408, 0.42279322, 0.01479322], **(14)**

$[0.6, 0.6412928, 0.68413681, 0.04284401], [0.8, 0.92354410, 1.0296386, 0.10609450],$
$[1.0, 1.2941308, 1.5574077, 0.2632769]$

```
> P1:= plot({seq([x(n),y(n)], n=0..N)}, style = POINT, color =
  blue):
> P2:= plot(tan(x), x=0..1):
> display(P1,P2, title="Soluções: Exata e Aproximada");
```



Cálculo da solução analítica do PVI - usamos o seguinte comando

```
> restart; with(DEtools):
> ode1 := diff(y(x), x) = y(x)^2 + 1:

> solução := dsolve( {ode1, y(0)=0}, y(x));
```

$$solução := y(x) = \tan(x) \tag{15}$$

Podemos também usar para o cálculo so solução analítica do PVI o comanado

```
> dsolve( {diff(y(x),x) = y(x)^2+1, y(0) = 0}, y(x));
```

$$y(x) = \tan(x) \tag{16}$$

Considere o problema de valor inicial constituído da equação diferencial $\frac{d}{dx}\,y(x) = f(x, y(x))$ e da condição inicial $y(x_0) = y_0$.

E queremos saber o valor da solução do pvi (Problema de Valor Inicial) num ponto x.
Vamos considerar o passo $h = x_{i+1} - x_i$.

A reta que r_0 que passa por $[x_0, y_0]$ com coeficiente angular $\frac{d}{dx}\, y(x)$ é

```
> r := x -> y[0] + D(y)(x[0]) * (x - x[0]): ' r(x) ' = r(x);
```

$$r(x) = y_0 + D(y)(x_0)\,(x - x_0) \tag{17}$$

é conhecida. Portanto escolhemos o valor da solução no ponto $x_1 = x_0 + h$ como o valor de $r(x_0 + h)$ que é

```
>  y[1] = y[0] + D(y)(x[0]) * h;
```

$$y_1 = y_0 + D(y)(x_0)\,h \tag{18}$$

E assim, repetimos sucessivamente o raciocínio para encontrar o val or de y num ponto x_2:

```
> y[2] := y[1] + D(y)(x[1]) * h;
```

$$y_2 := y_1 + D(y)(x_1)\,h \tag{19}$$

Seja $\frac{d}{dx}\, y(x) = x - y$.

```
> f := (x, y) -> x -  y:  'f(x,y)'= f(x,y);
```

$$f(x, y) = x - y \tag{20}$$

e que

```
> x[0] := 2; y[0] := 2;
```

$$x_0 := 2$$

$$y_0 := 2 \tag{21}$$

Seja

```
>  h := 0.1;
```

$$h := 0.1 \tag{22}$$

```
> for i from 1 to 3 do
  x[i] :=  x[0] + (i)*h;   y[i] := y[i-1] + h * f(x[i-1], y[i-1]
  ) od;
```

$$x_1 := 2.1$$

$$y_1 := 2.$$

$$x_2 := 2.2$$

$$y_2 := 2.01$$

$$x_3 := 2.3$$

$$y_3 := 2.029 \tag{23}$$

```
> for i from 1 to 3 do print( [x[i], y[i]]); od;
```

