

Higher-Order Taylor Methods

We begin by illustrating the Taylor methods of orders 2 and 5 in a step by step process by using the recursion formulas from the text. For the differential equation $y' = f(t, y)$, we enter right-hand side denoted as $f[0]$. In general, the n'th derivative of $f(t, y)$ with respect to t (recall $y = y(t)$) will be denoted as $f[n]$.

> restart;

> f[0] := -t*y(t) + (4*t)/y(t);

$$f_0 := -ty(t) + \frac{4t}{y(t)}$$

We find the first derivative.

> f[1] := diff(f[0], t);

$$f_1 := -y(t) - t \left(\frac{d}{dt} y(t) \right) + \frac{4}{y(t)} - \frac{4t \left(\frac{d}{dt} y(t) \right)}{y(t)^2}$$

We replace each instance of the derivative $\text{diff}(y(t), t)$ with $f[0]$.

> f[1] := subs(diff(y(t), t) = f[0], f[1]);

$$f_1 := -y(t) - t \left(-ty(t) + \frac{4t}{y(t)} \right) + \frac{4}{y(t)} - \frac{4t \left(-ty(t) + \frac{4t}{y(t)} \right)}{y(t)^2}$$

> f[1] := simplify(f[1]);

$$f_1 := \frac{-y(t)^4 + t^2 y(t)^4 + 4y(t)^2 - 16t^2}{y(t)^3}$$

We continue the process up to the 4th derivative.

> f[2] := diff(f[1], t);

$$f_2 := \frac{-4y(t)^3 \left(\frac{d}{dt} y(t) \right) + 2ty(t)^4 + 4t^2 y(t)^3 \left(\frac{d}{dt} y(t) \right) + 8y(t) \left(\frac{d}{dt} y(t) \right) - 32t}{y(t)^3} - \frac{3(-y(t)^4 + t^2 y(t)^4 + 4y(t)^2 - 16t^2) \left(\frac{d}{dt} y(t) \right)}{y(t)^4}$$

> f[2] := subs(diff(y(t), t) = f[0], f[2]);

$$f_2 := \frac{1}{y(t)^3} \left(-4y(t)^3 \left(-ty(t) + \frac{4t}{y(t)} \right) + 2ty(t)^4 + 4t^2 y(t)^3 \left(-ty(t) + \frac{4t}{y(t)} \right) + 8y(t) \left(-ty(t) + \frac{4t}{y(t)} \right) - 32t \right) - \frac{3(-y(t)^4 + t^2 y(t)^4 + 4y(t)^2 - 16t^2) \left(-ty(t) + \frac{4t}{y(t)} \right)}{y(t)^4}$$

> f[2] := simplify(f[2]);

$$f_2 := -\frac{t(-3y(t)^6 + y(t)^6 t^2 - 4t^2 y(t)^4 + 48y(t)^2 + 48t^2 y(t)^2 - 192t^2)}{y(t)^5}$$

> f[3]:=diff(f[2],t);

$$f_3 := -\frac{-3y(t)^6 + y(t)^6 t^2 - 4t^2 y(t)^4 + 48y(t)^2 + 48t^2 y(t)^2 - 192t^2}{y(t)^5} - \frac{1}{y(t)^5} \left(t \left(-18y(t)^5 \left(\frac{d}{dt} y(t) \right) + 6y(t)^5 t^2 \left(\frac{d}{dt} y(t) \right) + 2y(t)^6 t - 8ty(t)^4 - 16t^2 y(t)^3 \left(\frac{d}{dt} y(t) \right) + 96y(t) \left(\frac{d}{dt} y(t) \right) + 96ty(t)^2 + 96t^2 y(t) \left(\frac{d}{dt} y(t) \right) - 384t \right) + \frac{5t(-3y(t)^6 + y(t)^6 t^2 - 4t^2 y(t)^4 + 48y(t)^2 + 48t^2 y(t)^2 - 192t^2) \left(\frac{d}{dt} y(t) \right)}{y(t)^6} \right)$$

