Complex Variables, 1994, Vol. 25, pp. 23–33
Reprints available directly from the publisher
Photocopying permitted by license only

Printed in the United States of America

Taylor Polynomials of Implicit Functions, of Inverse Functions, and of Solutions of Ordinary Differential Equations

WOLFRAM KOEPF

Konrad-Zuse-Zentrum für Informationstechnik, Heilbronner Str. 10, 10711 Berlin

In this note we present a simple and efficient algorithm to calculate Taylor polynomials of implicit functions, of inverse functions, and of solutions of ordinary differential equations. We give implementations in the Computer Algebra system DERIVE that demonstrate the efficiency of the algorithms in practice.

AMS No. 68Q40, 30B10, 68-04, 30-04
Communicated: R. P. Gilbert
(*Received February 24, 1993*)

1. TAYLOR POLYNOMIALS OF EXPLICIT FUNCTIONS

Assume a real function $f : I \to \mathbb{R}$ is n times differentiable in an interval I containing the point x_0. Then the polynomial

$$T_n(f, x, x_0) := \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k \tag{1}$$

is called the Taylor polynomial of order n of f. Often it represents a global approximation converging to $f(x)$ in I when $n \to \infty$. The function

$$f(x) := \begin{cases} e^{-1/x^2} & \text{if} \quad x \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

has Taylor polynomials $T_n(f, x, 0) \equiv 0$ for all $n \in \mathbb{N}$, which shows that this is not always the case. Locally near x_0, however, the Taylor polynomial T_n gives an approximation of f of order $O((x - x_0)^n)$.

If a function f is explicitly given then (1) is an algorithm for an iterative calculation of the Taylor polynomials. We mention that in general, however, (1) has to be replaced by

$$T_n(f, x, x_0) := \sum_{k=0}^{n} \frac{\lim_{x \to x_0} f^{(k)}(x)}{k!}(x - x_0)^k.$$

This is as the functional expression by which f is given may not be valid for $x = x_0$. An example of that type is $f(x) = \sin x/x$ $(x \neq 0)$ at $x_0 = 0$ with

$$f(0) := \lim_{x \to 0} f(x) = 1,$$

$$f'(0) := \lim_{x \to 0} f'(x) = \lim_{x \to 0} \frac{x \cos x - \sin x}{x^2} = 0,$$

$$f''(0) := \lim_{x \to 0} f''(x) = \lim_{x \to 0} -\frac{2x \cos x + (x^2 - 2)\sin x}{x^3} = -\frac{1}{3},$$

and so on. Another example is function (2).

The calculation of Taylor polynomials for explicit functions is implemented in most Computer Algebra systems available. In this note, we use DERIVE [5], a Computer Algebra system that is especially easy to use, running on every IBM compatible personal computer, and on the other hand having strength enough for all computations we'll do. However, the following restrictions apply. If the order n of the Taylor polynomial searched for is too large, DERIVE may fail by reasons of memory overflow or time restrictions. Further DERIVE does not support all special functions that may be of interest.

The DERIVE function TAYLOR(f,x,x0,n) generates the Taylor polynomial of order n of f at the point x_0 with respect to the variable x. For example we get the following calculations of Taylor polynomials at the origin

function	DERIVE input	DERIVE output after Expand
$\frac{\sin x}{x}$	TAYLOR(SIN(x)/x,x,0,8)	$\frac{x^8}{362880} - \frac{x^6}{5040} + \frac{x^4}{120} - \frac{x^2}{6} + 1,$
e^{-1/x^2}	TAYLOR(EXP(-1/x^2),x,0,10)	0,
$\frac{1}{1-x}$	TAYLOR(1/(1-x),x,0,8)	$x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1,$
$\frac{\cos x - 1}{x^2}$	TAYLOR((COS(x)-1)/x^2,x,0,8)	$-\frac{x^8}{3628800} + \frac{x^6}{40320} - \frac{x^4}{720} + \frac{x^2}{24} - \frac{1}{2}.$

We mention that in the case of explicitly given functions also an algorithm (see [1], [2], and [3]) is available with which in many cases one can find a closed form representation of the Taylor series

$$\sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k$$

for f, i.e. a formula for $f^{(k)}(x_0)$ for symbolic k.