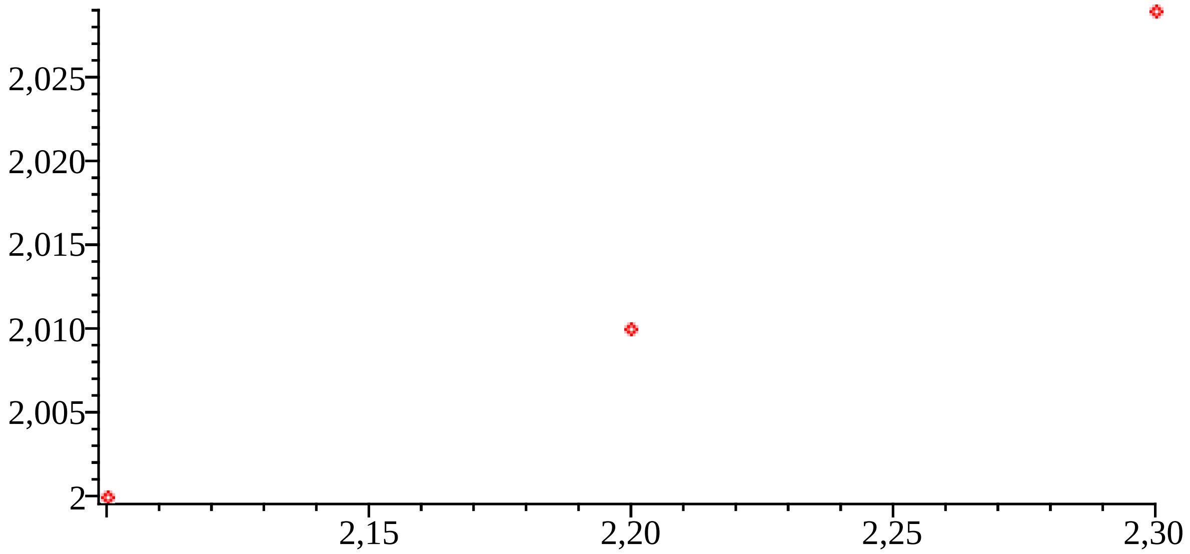
$$[2.1, 2.]$$

$$[2.2, 2.01]$$

$$[2.3, 2.029] \tag{24}$$

```
> restart; with(plots):
```

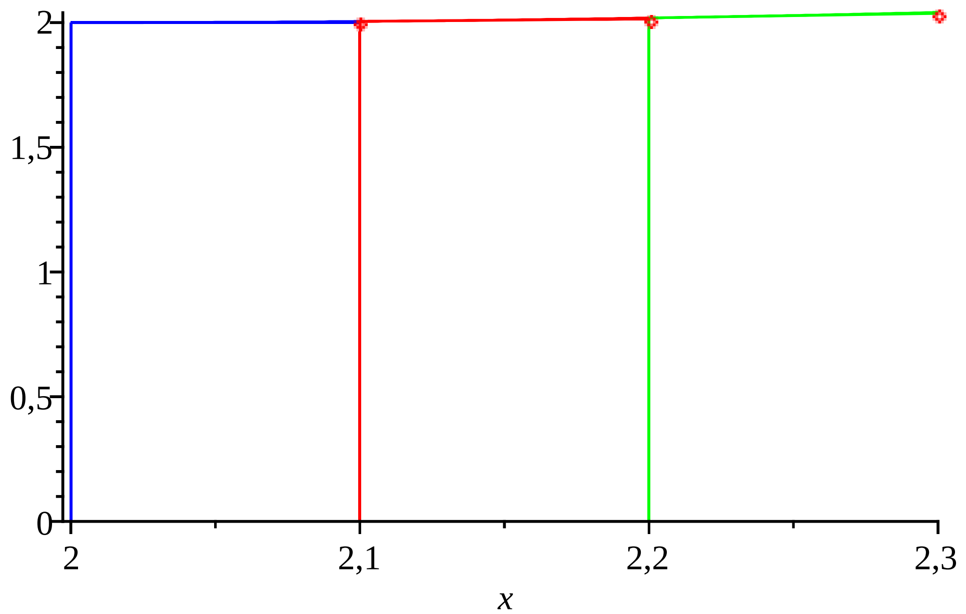
```
> pointplot( {[2.1, 2], [2.2, 2.01], [2.3, 2.029]}, color =red);
  gp :=%:
```




```
> S := dsolve( {diff(y(x), x) =  x -  y(x), y(2)=2} , y(x));
```

$$S := y(x) = -1 + x + \frac{e^{-x}}{e^{-2}} \tag{25}$$

```
> student[showtangent](rhs(S),x=2,x=2..2.1, color =blue): gs1 :=
  %: student[showtangent](rhs(S),x=2.1,x=2.1..2.2,color=red):
  gs2 := %: student[showtangent](rhs(S),x=2.2,x=2.2..2.3, color
  =green): gs3 := %:
> display( gs1, gs2, gs3, gp);
```




```
> restart;  # Euler
```

Na equação

```
> f := (x, y) -> y;
```

$$f := (x, y) \rightarrow y \tag{26}$$

```
> x[0] := 0; y[0] := 1;  h := 0.02;
```

$$x_0 := 0$$

$$y_0 := 1$$

$$h := 0.02 \tag{27}$$

```
> x[i] := x[0] + i*h;
```

$$x_i := 0.02\, i \tag{28}$$

```
> for i from 0 to 10 do  y[i+1] := y[i] + h*f(x[i], y[i]) od;
```

$$y_1 := 1.02$$

$$y_2 := 1.0404$$

$$y_3 := 1.061208$$

$$y_4 := 1.08243216$$

$$y_5 := 1.104080803$$

$$y_6 := 1.126162419$$

$$y_7 := 1.148685667$$

$$y_8 := 1.171659380$$

$$y_9 := 1.195092568$$

$$y_{10} := 1.218994419$$

$$y_{11} := 1.243374307 \tag{29}$$

```
> restart; with(plots): #Euler
> eq1 := diff(x(t), t) = 2 * t * x(t);
```

$$eq1 := \frac{\mathrm{d}}{\mathrm{d}t}\, x(t) = 2\, t\, x(t) \tag{30}$$

```
> f := (t, x) -> 2* t * x;
```

$$f := (t, x) \rightarrow 2\, t\, x \tag{31}$$

```
> t[0] := 1; x[0] := 1; h := .1;
```

$$t_0 := 1$$

$$x_0 := 1$$

$$h := 0.1 \tag{32}$$

```
> for i from 0 to 10  do t[i] := t[0]+i*h;
                           x[i+1] := x[i]+f(t[i], x[i])*h     od;
```

$$t_0 := 1.$$

$$x_1 := 1.2$$

$$t_1 := 1.1$$

$$x_2 := 1.464$$

$$t_2 := 1.2$$

$$x_3 := 1.81536$$

$$t_3 := 1.3$$

$$x_4 := 2.2873536$$

$$t_4 := 1.4$$

$$x_5 := 2.927812608$$

$$t_5 := 1.5$$

$$x_6 := 3.806156390$$

$$t_6 := 1.6$$

$$x_7 := 5.024126435$$

$$t_7 := 1.7$$

$$x_8 := 6.732329423$$

$$t_8 := 1.8$$

$$x_9 := 9.155968015$$

$$t_9 := 1.9$$

$$x_{10} := 12.63523586$$

$$t_{10} := 2.0$$

$$x_{11} := 17.68933020 \qquad \textbf{(33)}$$

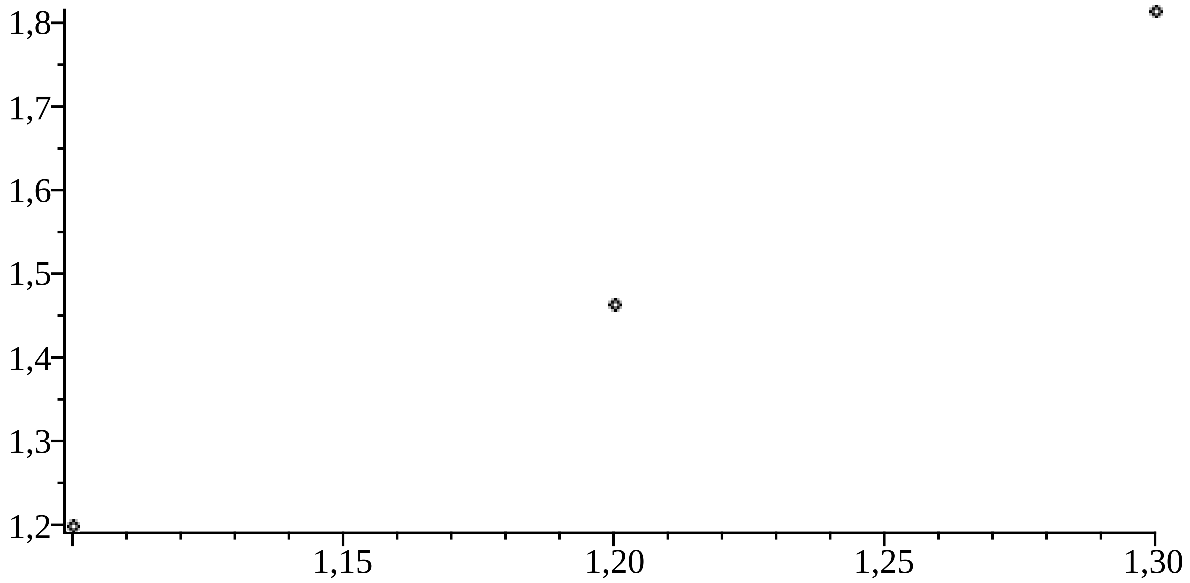
```
> for i from 1 to 3 do print([t[i], x[i]]) od:
```