> f[3]:=subs(diff(y(t),t)=f[0],f[3]);

$$f_3 := -\frac{-3y(t)^6 + y(t)^6 t^2 - 4t^2 y(t)^4 + 48y(t)^2 + 48t^2 y(t)^2 - 192t^2}{y(t)^5} - \frac{1}{y(t)^5} \left(t \left(-18y(t)^5 \left(-ty(t) + \frac{4t}{y(t)} \right) + 6y(t)^5 t^2 \left(-ty(t) + \frac{4t}{y(t)} \right) + 2y(t)^6 t - 8ty(t)^4 - 16t^2 y(t)^3 \left(-ty(t) + \frac{4t}{y(t)} \right) + 96y(t) \left(-ty(t) + \frac{4t}{y(t)} \right) + 96ty(t)^2 + 96t^2 y(t) \left(-ty(t) + \frac{4t}{y(t)} \right) - 384t \right) + \frac{1}{y(t)^6} \left(5t(-3y(t)^6 + y(t)^6 t^2 - 4t^2 y(t)^4 + 48y(t)^2 + 48t^2 y(t)^2 - 192t^2) \left(-ty(t) + \frac{4t}{y(t)} \right) \right)$$

> f[3]:=simplify(f[3]);

$$f_3 := \frac{1}{y(t)^7} (3y(t)^8 - 6y(t)^8 t^2 + 24y(t)^6 t^2 - 48y(t)^4 - 288t^2 y(t)^4 + 1152t^2 y(t)^2 + t^4 y(t)^8 - 160t^4 y(t)^4 + 1536t^4 y(t)^2 - 3840t^4)$$

> f[4]:=diff(f[3],t);

$$f_4 := \frac{1}{y(t)^7} \left(24y(t)^7 \left(\frac{d}{dt} y(t) \right) - 48y(t)^7 t^2 \left(\frac{d}{dt} y(t) \right) - 12y(t)^8 t + 144y(t)^5 t^2 \left(\frac{d}{dt} y(t) \right) + 48y(t)^6 t - 192y(t)^3 \left(\frac{d}{dt} y(t) \right) - 576ty(t)^4 - 1152t^2 y(t)^3 \left(\frac{d}{dt} y(t) \right) + 2304ty(t)^2 + 2304t^2 y(t) \left(\frac{d}{dt} y(t) \right) + 4t^3 y(t)^8 + 8t^4 y(t)^7 \left(\frac{d}{dt} y(t) \right) - 640t^3 y(t)^4 - 640t^4 y(t)^3 \left(\frac{d}{dt} y(t) \right) + 6144t^3 y(t)^2 \right)$$

$$\begin{aligned}
& + 3072 t^4 y(t) \left(\frac{d}{dt} y(t) \right) - 15360 t^3 \Big) - \frac{1}{y(t)^8} \left(7 (3 y(t)^8 - 6 y(t)^8 t^2 + 24 y(t)^6 t^2 \right. \\
& - 48 y(t)^4 - 288 t^2 y(t)^4 + 1152 t^2 y(t)^2 + t^4 y(t)^8 - 160 t^4 y(t)^4 + 1536 t^4 y(t)^2 \\
& \left. - 3840 t^4 \right) \left(\frac{d}{dt} y(t) \right) \Big)
\end{aligned}$$

> f[4]:=subs(diff(y(t),t)=f[0],f[4]);

$$\begin{aligned}
f_4 := & \frac{1}{y(t)^7} \left(24 y(t)^7 \left(-t y(t) + \frac{4t}{y(t)} \right) - 48 y(t)^7 t^2 \left(-t y(t) + \frac{4t}{y(t)} \right) - 12 y(t)^8 t \right. \\
& + 144 y(t)^5 t^2 \left(-t y(t) + \frac{4t}{y(t)} \right) + 48 y(t)^6 t - 192 y(t)^3 \left(-t y(t) + \frac{4t}{y(t)} \right) \\
& - 576 t y(t)^4 - 1152 t^2 y(t)^3 \left(-t y(t) + \frac{4t}{y(t)} \right) + 2304 t y(t)^2 + 2304 t^2 y(t) \left(-t y(t) \right. \\
& \left. + \frac{4t}{y(t)} \right) + 4 t^3 y(t)^8 + 8 t^4 y(t)^7 \left(-t y(t) + \frac{4t}{y(t)} \right) - 640 t^3 y(t)^4 - 640 t^4 y(t)^3 \left(\right. \\
& \left. -t y(t) + \frac{4t}{y(t)} \right) + 6144 t^3 y(t)^2 + 3072 t^4 y(t) \left(-t y(t) + \frac{4t}{y(t)} \right) - 15360 t^3 \Big) \\
& - \frac{1}{y(t)^8} \left(7 (3 y(t)^8 - 6 y(t)^8 t^2 + 24 y(t)^6 t^2 - 48 y(t)^4 - 288 t^2 y(t)^4 + 1152 t^2 y(t)^2 \right. \\
& \left. + t^4 y(t)^8 - 160 t^4 y(t)^4 + 1536 t^4 y(t)^2 - 3840 t^4 \right) \left(-t y(t) + \frac{4t}{y(t)} \right) \Big)
\end{aligned}$$