2. TAYLOR POLYNOMIALS OF IMPLICIT FUNCTIONS

In applications, functions often are given only implicitly by an equation

$$F(x, y) = 0,$$

where $y = g(x)$ is considered to be a function of the variable x. The implicit function theorem guarantees under certain weak conditions the local existence of such a function g for which $F(x,g(x)) = 0$ in a neighborhood of a given point (x_0, y_0) for which $F(x_0, y_0) = 0$.

A typical example is the equation of the unit circle

$$F(x,y) = x^2 + y^2 - 1 = 0,$$

where for each (x_0, y_0) with $x_0^2 + y_0^2 = 1$ and $y_0 > 0$ the function

$$g_+(x) := \sqrt{1 - x^2},$$

and for each (x_0, y_0) with $x_0^2 + y_0^2 = 1$ and $y_0 < 0$ the function

$$g_-(x) := -\sqrt{1 - x^2}$$

are corresponding explicit functions locally.

Here we present an iterative procedure to generate the Taylor polynomials of an implicitly given function. Differentiating the defining equation

$$F(x,g(x)) = 0$$

by the two-dimensional chain rule, we get

$$\frac{\partial F}{\partial x}(x,y) + \frac{\partial F}{\partial y}(x,y)g'(x)\Big|_{y=g(x)} = 0,$$

and thus

$$g'(x) = -\frac{\dfrac{\partial F}{\partial x}(x,y)}{\dfrac{\partial F}{\partial y}(x,y)} =: F_1(x,y)\Big|_{y=g(x)}.$$

To get the higher derivatives of g successively, we may differentiate now F_1 by the chain rule to produce

$$g''(x) = \frac{\partial F_1}{\partial x}(x,y) + \frac{\partial F_1}{\partial y}(x,y)g'(x)$$

$$= \frac{\partial F_1}{\partial x}(x,y) + \frac{\partial F_1}{\partial y}(x,y)F_1(x,y) =: F_2(x,y)\Big|_{y=g(x)},$$

further

$$g'''(x) = \frac{\partial F_2}{\partial x}(x,y) + \frac{\partial F_2}{\partial y}(x,y)F_1(x,y) =: F_3(x,y)\Big|_{y=g(x)},$$

and so on, inductively. By this procedure we produce a list of the first n derivatives of g in terms of x and y, and by taking the limits for $y \to y_0$ and $x \to x_0$, we may produce the Taylor polynomial of order n.

As an example we consider again

$$F(x,y) = x^2 + y^2 - 1 = 0,$$

with $(x_0, y_0) = (0,1)$. By the above procedure we get

$$g'(x) = -\frac{\frac{\partial F}{\partial x}(x,y)}{\frac{\partial F}{\partial y}(x,y)} = -\frac{2x}{2y} = -\frac{x}{y} =: F_1(x,y),$$

and iteratively

$$g''(x) = \frac{\partial F_1}{\partial x}(x,y) + \frac{\partial F_1}{\partial y}(x,y)F_1(x,y) = -\frac{1}{y} + \frac{x}{y^2} = -\frac{x^2+y^2}{y^3} =: F_2(x,y),$$

$$g'''(x) = \frac{\partial F_2}{\partial x}(x,y) + \frac{\partial F_2}{\partial y}(x,y)F_1(x,y) = -\frac{3x(x^2+y^2)}{y^5} =: F_3(x,y),$$

and

$$g''''(x) = \frac{\partial F_3}{\partial x}(x,y) + \frac{\partial F_3}{\partial y}(x,y)F_1(x,y) = -\frac{3(x^2+y^2)(5x^2+y^2)}{y^7} =: F_4(x,y).$$

The fourth order Taylor polynomial thus is given by

$$T_4(g,x,(0,1)) = y_0 + \sum_{k=1}^{4} \frac{F_k(x_0,y_0)}{k!}x^k = 1 - \frac{x^2}{2} - \frac{x^4}{8}.$$

DERIVE (like other Computer Algebra systems) does not directly support the calculation of Taylor polynomials of implicit functions. On the other hand, we can easily implement our given algorithmic procedure into DERIVE. The DERIVE function

```
IMPLICIT_TAYLOR_AUX(f,x,y,x0,y0,n,aux) :=
y0 + SUM(LIM(LIM(ELEMENT(aux,k_),y,y0),x,x0)/k_!*(x - x0)^k_,k_,1,n)
```

calculates the Taylor polynomial of order n of the function $y = g(x)$ given by the equation $F(x,y) = 0$ using limits where aux represents the list of derivatives g', $g'',\ldots,g^{(n)}$. If yprime is the first derivative, then the derivative list is produced by the command

```
ITERATES(DIF(g_,y)*yprime + DIF(g_,x),g_,yprime,n - 1).
```