$$[1.1, 1.2]$$

$$[1.2, 1.464]$$

$$[1.3, 1.81536] \qquad \textbf{(34)}$$

```
> pointplot( { [1.1, 1.2], [1.2, 1.464], [1.3, 1.81536]}); gp3
  := %:
```



```
> restart;  with(plots):
> eq1 := diff(x(t), t) = 2 * t * x(t);
```

$$eq1 := \frac{\mathrm{d}}{\mathrm{d}t}\, x(t) = 2\, t\, x(t) \tag{35}$$

```
> dsolve({ eq1, x(1) = 1}, x(t));
```

$$x(t) = \frac{\mathrm{e}^{t^2}}{\mathrm{e}} \tag{36}$$

```
> simplify(%);
```

$$x(t) = \mathrm{e}^{(t-1)\,(t+1)} \tag{37}$$

```
> x := unapply( rhs(%), t);
```

$$x := t \rightarrow \mathrm{e}^{(t-1)\,(t+1)} \tag{38}$$

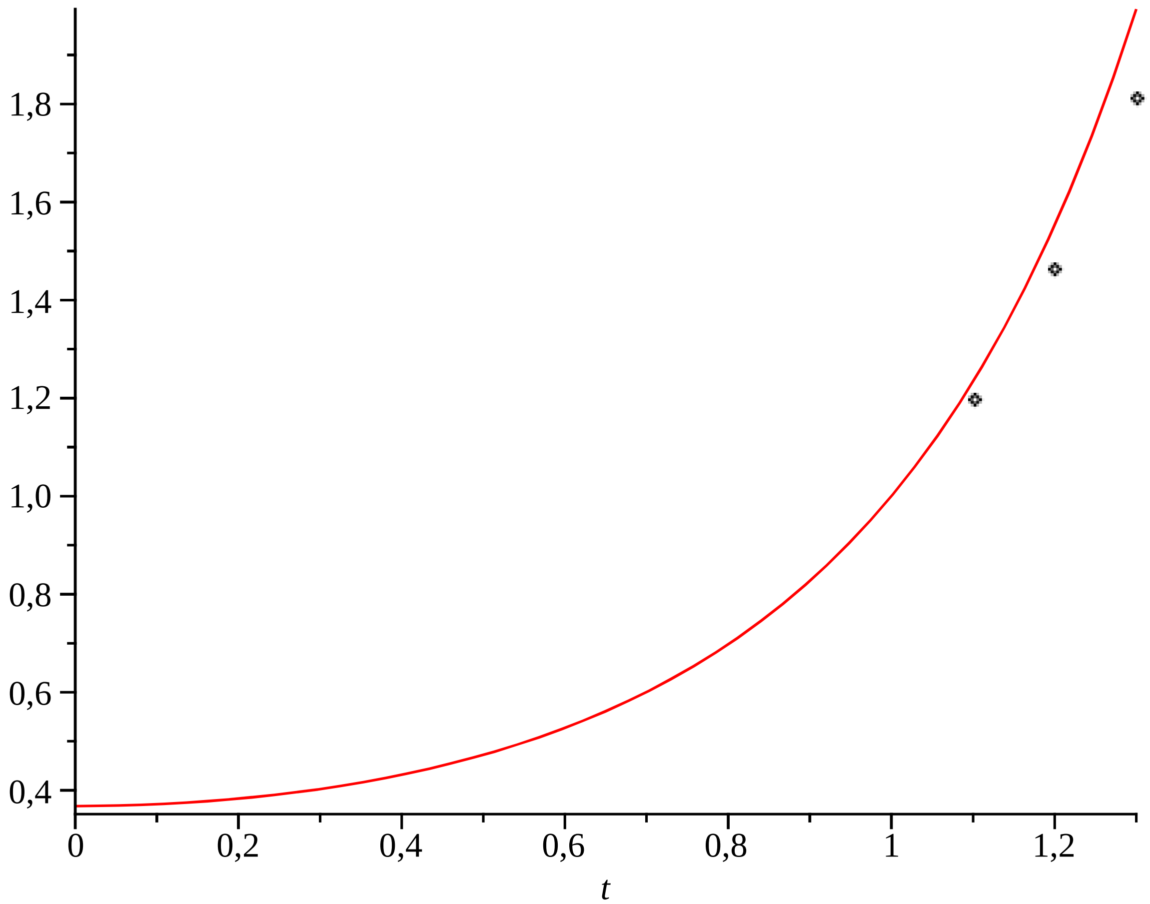
```
> 'x(1.1)' = x(1.1); 'x(1.2)' = x(1.2); 'x(1.3)' = x(1.3);
```

$$x(1.1) = 1.233678060$$

$$x(1.2) = 1.552707219$$

$$x(1.3) = 1.993715533 \tag{39}$$

```
> gp3 := pointplot( { [1.1, 1.2], [1.2, 1.464], [1.3, 1.81536]})
  :
> gx := plot( x(t), t=0..1.3):
> display( gp3,  gx);
```




```
> restart; with(plots): # Taylor
```

Na equação

```
> f := (x, y) -> 1 -  y/x;
```
$$f := (x, y) \to 1 - \frac{y}{x} \tag{40}$$

```
>  x[0] := 2; y[0] := 2; h := 0.1;
```
$$x_0 := 2$$
$$y_0 := 2$$
$$h := 0.1 \tag{41}$$

Aproximação linear: Euler

```
> y[i] := y[0] + (x - x[0])*f(x[0], y[0]);
```
$$y_i := 2 \tag{42}$$

```
> subs( x= 2.1, %);
```
$$2 \tag{43}$$

Aproximação por um polinômio de segunda

```
> y[i] := y[0] + (x - x[0]) * f(x[0], y[0]) + (x - x[0])^2/2* (
  D[1](f)(x[0],y[0]) + D[2](f)(x[0],y[0]) * f(x[0], y[0]) );
```
$$y_i := 2 + \frac{1}{4}\,(x-2)^2 \tag{44}$$

```
> subs( x=2.1, %);
```
$$2.002500000 \tag{45}$$