> f[4]:=simplify(f[4]);

$$\begin{aligned}
f_4 := & -\frac{1}{y(t)^9} \left(t (-15360 t^2 y(t)^4 + 38400 t^2 y(t)^2 + 1600 y(t)^6 t^2 - 4 t^4 y(t)^8 + 480 t^4 y(t)^6 \right. \\
& - 9600 t^4 y(t)^4 + 57600 t^4 y(t)^2 - 60 y(t)^8 - 107520 t^4 + 720 y(t)^6 - 2880 y(t)^4 \\
& \left. - 10 y(t)^{10} t^2 + y(t)^{10} t^4 + 15 y(t)^{10} \right)
\end{aligned}$$

We set a value for **h**.

> h:=.25;

$$h := 0.25$$

After replacing **y(t)** with **y**, we rewrite each of the Maple expressions **f[n]** as Maple functions **F[n](t,y)**.

> for i from 0 to 4 do
f[i]:=subs(y(t)=y,f[i]):
F[i]:=unapply(f[i],(t,y));
end do;

$$f_0 := -t y + \frac{4t}{y}$$

$$F_0 := (t, y) \rightarrow -t y + \frac{4t}{y}$$

$$f_1 := \frac{-y^4 + t^2 y^4 + 4 y^2 - 16 t^2}{y^3}$$

$$F_1 := (t, y) \rightarrow \frac{-y^4 + t^2 y^4 + 4y^2 - 16t^2}{y^3}$$

$$f_2 := -\frac{t(-3y^6 + y^6 t^2 - 4t^2 y^4 + 48y^2 + 48t^2 y^2 - 192t^2)}{y^5}$$

$$F_2 := (t, y) \rightarrow -\frac{t(-3y^6 + y^6 t^2 - 4t^2 y^4 + 48y^2 + 48t^2 y^2 - 192t^2)}{y^5}$$

$$f_3 := \frac{1}{y^7} (3y^8 - 6y^8 t^2 + 24y^6 t^2 - 48y^4 - 288t^2 y^4 + 1152t^2 y^2 + t^4 y^8 - 160t^4 y^4 + 1536t^4 y^2 - 3840t^4)$$

$$F_3 := (t, y) \rightarrow \frac{1}{y^7} (3y^8 - 6y^8 t^2 + 24y^6 t^2 - 48y^4 - 288t^2 y^4 + 1152t^2 y^2 + t^4 y^8 - 160t^4 y^4 + 1536t^4 y^2 - 3840t^4)$$

$$f_4 := -\frac{1}{y^9} (t(-15360t^2 y^4 + 38400t^2 y^2 + 1600y^6 t^2 - 4t^4 y^8 + 480t^4 y^6 - 9600t^4 y^4 + 57600t^4 y^2 - 60y^8 - 107520t^4 + 720y^6 - 2880y^4 - 10y^{10} t^2 + y^{10} t^4 + 15y^{10}))$$

$$F_4 := (t, y) \rightarrow -\frac{1}{y^9} (t(-15360t^2 y^4 + 38400t^2 y^2 + 1600y^6 t^2 - 4t^4 y^8 + 480t^4 y^6 - 9600t^4 y^4 + 57600t^4 y^2 - 60y^8 - 107520t^4 + 720y^6 - 2880y^4 - 10y^{10} t^2 + y^{10} t^4 + 15y^{10}))$$

We compute **T2** and **T5** for our chosen orders as Maple functions of T and W.

> T2 := (T, W) -> F[0](T, W) + (h/2) * F[1](T, W);

$$T2 := (T, W) \rightarrow F_0(T, W) + \frac{1}{2} h F_1(T, W)$$

> T5 := (T, W) -> F[0](T, W) + (h/2) * F[1](T, W) + (h^2/3!) * F[2](T, W) + (h^3/4!) * F[3](T, W) + (h^4/5!) * F[4](T, W);

$$T5 := (T, W) \rightarrow F_0(T, W) + \frac{1}{2} h F_1(T, W) + \frac{h^2 F_2(T, W)}{3!} + \frac{h^3 F_3(T, W)}{4!} + \frac{h^4 F_4(T, W)}{5!}$$

We set the initial conditions and take four steps of size **h=.25** for order 2.