Thus the DERIVE function IMPLICIT_TAYLOR(f,x,y,x0,y0,n), given by

```
IMPLICIT_TAYLOR_YPRIME(f,x,y,x0,y0,n,yprime) :=
ITERATES(DIF(g_,y)*yprime + DIF(g_,x),g_,yprime,n - 1))
```

```
IMPLICIT_TAYLOR_AUX(f,x,y,x0,y0,n,
IMPLICIT_TAYLOR(f,x,y,x0,y0,n) :=
IMPLICIT_TAYLOR_YPRIME(f,x,y,x0,y0,n,-DIF(f,x)/DIF(f,y))
```

results in the desired Taylor polynomial where we used (1) to calculate g'. Here is a list of example calculations.

DERIVE input	DERIVE output after Expand
IMPLICIT_TAYLOR(x^2+y^2−1,x,y,0,1,8)	$-\frac{5x^8}{128}-\frac{x^6}{16}-\frac{x^4}{8}-\frac{x^2}{2}+1$,
IMPLICIT_TAYLOR(x+y^3−y,x,y,0,0,8)	$12x^7+3x^5+x^3+x$,
IMPLICIT_TAYLOR(x+y^3−y,x,y,0,1,5)	$-\frac{3x^5}{2}-\frac{105x^4}{128}\quad\frac{x^3}{2}-\frac{3x^2}{8}-\frac{x}{2}+1$,
IMPLICIT_TAYLOR(x+y^3−y,x,y,0,−1,5)	$-\frac{3x^5}{2}+\frac{105x^4}{128}-\frac{x^3}{2}+\frac{3x^2}{8}-\frac{x}{2}-1$,
IMPLICIT_TAYLOR(y*EXP(y)−x,x,y,0,0,5)	$\frac{125x^5}{24}-\frac{8x^4}{3}+\frac{3x^3}{2}-x^2+x$,
IMPLICIT_TAYLOR(x^2*LN(y)+1,x,y,0,0,3)	0.

The last example is equivalent to the explicit example (2). Here the order of taking the iterated limit in IMPLICIT_TAYLOR_AUX is essential as a two dimensional limit does not exist.

We mention that for implicitly given algebraic functions, i.e. if $F(x,y)$ is a polynomial in both x and y, then again, an algorithm [4] is available with which in many cases one can find a closed form representation of the Taylor series.

3. TAYLOR POLYNOMIALS OF INVERSE FUNCTIONS

The local inverse function $g^{-1}(y)$ of $g(x)$ in a neighborhood of $x = 0$ is a special case of an implicit function: It is the local solution of the equation

$$F(x,y) := g(y) - x = 0$$

near $(0,g(0))$. An application of the general algorithm of the last section generates the Taylor polynomial of g^{-1} at the point $g(0)$. This procedure is covered by the DERIVE function

```
INVERSE_TAYLOR(g,y,x,n) := IF(LIM(DIF(g,y),y,0) = 0,
                           "Taylor expansion of inverse does not exist",
                           IMPLICIT_TAYLOR(g−x,x,y,LIM(g,y,0),0,n)
                           )
```

We get for example

DERIVE input	DERIVE output after Expand
INVERSE_TAYLOR(EXP(x) − 1, x, y, 8)	$-\frac{y^8}{8}+\frac{y^7}{7}-\frac{y^6}{6}+\frac{y^5}{5}-\frac{y^4}{4}+\frac{y^3}{3}-\frac{y^2}{2}+y,$
INVERSE_TAYLOR(SQRT(x), x, y, 5)	$y^2,$
INVERSE_TAYLOR(x^2, x, y, 5)	"Taylor expansion of inverse does not exist",
INVERSE_TAYLOR(LN(1 + x), x, y, 5)	$\frac{y^5}{120}+\frac{y^4}{24}+\frac{y^3}{6}+\frac{y^2}{2}+y,$
INVERSE_TAYLOR((#e^x − #e^ − x)/2, x, y, 5)	$\frac{3y^5}{40}\quad\frac{y^3}{6}+y,$
INVERSE_TAYLOR((#e^x + #e^ − x)/2, x, y, 5)	"Taylor expansion of inverse does not exist",
INVERSE_TAYLOR(EXP(x^2 − 1), x, y, 5)	"Taylor expansion of inverse does not exist",
INVERSE_TAYLOR(EXP(SQRT(x)) − 1, x, y, 5)	$-\frac{5y^5}{6}+\frac{11y^4}{12}\quad y^3+y^2,$
INVERSE_TAYLOR(x*EXP(x), x, y, 5)	$\frac{125y^5}{24}-\frac{8y^4}{3}+\frac{3y^3}{2}-y^2+y.$