Aproximação por um polinômio de 3a. ordem

```
> y[i] := y[0] + (x - x[0])*f(x[0], y[0]) + (x - x[0])^2/2*
  (D[1](f)(x[0], y[0]) + D[2](f)(x[0],y[0]) * f(x[0], y[0]) )  +
  (x-x[0])^3/3! * ( D[1,1](f)(x[0], y[0]) + 2*D[1,2](f)(x[0], y
  [0])*f(x[0], y[0]) + D[2,2](f)(x[0], y[0]) * f(x[0], y[0])^2 +
  D[1](f)(x[0], y[0]) * D[2](f)(x[0], y[0]) + (D[2](f)(x[0], y
  [0]))^2 * f(x[0], y[0]));
> subs( x=2.1, %);
```
$$y_i := 2 + \frac{1}{4}\,(x-2)^2 - \frac{1}{8}\,(x-2)^3$$
$$2.002375000 \tag{46}$$

```
> restart; with(plots):
> s(2.1);
```
$$s(2.1) \tag{47}$$

```
> restart;
> dsolve({ x*diff(y(x),x) = x - y(x), y(2) = 2},  y(x), type=
  series);
```
$$y(x) = 2 + \frac{1}{4}\,(x-2)^2 - \frac{1}{8}\,(x-2)^3 + \frac{1}{16}\,(x-2)^4 - \frac{1}{32}\,(x-2)^5 + \mathrm{O}\left((x-2)^6\right) \tag{48}$$

```
> y:= unapply( convert( rhs(%), polynom), x);
```
$$y := x \to 2 + \frac{1}{4}\,(x-2)^2 - \frac{1}{8}\,(x-2)^3 + \frac{1}{16}\,(x-2)^4 - \frac{1}{32}\,(x-2)^5 \tag{49}$$

```
> y(x);
```

$$2+\frac{1}{4}(x-2)^2-\frac{1}{8}(x-2)^3+\frac{1}{16}(x-2)^4-\frac{1}{32}(x-2)^5 \quad \textbf{(50)}$$

```
> 'y(1.1)'= y(1.1); 'y(1.3)'= y(1.3);
```

$$y(1.1)=2.353084062$$
$$y(1.3)=2.185633438 \quad \textbf{(51)}$$

Ex. Considere o problema de encontrar a solução do pvi constituído da eq1 $\frac{d}{dx}y(x)=\frac{1(2y+x+1)}{x}$ e da condição inicial $y(1)=0.5$

```
> restart; with(plots):
> eq1 := diff(y(x), x) = 1/x*(2*y(x)+x+1);
```

$$eq1 := \frac{d}{dx}y(x)=\frac{2y(x)+x+1}{x} \quad \textbf{(52)}$$

```
> ci := y(1)=0.5;
```

$$ci := y(1)=0.5 \quad \textbf{(53)}$$

```
> dsolve({eq1,ci}, y(x));
```

$$y(x)=-x-\frac{1}{2}+2x^2 \quad \textbf{(54)}$$

```
> y2 := subs( x=2., rhs(%));
```

$$y2 := 5.500000000 \quad \textbf{(55)}$$

Agora numéricamente:

```
> Digits := 4:
> h := 0.1;
```

$$h := 0.1 \quad \textbf{(56)}$$

```
> x[0] := 1; y[0] := 0.5;
```

$$x_0 := 1$$
$$y_0 := 0.5 \quad \textbf{(57)}$$

```
> f := (x,y) -> 1/x*(2*y+x+1): 'f(x,y)' = f(x,y);
```

$$f(x,y)=\frac{2y+x+1}{x} \quad \textbf{(58)}$$

```
> y[1] := y[0]+f(x[0],y[0])*h;
```

$$y_1 := 0.80 \quad \textbf{(59)}$$

```
> for i from 0 to 10  do x[i]:=x[0]+i*h;y[i+1]:=y[i]+f(x[i],y[i]
  )*h od;
```

$$x_0 := 1.$$
$$y_1 := 0.80$$
$$x_1 := 1.1$$
$$y_2 := 1.136$$

$$x_2 := 1.2$$

$$y_3 := 1.509$$

$$x_3 := 1.3$$

$$y_4 := 1.918$$

$$x_4 := 1.4$$

$$y_5 := 2.363$$

$$x_5 := 1.5$$

$$y_6 := 2.845$$

$$x_6 := 1.6$$

$$y_7 := 3.363$$

$$x_7 := 1.7$$

$$y_8 := 3.917$$

$$x_8 := 1.8$$

$$y_9 := 4.508$$

$$x_9 := 1.9$$

$$y_{10} := 5.135$$

$$x_{10} := 2.0$$

$$y_{11} := 5.798 \tag{60}$$

```
> for i from 0 to 10 do print ([x[i],y[i]]);od;
```

$$[1., 0.5]$$

$$[1.1, 0.80]$$

$$[1.2, 1.136]$$

$$[1.3, 1.509]$$

$$[1.4, 1.918]$$

$$[1.5, 2.363]$$

$$[1.6, 2.845]$$

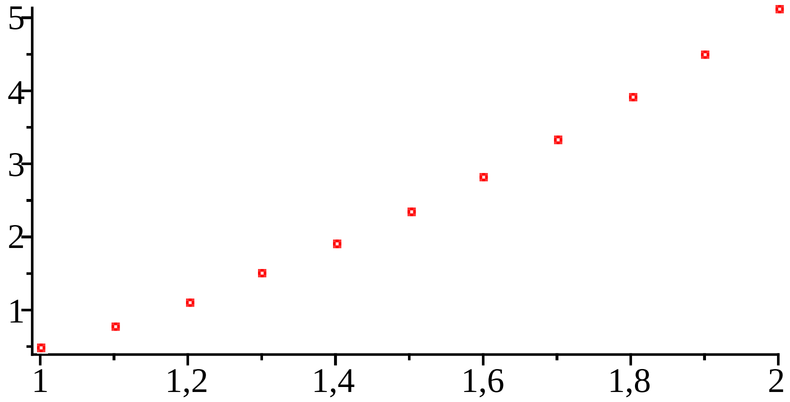
$$[1.7, 3.363]$$

$$[1.8, 3.917]$$

$$[1.9, 4.508]$$

$$[2.0, 5.135] \tag{61}$$

```
> pointplot ({[1,0.5], [1.1,.8],  [1.2, 1.136],[1.3,1.509],[1.4,
  1.918],[1.5,2.363],[1.6,2.845],[1.7,3.363],[1.8,3.917],[1.9,
  4.508], [2, 5.135]},color=red,symbol=circle); gp:=%:
```