> t[0] := 0; w[0] := 1;

$$t_0 := 0$$

$$w_0 := 1$$

**> for i from 0 to 3 do
t[i+1] := t[i] + h;
w[i+1] := w[i] + h * T2(t[i], w[i]);
end do;**

$$t_1 := 0.25$$

$$w_1 := 1.093750000$$

$$t_2 := 0.50$$

$$w_2 := 1.312320929$$

```

t3 := 0.75
w3 := 1.538470504
t4 := 1.00
w4 := 1.720483263

```

We do the same for order 5.

```
> tt[0]:=0;ww[0]:=1;
```

```
tt0 := 0
```

```
ww0 := 1
```

```
> for i from 0 to 3 do
tt[i+1]:=tt[i]+h;
ww[i+1]:=ww[i]+h*T5(tt[i],ww[i]);
end do;
```

```
tt1 := 0.25
```

```
ww1 := 1.086425781
```

```
tt2 := 0.50
```

```
ww2 := 1.289649370
```

```
tt3 := 0.75
```

```
ww3 := 1.513459126
```

```
tt4 := 1.00
```

```
ww4 := 1.701847539
```

The Taylor methods are implemented in Maple by using the options **type=numeric** and **method=taylorseries** in **dsolve**. However, we will not get the same results as above for the same problem because this Maple routine does not give us control over the stepsize. Note that I am using an option that lists the x values for which I want data.

```
> deq:=diff(y(t),t)=-t*y(t)+(4*t)/y(t);
```

$$deq := \frac{d}{dt} y(t) = -ty(t) + \frac{4t}{y(t)}$$

```
> init:=y(0)=1;
```

```
init := y(0) = 1
```

```
> tay:=dsolve({deq, init}, type=numeric,
method=taylorseries,range=0..1,output=array([0,.25,.5,.75,
1]));
```

$$tay := \begin{bmatrix} [t & y(t)] \\ 0. & 1. \\ 0.25 & 1.08708822622174 \\ 0.5 & 1.28980527631485 \\ 0.75 & 1.51348984992225 \\ 1. & 1.70187005275949 \end{bmatrix}$$

Now let's consider the IVP

$$\frac{\partial}{\partial t} y = y - t^2 + 1, y(0) = 0.5 \text{ on } [0,2].$$

We enter the IVP after resetting some variables that have been given values.

```
> y:='y';t:='t';
```

```
y:=y
```

```
t:=t
```

```
> deq:=D(y)(t)=y(t)-t*t+1;
```

```
deq := D(y)(t) = y(t) - t^2 + 1
```

```
> init:=y(0)=0.5;
```

```
init := y(0) = 0.5
```

We first find the exact solution, which we rewrite as a function Y.

```
> soln:=dsolve({deq,init},y(t));
```

```
soln := y(t) = 1 + 2 t + t^2 - 1/2 e^t
```

```
> Y:=unapply(rhs(soln),t);
```

```
Y := t -> 1 + 2 t + t^2 - 1/2 e^t
```

We apply the Taylor series method, with the option **abserr = Float(1,-10)** indicating that we want an absolute error of no more than 1.0×10^{-10} for each step. The default for the global error is $10^{-Digits}$. We can specify an order for the Taylor series used provided it is at least 5.

```
> tay:=dsolve({deq, init}, type=numeric,
method=taylorseries,range=0..2,abserr = Float(1,-10),order=
5);
```

```
tay := proc(x_taylorseries) ... end proc
```

Let's print out a table to see what we get.

```
> printf("t[i]          y(t[i])          w[i]
|y(t[i])-w[i]|\n");
printf("\n");
for i from 0 to 20 do
printf("%4.1f    %20.12f    %20.12f    %20.12f\n",eval(t,tay
(.1*i)),Y(eval(t,tay(.1*i))),eval(y(t),tay(.1*i)),abs(Y(eval(t,tay
(.1*i)))-eval(y(t),tay(.1*i))));
od;
```

```
t[i]          y(t[i])          w[i]          |y(t
[i])-w[i]|
0.0          0.500000000000    0.500000000000
```

```

0.000000000000
0.1      0.657414541000      0.657414540965
0.000000000000
0.2      0.829298621000      0.829298620929
0.000000000100
0.3      1.015070596000      1.015070596227
0.000000000000
0.4      1.214087651000      1.214087651201
0.000000000000
0.5      1.425639364000      1.425639364680
0.000000001000
0.6      1.648940600000      1.648940599844
0.000000000000
0.7      1.883123646000      1.883123646314
0.000000000000
0.8      2.127229536000      2.127229535814
0.000000000000
0.9      2.380198444000      2.380198444493
0.000000000000
1.0      2.640859086000      2.640859085856
0.000000000000
1.1      2.907916988000      2.907916988127
0.000000000000
1.2      3.179941538000      3.179941538748
0.000000001000
1.3      3.455351666000      3.455351666324
0.000000000000
1.4      3.732400016000      3.732400016732
0.000000001000
1.5      4.009155465000      4.009155465006
0.000000000000
1.6      4.283483788000      4.283483788002
0.000000000000
1.7      4.553026304000      4.553026304363
0.000000000000
1.8      4.815176268000      4.815176268049
0.000000000000
1.9      5.067052779000      5.067052779149
0.000000000000
2.0      5.305471950000      5.305471950859
0.000000001000