One may use DERIVE's graphical capabilities to explore the quality of the given approximations by plotting both g and the calculated Taylor polynomials of g^{-1}. We consider the last example. The function $g(x) := xe^x$ has a local minimum at $x = -1$ of value $-e^{-1} \approx -0.367879$. It follows from complex analysis that the radius of convergence of the Taylor series of the inverse function does not exceed e^{-1}. Thus only in the interval $(-e^{-1}, e^{-1})$ can we expect that the Taylor polynomials converge to $g^{-1}(y)$. As an illustration Figure 1 shows the graph of g together with the first five Taylor polynomials $T_1, \ldots, T_5$ of g^{-1} that can be calculated by DERIVE simplifying

```
VECTOR(INVERSE_TAYLOR(x*EXP(x), x, y, k), k, 1, 5).
```

We mention that the calculation of Taylor polynomials for inverse functions is implemented in some Computer Algebra systems, e.g. in MATHEMATICA [6], where the function

```
InverseTaylor[g_, x_, y_, n_] := InverseSeries[Series[g, {x, 0, n}], y]
```

does the job desired. This method of inverting the Taylor polynomial of order n of g to get the Taylor polynomial of order n of g^{-1}, however, is static with respect to the order n. If we decide to calculate the Taylor polynomial of order $n + 1$ later, we have to redo the whole calculation. With our method, in principle it is possible to work dynamically in the order as one may store the derivatives and limits up to order n that are already calculated in memory, i.e. one may work with streams and lazy evaluation.

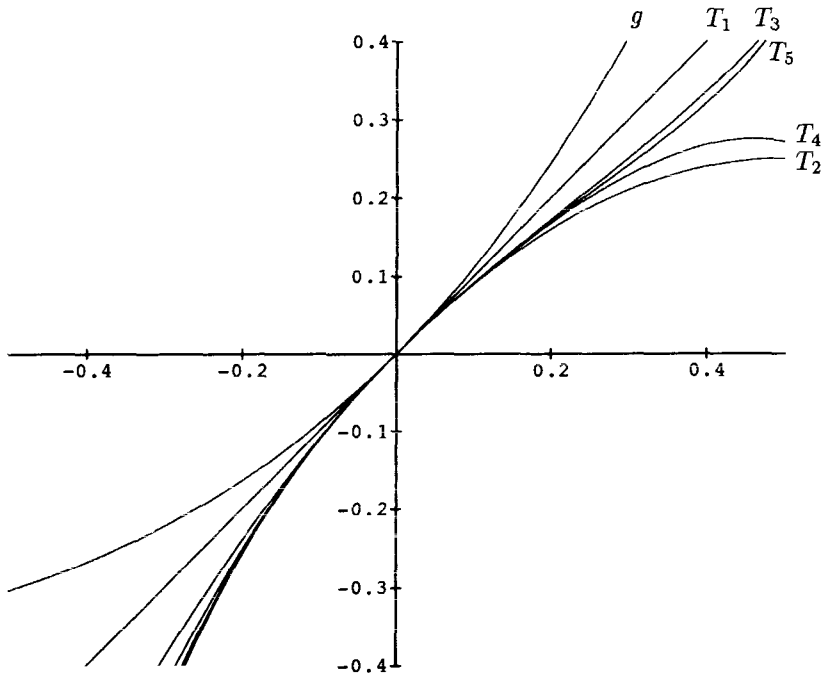

FIGURE 1. The graph of $g(x) = xe^x$ and the first 5 Taylor polynomials of g^{-1}.