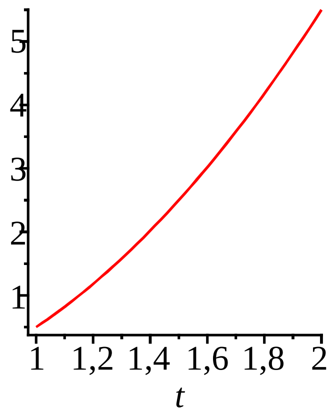

```
> dsolve({diff(r(t), t) = 1/t*(2*r(t)+t+1), r(1)=.5}, r(t));
```

$$r(t) = -t - \frac{1}{2} + 2\,t^2 \qquad \textbf{(62)}$$

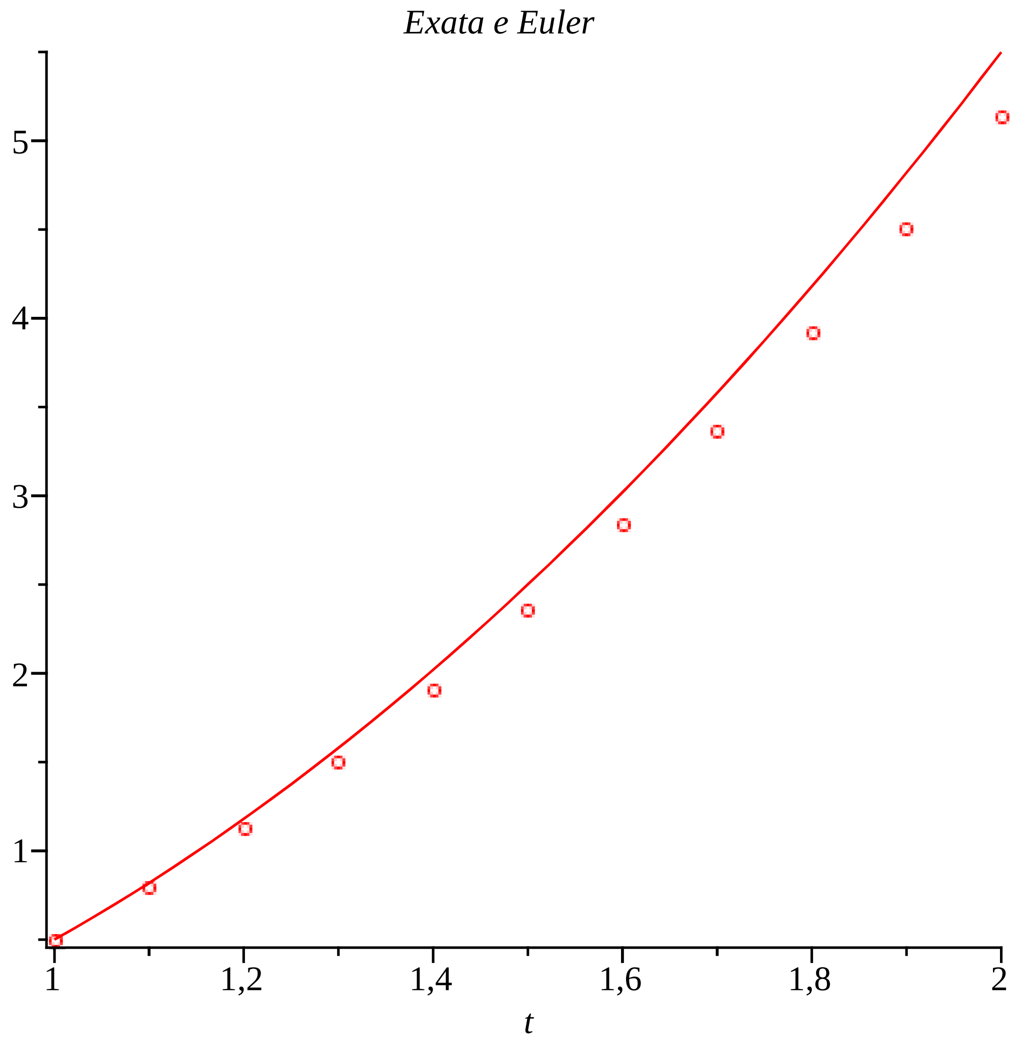
```
> r := unapply(rhs(%), t);
```

$$r := t \rightarrow -t - \frac{1}{2} + 2\,t^2 \qquad \textbf{(63)}$$

```
> plot( r(t), t=1..2); gf:=%:
```




```
> display(gp, gf, title = `Exata e Euler` );
```



Calcule o erro cometido ao aproximar $y(1.3)$ pelo método de Euler.

Método de Runge- Kutta de Ordem 2

Vamos, a seguir, resolver o mesmo PVI usando um método de ordem p = 2 baseado na Série de Taylor

y(n+1) := y(n) + h*f(x(n),y(n)) (h^2/2) (f(x)+f*f(y))

```
> restart; with(DEtools): with(plots):
> Digits:=8:
> x:='x': y:='y':
> f:= (x,y)-> y(x)^2+1;
```

$$f := (x, y) \rightarrow y(x)^2 + 1 \quad (64)$$

```
> f1:= (x,y)-> 2*y(x)^3+2*y;
```