```

We see that we get the very accurate results we wanted. Let's check $t=1.27$. First the actual value.

```

> Y(1.27);
3.372473719

```

Now the Taylor method value.

```

> eval(y(t), tay(1.27));
3.37247371895055

```

The following clearly shows that linear interpolation is not used.

```

> eval(y(t), tay(1.2)) + .7 * (eval(y(t), tay(1.3)) - eval(y(t), tay(1.2)));
3.372728628

```

If we do not specify an order, Maple uses $\max(22, \text{trunc}(3 * \text{Digits}/2))$. Since we are using the default of 10 for Digits, the order will be 22 here.

```

> tay:=dsolve({deq, init}, type=numeric,
method=taylorseries, range=0..2, abserr = Float(1, -10));
tay := proc(x_taylorseries) ... end proc

```

Let's print out a table to see what we get.

```

> printf("t[i]          y(t[i])          w[i]
        |y(t[i])-w[i]|\n");
printf("\n");
for i from 0 to 20 do
printf("%4.1f    %20.12f    %20.12f    %20.12f\n",eval(t,tay
(.1*i)),Y(eval(t,tay(.1*i))),eval(y(t),tay(.1*i)),abs(Y(eval(t,tay
(.1*i)))-eval(y(t),tay(.1*i))));
od;

```

t[i] [i])-w[i]	y(t[i])	w[i]	y(t
0.0	0.500000000000	0.500000000000	
0.000000000000			
0.1	0.657414541000	0.657414540962	
0.000000000000			
0.2	0.829298621000	0.829298620920	
0.000000000100			
0.3	1.015070596000	1.015070596212	
0.000000000000			
0.4	1.214087651000	1.214087651179	
0.000000000000			
0.5	1.425639364000	1.425639364650	
0.000000001000			
0.6	1.648940600000	1.648940599805	
0.000000000000			
0.7	1.883123646000	1.883123646265	
0.000000000000			
0.8	2.127229536000	2.127229535754	
0.000000000000			
0.9	2.380198444000	2.380198444422	
0.000000000000			
1.0	2.640859086000	2.640859085770	
0.000000000000			
1.1	2.907916988000	2.907916988027	
0.000000000000			
1.2	3.179941538000	3.179941538632	
0.000000001000			
1.3	3.455351666000	3.455351666190	
0.000000000000			
1.4	3.732400016000	3.732400016578	
0.000000001000			
1.5	4.009155465000	4.009155464831	
0.000000000000			
1.6	4.283483788000	4.283483787802	
0.000000000000			
1.7	4.553026304000	4.553026304136	
0.000000000000			
1.8	4.815176268000	4.815176267794	
0.000000000000			
1.9	5.067052779000	5.067052778860	
0.000000000000			
2.0	5.305471950000	5.305471950535	
0.000000001000			

We see that the order makes no difference here. Raising **Digits** eliminates some round-off error.

```
> Digits:=14;
```

```
Digits := 14
```

```
> tay:=dsolve({deq, init}, type=numeric,
method=taylorseries,range=0..2,abserr = Float(1,-10));
tay := proc(x_taylorseries) ... end proc
```

Let's print out a table to see what we get.

```
> printf("t[i]          y(t[i])          w[i]\n");
printf("\n");
for i from 0 to 20 do
printf("%4.1f      %20.12f          %20.12f          %20.12f\n",eval(t,tay
(.1*i)),Y(eval(t,tay(.1*i))),eval(y(t),tay(.1*i)),abs(Y(eval(t,tay
(.1*i)))-eval(y(t),tay(.1*i))));
od;
t[i]          y(t[i])          w[i]          |y(t
[i])-w[i]|
0.0          0.500000000000          0.500000000000
0.000000000000
0.1          0.657414540962          0.657414540962
0.000000000000
0.2          0.829298620920          0.829298620920
0.000000000000
0.3          1.015070596212          1.015070596212
0.000000000000
0.4          1.214087651179          1.214087651179
0.000000000000
0.5          1.425639364650          1.425639364650
0.000000000000
0.6          1.648940599805          1.648940599805
0.000000000000
0.7          1.883123646265          1.883123646265
0.000000000000
0.8          2.127229535754          2.127229535754
0.000000000000
0.9          2.380198444422          2.380198444422
0.000000000000
1.0          2.640859085770          2.640859085770
0.000000000000
1.1          2.907916988027          2.907916988027
0.000000000000
1.2          3.179941538632          3.179941538632
0.000000000000
1.3          3.455351666190          3.455351666190
0.000000000000
1.4          3.732400016578          3.732400016578
0.000000000000
1.5          4.009155464831          4.009155464831
0.000000000000
1.6          4.283483787802          4.283483787802
0.000000000000
1.7          4.553026304136          4.553026304136
0.000000000000
1.8          4.815176267794          4.815176267794
0.000000000000
1.9          5.067052778860          5.067052778860
0.000000000000
2.0          5.305471950535          5.305471950535
0.000000000000
```