4. TAYLOR POLYNOMIAL SOLUTIONS OF FIRST ORDER DIFFERENTIAL EQUATIONS

We consider the solution of an initial value problem given by an explicit first order differential equation

$$y' = F(x, y) \tag{3}$$

and initial data

$$y(x_0) = y_0. \tag{4}$$

Then—if the solution $g(x)$ of (3)–(4) is n times differentiable—we search for its nth Taylor polynomial. A typical situation is an initial value problem (3)–(4) for a complex function g with analytic right hand side F. In this case the solution g is analytic in a neighborhood of the initial point x_0 (we hold on to use the symbol x rather than the usual z). To find its Taylor coefficients, we use the same method as before. Obviously

$$T_n(g, x, x_0) = y_0 + \sum_{k=1}^{n} \frac{g^{(k)}(x_0)}{k!}(x - x_0)^k,$$

and with the derivatives list

$$g'(x) = F(x,y)|_{y=g(x)}$$

$$g''(x) = \frac{\partial F}{\partial x}(x,y) + \frac{\partial F}{\partial y}(x,y)g'(x) = \frac{\partial F}{\partial x}(x,y) + \frac{\partial F}{\partial y}(x,y)F(x,y)\Big|_{y=g(x)} =: F_2(x,y)$$

$$\vdots \qquad \vdots$$

$$g^{(k)}(x) = \frac{\partial F_{k-1}}{\partial x}(x,y) + \frac{\partial F_{k-1}}{\partial y}(x,y)F(x,y)\Big|_{y=g(x)} =: F_k(x,y)$$

an application of the algorithm for implicit functions yields the Taylor polynomial. The DERIVE function

```
DSOLVE1_TAYLOR(f,x,y,x0,y0,n) :=
IMPLICIT_TAYLOR_AUX(f,x,y,x0,y0,n,ITERATES(DIF(g_,y)*f + DIF(g_,x),g_,f,n - 1))
```

uses this approach. We give the following examples.

DERIVE input	DERIVE output after Expand w.r.t. x
DSOLVE1_TAYLOR(y/x,x,y,x0,y0,5)	$\frac{y_0 x}{x_0}$,
DSOLVE1_TAYLOR(x/y,x,y,0,1,10)	$\frac{7x^{10}}{256} - \frac{5x^8}{128} + \frac{x^6}{16} - \frac{x^4}{8} + \frac{x^2}{2} + 1$,
DSOLVE1_TAYLOR(x^2 + y^3,x,y,0,1,5)	$\frac{337x^5}{40} + \frac{37x^4}{8} + \frac{17x^3}{6} + \frac{3x^2}{2} + x + 1$,
DSOLVE1_TAYLOR(y,x,y,0,1,5)	$\frac{x^5}{120} + \frac{x^4}{24} + \frac{x^3}{6} + \frac{x^2}{2} + x + 1$,
DSOLVE1_TAYLOR(EXP(x)*y,x,y,0,1,5)	$\frac{13x^5}{30} + \frac{5x^4}{8} + \frac{5x^3}{6} + x^2 + x + 1$,
DSOLVE1_TAYLOR(EXP(x*y),x,y,0,1,5)	$\frac{49x^5}{120} + \frac{5x^4}{12} + \frac{x^3}{2} + \frac{x^2}{2} + x + 1$,
DSOLVE1_TAYLOR(SQRT((1 - y^2)*(1 - k^2*y^2)),x,y,0,0,5)	$\frac{x^5(k^4 + 14k^2 + 1)}{120} - \frac{x^3(k^2 + 1)}{6} + x$.

For the algorithmic construction of the Taylor series solution of homogeneous linear differential equations with polynomial coefficients (of arbitrary order) rather than Taylor polynomial approximations we again refer to [1].

We remark further that the same method can be applied to solve explicit first order systems, with the only difference that here $\partial F_k/\partial y$ then denotes the Jacobian.