$$f1 := (x, y) \rightarrow 2\, y(x)^3 + 2\, y \quad (65)$$

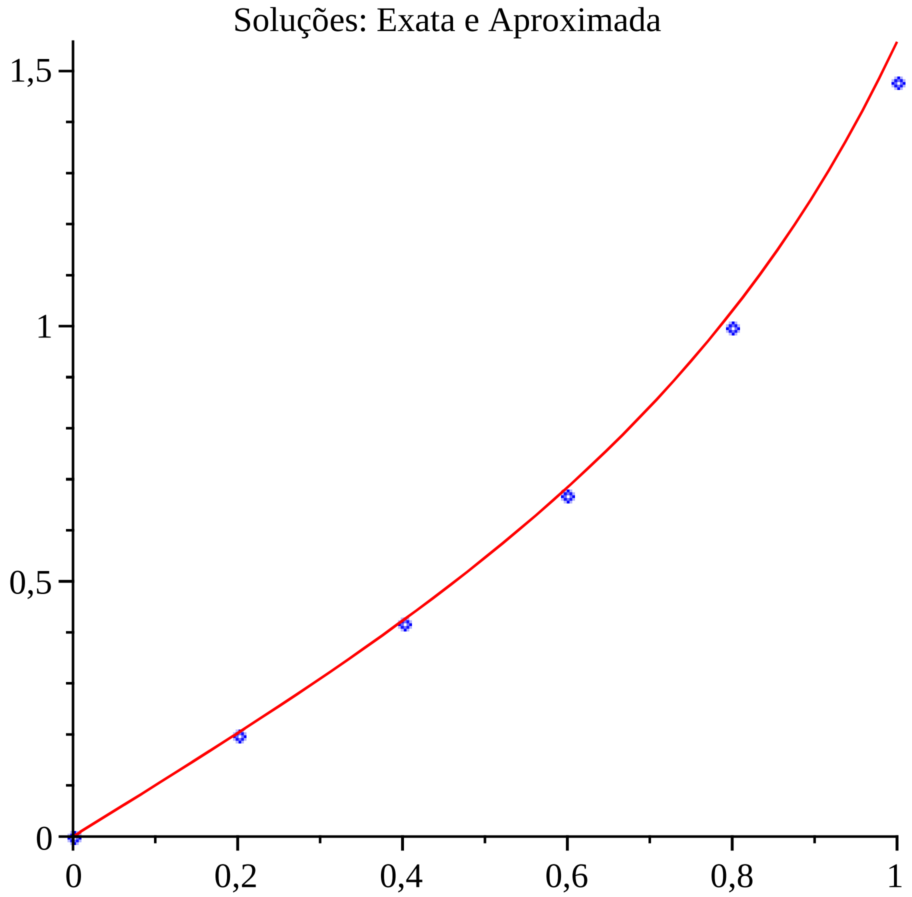
```
> h:=0.2:  N:=5:  x(0):=0:  y(0):=0:        # [a, b] = [0, 1]:
```

```
> for n from 0 to N do
>    y(n+1):= y(n) + h*f(x(n),y(n))+(h^2/2)*f1(x(n),y(n));
>    x(n+1):= x(n)+h;
> od:
```

```
> seq([x(n),y(n), tan(x(n)), tan(x(n))-y(n)], n=0..N);
```

$$[0, 0, 0, 0], [0.2, 0.2, 0.20271004, 0.00271004], [0.4, 0.41632000, 0.42279322, 0.00647322], [0.6, 0.67052356, 0.68413681, 0.01361325], [0.8, 0.99932361, 1.0296386, 0.03031499], [1.0, 1.4789450, 1.5574077, 0.0784627] \quad (66)$$

```
> P1:= plot({seq([x(n),y(n)], n=0..N)}, style = POINT, color =
  blue):
> P2:= plot(tan(x), x=0..1):
> display(P1,P2, title="Soluções: Exata e Aproximada");
```



Runge-Kutta de Ordem 2 - Euler Modificado

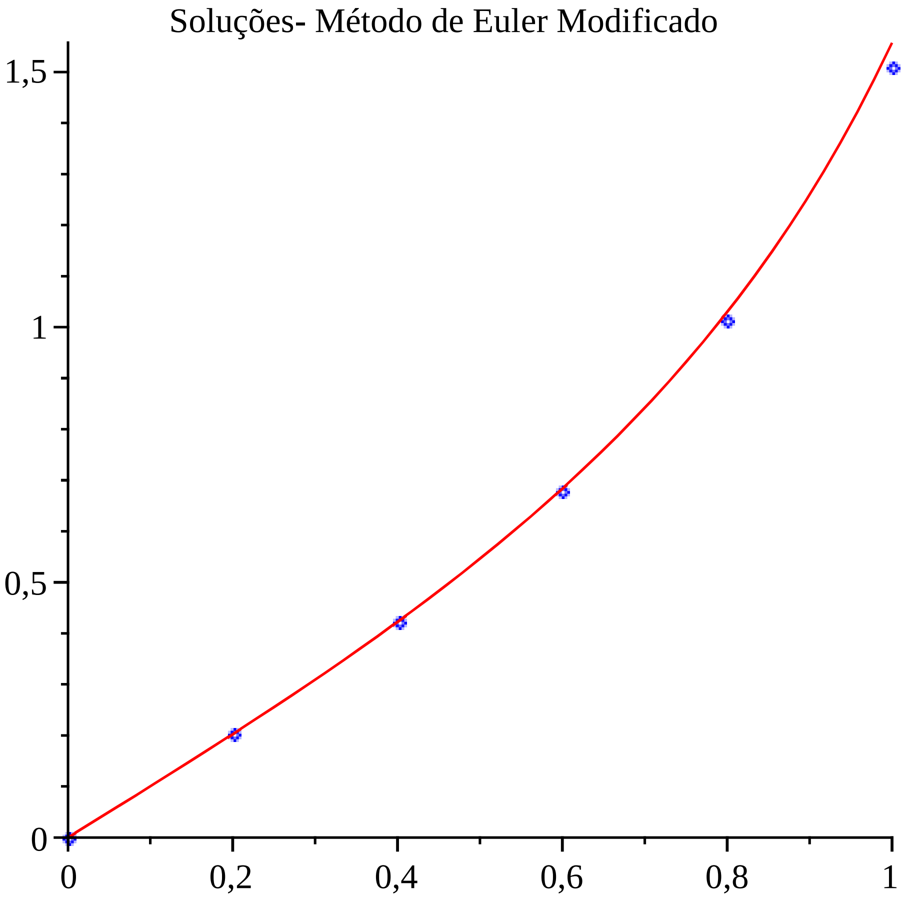
```
> restart; with(DEtools): with(plots):
> Digits:=8:
> x:='x': y:='y':
> f:= (x,y)-> y(x)^2+1;
```

$$f := (x, y) \to y(x)^2 + 1 \quad (67)$$

```
> h:=0.2:  N:=5:  x(0):=0:  y(0):=0:        # [a, b] = [0, 1]:
> for n from 0 to N do
>     K1:= f(x(n),y(n));
>     K2:= f(x(n)+(h/2),y(n)+(h/2)*K1);
>   y(n+1):= y(n)+h*K2;
>   x(n+1):= x(n)+h;
> od:
> seq([x(n),y(n),tan(x(n)),tan(x(n))-y(n)],n=0..N);
```

$$[0, 0, 0, 0], [0.2, 0.20200000, 0.20271004, 0.00071004], [0.4, 0.42073704, 0.42279322, 0.00205618], [0.6, 0.67872036, 0.68413681, 0.00541645], [0.8, 1.0147749, 1.0296386, 0.0148637], [1.0, 1.5113587, 1.5574077, 0.0460490] \quad (68)$$

```
> P1:= plot({seq([x(n),y(n)],n=0..N)},style = POINT, color =
  blue):
> P2:= plot(tan(x),x=0..1):
> display(P1,P2, title="Soluções- Método de Euler Modificado" );
```