NumericalAnalysis

```
> with(Student[NumericalAnalysis]);
[AbsoluteError, AdamsBashforth, AdamsBashforthMoulton, AdamsMoulton, AdaptiveQuadrature,
AddPoint, ApproximateExactUpperBound, ApproximateValue, BackSubstitution, BasisFunctions,
Bisection, CubicSpline, DataPoints, Distance, DividedDifferenceTable, Draw, Euler, EulerTutor,
```

ExactValue, FalsePosition, FixedPointIteration, ForwardSubstitution, Function, InitialValueProblem, InitialValueProblemTutor, Interpolant, InterpolantRemainderTerm, IsConvergent, IsMatrixShape, IterativeApproximate, IterativeFormula, IterativeFormulaTutor, LeadingPrincipalSubmatrix, LinearSolve, LinearSystem, MatrixConvergence, MatrixDecomposition, MatrixDecompositionTutor, ModifiedNewton, NevilleTable, Newton, NumberOfSignificantDigits, PolynomialInterpolation, Quadrature, RateOfConvergence, RelativeError, RemainderTerm, Roots, RungeKutta, Secant, SpectralRadius, Steffensen, Taylor, TaylorPolynomial, UpperBoundOfRemainderTerm, VectorLimit]

We will solve the same problem with which we began the worksheet by using the [Taylor](#) command, which is short for [InitialValueProblem](#) with **method=taylor**. We enter the IVP after resetting some variables that have been given values.

```
> y=`y`;t=`t`;Digits:=10;
```

$$y = y$$

$$t = t$$

$$\text{Digits} := 10$$

```
> DE:=diff(y(t),t)=-t*y(t)+(4*t)/y(t);
```

$$DE := \frac{d}{dt} y(t) = -ty(t) + \frac{4t}{y(t)}$$

We first use orders 2 and 5 with 4 steps to go from 0 to 1, with 10 digits returned and output=information so as to compare with our work above. We follow that with the output options of solution, Error, and plot.

```
> Taylor(DE,y(0)=1,t=1,order=2,numsteps=4,digits=10,output=information);
```

t	$y(t)$	[2nd-Order Taylor]	[Error]
0.	1.	1.	0.
0.2500000000	1.087088226	1.093750000	0.006661774
0.5000000000	1.289805276	1.312320929	0.022515653
0.7500000000	1.513489850	1.538470504	0.024980654
1.0000000000	1.701870053	1.720483263	0.018613210

These are exactly the same as the above.

```
> Taylor(DE,y(0)=1,t=1,order=5,numsteps=4,digits=10,output=information);
```

t	$y(t)$	[5th-Order Taylor]	[Error]
0.	1.	1.	0.
0.2500000000	1.087088226	1.086425781	0.000662445
0.5000000000	1.289805276	1.289649370	0.000155906
0.7500000000	1.513489850	1.513459126	0.000030724
1.0000000000	1.701870053	1.701847539	0.000022514

These are also exactly the same as above.

```
> Taylor(DE,y(0)=1,t=1,order=2,numsteps=4,digits=10,output=solution);
```

1.720483263

```
> Taylor(DE,y(0)=1,t=1,order=5,numsteps=4,digits=10,output=solution);
```