Runge-Kutta de ordem 2 - Euler Melhorado

```
> restart;  with(DEtools): with(plots):
```

```
> Digits:=8:
> x:='x': y:='y':
> f:= (x,y)-> y(x)^2+1;
```

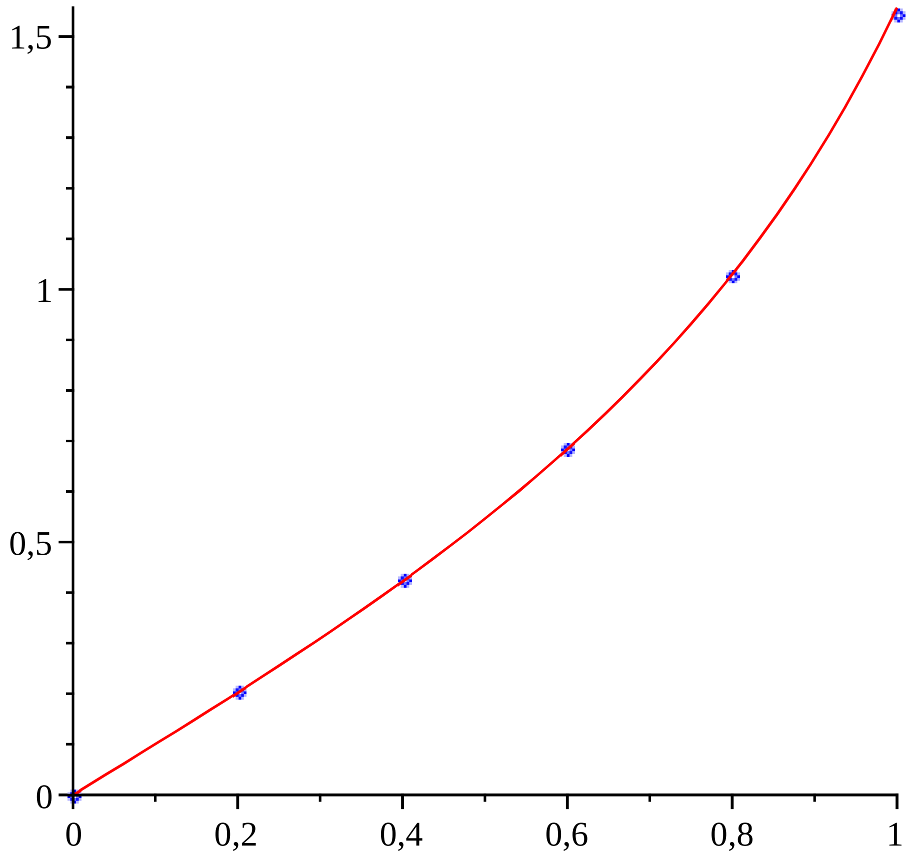
$$f := (x, y) \rightarrow y(x)^2 + 1 \tag{69}$$

```
> h:=0.2:  N:=5:  x(0):=0:  y(0):=0:        # [a, b] = [0, 1]:
> for n from 0 to N do
>     K1:= f(x(n),y(n));
>     K2:= f(x(n)+h,y(n)+h*K1);
>   y(n+1):= y(n)+(h/2)*(K1 + K2);
>   x(n+1):= x(n)+h;
> od:
> seq([x(n),y(n),tan(x(n)),tan(x(n))-y(n)],n=0..N);
```

$$[0, 0, 0, 0], [0.2, 0.20400000, 0.20271004, -0.00128996], [0.4, 0.42516264, 0.42279322, -0.00236942], [0.6, 0.68697276, 0.68413681, -0.00283595], [0.8, 1.0304725, 1.0296386, -0.0008339], [1.0, 1.5448406, 1.5574077, 0.0125671] \tag{70}$$

```
> P1:= plot({seq([x(n),y(n)],n=0..N)},style = POINT, color =
  blue):
> P2:= plot(tan(x),x=0..1):
> display(P1,P2,title="Soluções- Método de Euler Melhorado");
```

Soluções- Método de Euler Melhorado



e) Runge-Kutta de Ordem R = 3

```
> restart; with(DEtools): with(plots):
> Digits:=8:
> x:='x': y:='y':
> f:= (x,y)-> y(x)^2+1;
```

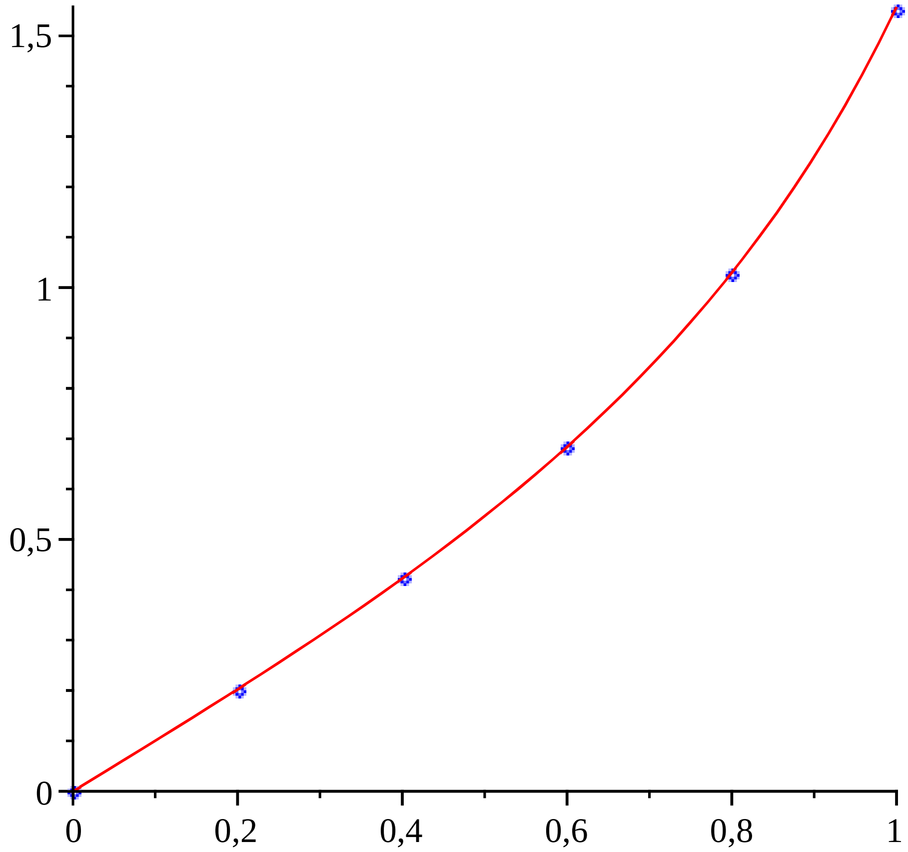
$$f := (x, y) \rightarrow y(x)^2 + 1 \tag{71}$$

```
> h:=0.2:  N:=5:  x(0):=0:  y(0):=0:        # [a, b] = [0, 1]:
> for n from 0 to N do
>     K1:= f(x(n),y(n));
>     K2:= f(x(n)+(h/3),y(n)+(h/3)*K1);
>     K3:= f(x(n)+((2*h)/3),y(n)+((2*h)/3)*K2);
>   y(n+1):= y(n)+h/4*(K1+3*K3);
>   x(n+1):= x(n)+h;
> od:
> seq([x(n),y(n),tan(x(n)),tan(x(n))-y(n)],n=0..N);
```

[0, 0, 0, 0], [0.2, 0.20269043, 0.20271004, 0.00001961], [0.4, 0.42269113, 0.42279322, 0.00010209], [0.6, 0.68375585, 0.68413681, 0.00038096], [0.8, 1.0282382, 1.0296386, 0.0014004], [1.0, 1.5515106, 1.5574077, 0.0058971] **(72)**

```
> P1:= plot({seq([x(n),y(n)],n=0..N)},style = POINT, color =
  blue):
> P2:= plot(tan(x),x=0..1):
> display(P1,P2, title="Soluções- Método de Runge Kutta de 3
  estágios " );
```

Soluções- Método de Runge Kutta de 3 estágios



Runge-Kutta de Ordem R = 4

```
> restart; with(DEtools): with(plots):
> Digits:=8:
> f:= (x,y)-> y(x)^2+1;
```

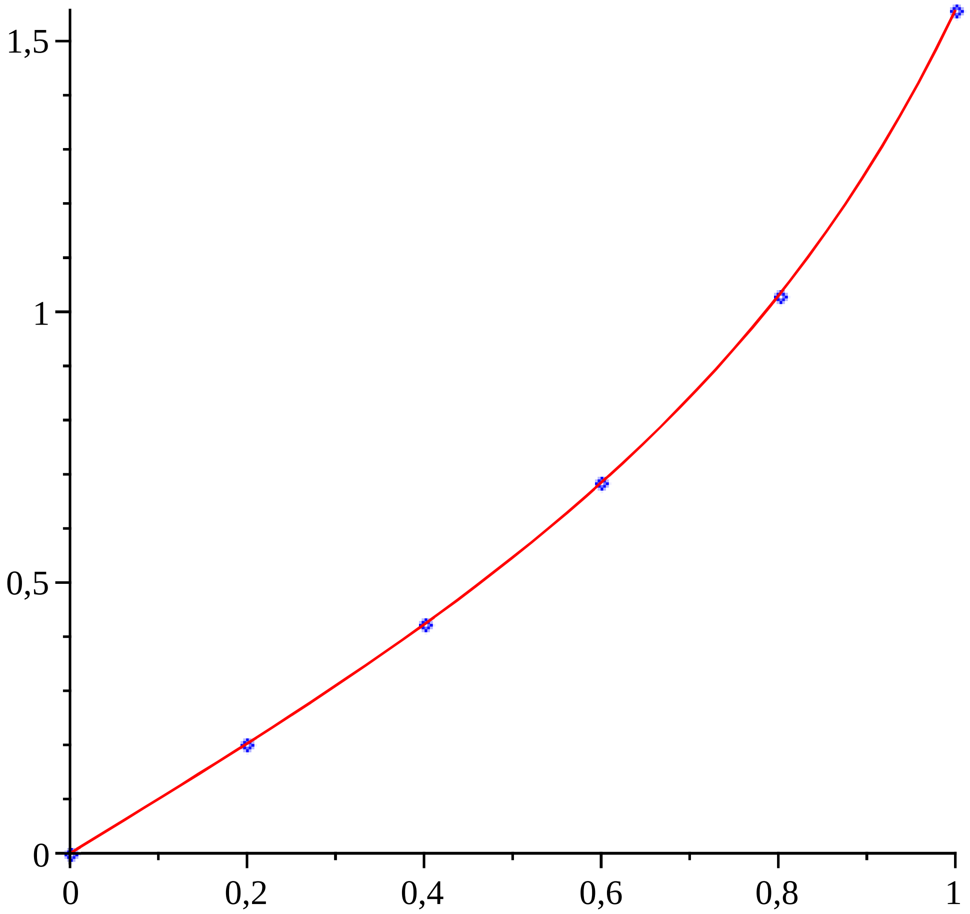
$$f := (x, y) \rightarrow y(x)^2 + 1 \qquad \textbf{(73)}$$

```
> h:=0.2:  N:=5:  x(0):=0:  y(0):=0:        # [a, b] = [0, 1]:
> for n from 0 to N do
>     K1:= f(x(n),y(n));
>     K2:= f(x(n)+(h/2),y(n)+(h/2)*K1);
>     K3:= f(x(n)+(h/2),y(n)+(h/2)*K2);
>     K4:= f(x(n)+h,y(n)+h*K3);
>   y(n+1):= y(n)+(h/6)*(K1 + 2*K2 + 2*K3 + K4);
>   x(n+1):= x(n)+h;
> od:
> seq([x(n),y(n),tan(x(n)),tan(x(n))-y(n)],n=0..N);
```

[0, 0, 0, 0], [0.2, 0.20270740, 0.20271004, 0.00000264], [0.4, 0.42278898, 0.42279322, 0.00000424], [0.6, 0.68413338, 0.68413681, 0.00000343], [0.8, 1.0296366, 1.0296386, 0.0000020], [1.0, 1.5573517, 1.5574077, 0.0000560] **(74)**

```
> P1:= plot({seq([x(n),y(n)],n=0..N)},style = POINT, color =
  blue):
> P2:= plot(tan(x),x=0..1):
> display(P1,P2,title="Soluções- Método de Runge - Kutta de 4
  estágios");
```

Soluções- Método de Runge - Kutta de 4 estágios



JAS 2012

```
>
>
```