1.701847539

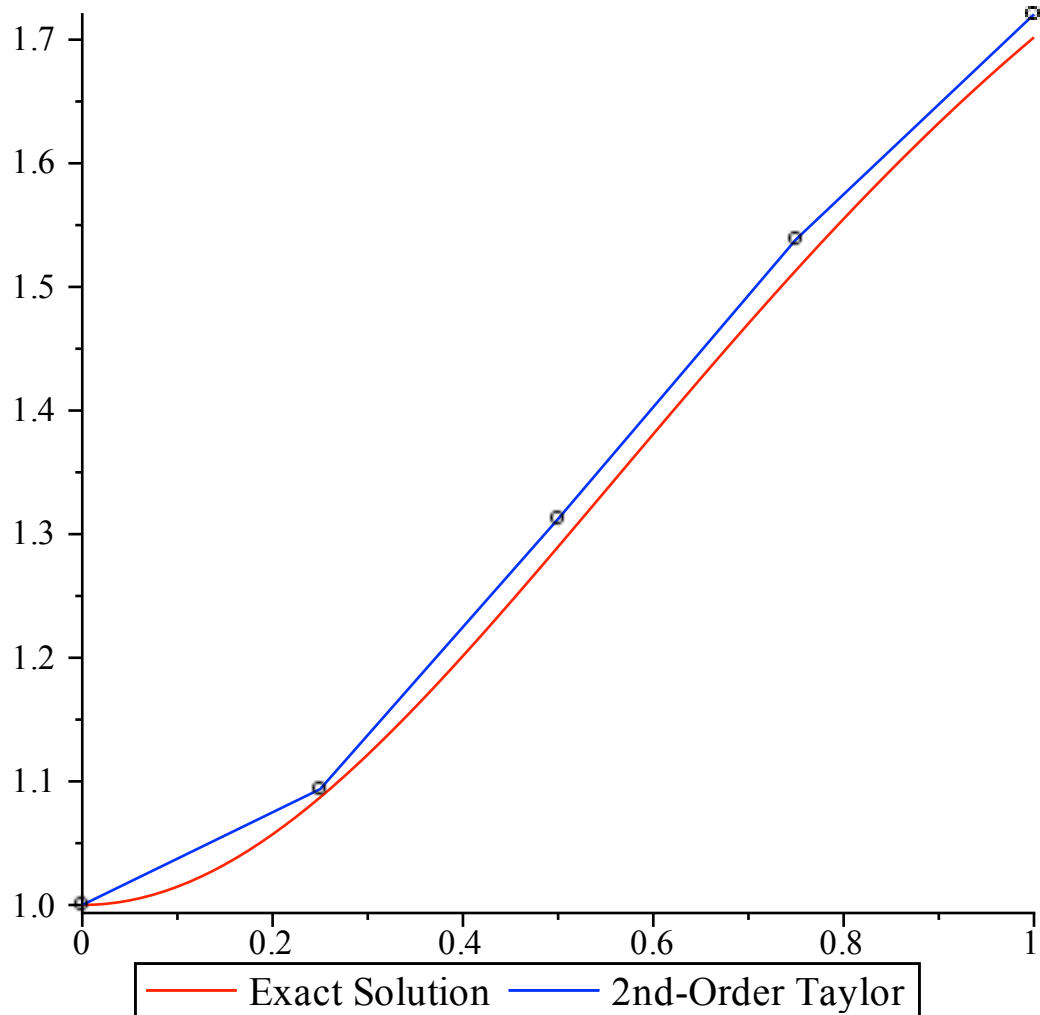
```
> Taylor(DE,y(0)=1,t=1,order=2,numsteps=4,digits=10,output=Error);
```

0.018613210

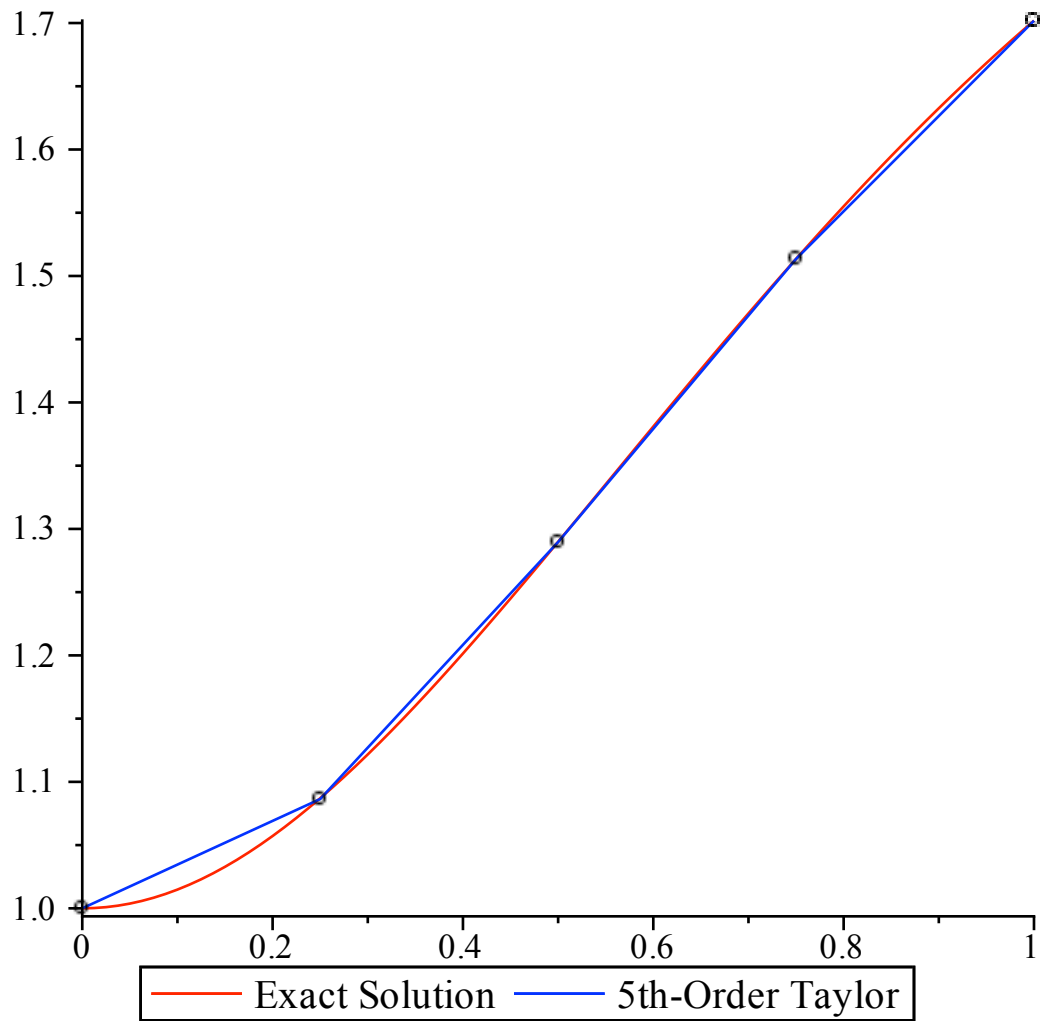
```
> Taylor(DE,y(0)=1,t=1,order=5,numsteps=4,digits=10,output=Error);
```

0.000022514

```
> Taylor(DE,y(0)=1,t=1,order=2,numsteps=4,digits=10,output=plot);
```



```
> Taylor(DE,y(0)=1,t=1,order=5,numsteps=4,digits=10,output=plot);
```



Now we move to a second IVP.

```
> DE:=diff(y(t),t)=y(t)-t*t+1;
```

$$DE := \frac{d}{dt} y(t) = y(t) - t^2 + 1$$

```
> init:=y(0)=0.5;
```

$$init := y(0) = 0.5$$

```
> interface(rtablesize=100);
```

10

```
> Taylor(DE,init,t=2,order=5,numsteps=20,digits=10,output=information);
```

t	$y(t)$	[5th-Order Taylor]	[Error]
0.	0.5	0.5	0.
0.1000000000	0.6574145410	0.6574145417	$7. 10^{-10}$
0.2000000000	0.8292986209	0.8292986226	$1.7 10^{-9}$
0.3000000000	1.015070596	1.015070599	$3. 10^{-9}$
0.4000000000	1.214087651	1.214087655	$4. 10^{-9}$
0.5000000000	1.425639365	1.425639370	$5. 10^{-9}$
0.6000000000	1.648940600	1.648940607	$7. 10^{-9}$
0.7000000000	1.883123646	1.883123656	$1.0 10^{-8}$
0.8000000000	2.127229536	2.127229548	$1.2 10^{-8}$
0.9000000000	2.380198444	2.380198460	$1.6 10^{-8}$
1.000000000	2.640859086	2.640859105	$1.9 10^{-8}$
1.100000000	2.907916988	2.907917011	$2.3 10^{-8}$
1.200000000	3.179941539	3.179941566	$2.7 10^{-8}$
1.300000000	3.455351666	3.455351699	$3.3 10^{-8}$
1.400000000	3.732400017	3.732400055	$3.8 10^{-8}$
1.500000000	4.009155465	4.009155510	$4.5 10^{-8}$
1.600000000	4.283483788	4.283483841	$5.3 10^{-8}$
1.700000000	4.553026304	4.553026366	$6.2 10^{-8}$
1.800000000	4.815176268	4.815176340	$7.2 10^{-8}$
1.900000000	5.067052779	5.067052863	$8.4 10^{-8}$
2.000000000	5.305471951	5.305472048	$9.7 10^{-8}$

> Taylor(DE,init,t=2,order=5,numsteps=20,digits=10,output=plot);