5. TAYLOR POLYNOMIAL SOLUTIONS OF HIGHER ORDER DIFFERENTIAL EQUATIONS

In the case of an explicit second order differential equation

$$y'' = F(x, y, y') \tag{5}$$

with initial data

$$y(x_0) = y_0 \qquad \text{and} \qquad y'(x_0) = y_1 \tag{6}$$

the nth order Taylor polynomial of the solution of (5)–(6) has the form

$$T_n(g, x, x_0) = y_0 + y_1(x - x_0) + \sum_{k=2}^{n} \frac{g^{(k)}(x_0)}{k!}(x - x_0)^k,$$

and we can apply the same method as before after calculation of $g^{(k)}(x_0)$ $(k \geq 2)$. Let the given right hand side of (5) $F(x, y, u)$ be a function of the three variables x, y, and u, that is often enough differentiable. Again, we iteratively differentiate the defining equation of g

$$g''(x) = F(x, g(x), g'(x))$$

by the chain rule to get

$$g'''(x) = \frac{\partial F}{\partial x}(x, y, u) + \frac{\partial F}{\partial y}(x, y, u)g'(x) + \frac{\partial F}{\partial u}(x, y, u)g''(x)$$

$$= \frac{\partial F}{\partial x}(x, y, u) + \frac{\partial F}{\partial y}(x, y, u)u + \frac{\partial F}{\partial u}(x, y, u)F(x, y, u) =: F_3(x, y, u)\Big|_{\substack{y=g(x) \\ u=g'(x)}},$$

and iteratively

$$g^{(k)}(x) = \frac{\partial F_{k-1}}{\partial x}(x, y, u) + \frac{\partial F_{k-1}}{\partial y}(x, y, u)u + \frac{\partial F_{k-1}}{\partial u}(x, y, u)F(x, y, u)\Big|_{\substack{y=g(x) \\ u=g'(x)}}.$$

To get $g^{(k)}(x_0)$ $(k \geq 2)$ we have to evaluate $g^{(k)}(x)$ at the point $x = x_0$, i.e. we take the limit for $u \to y_1$, $y \to y_0$, and $x \to x_0$, which yields the result. This is done by the DERIVE function

```
DSOLVE2_TAYLOR(f, x, y, u, x0, y0, y1, n) givenby

DSOLVE2_TAYLOR_AUX(f, x, y, u, x0, y0, y1, n, aux) := y0 + y1*(x - x0) +
SUM(LIM(LIM(LIM(ELEMENT(aux, k_ - 1), u, y1), y, y0), x, x0)/k_!*(x - x0)^k_, k_, 2, n)

DSOLVE2_TAYLOR(f, x, y, u, x0, y0, y1, n) := DSOLVE2_TAYLOR_AUX(f, x, y, u, x0, y0, y1, n,
ITERATES(DIF(g_, u)*f + DIF(g_, y)*u + DIF(g_, x), g_, f, n - 2))
```

We present the following examples

DERIVE input	DERIVE output after Expand
DSOLVE2_TAYLOR(y^3, x, y, u, 0, 1, 0, 10)	$\frac{61x^{10}}{19200} + \frac{7x^8}{640} + \frac{3x^6}{80} + \frac{x^4}{8} + \frac{x^2}{2} + 1,$
DSOLVE2_TAYLOR(−y, x, y, u, 0, 0, 1, 10)	$\frac{x^9}{362880} - \frac{x^7}{5040} + \frac{x^5}{120} - \frac{x^3}{6} + x,$
DSOLVE2_TAYLOR(−y, x, y, u, 0, 1, 0, 8)	$\frac{x^8}{40320} - \frac{x^6}{720} + \frac{x^4}{24} - \frac{x^2}{2} + 1,$
DSOLVE2_TAYLOR(2*x*u + x^2*y + 3*x, x, y, u, 0, 0, 1, 10)	$\frac{71x^9}{3240} + \frac{23x^7}{252} + \frac{3x^5}{10} + \frac{5x^3}{6} + x,$
DSOLVE2_TAYLOR(EXP(u)*y^2 − SIN(x), x, y, u, 0, 0, 1, 5)	$\frac{ex^4}{12} + \frac{x^5}{120} - \frac{x^3}{6} + x.$

This method—as in the first order case—works independent of the linearity of the differential equation. For the linear differential equation

$$y''(x) + p(x)y'(x) + q(x)y(x) = r(x)$$

with initial conditions

$$y(x_0) = y_0 \quad \text{and} \quad y'(x_0) = y_1$$

the DERIVE function

```
DSOLVE2_LINEAR_TAYLOR(p, q, r, x, y, x0, y0, y1, n) :=
DSOLVE2_TAYLOR(−p*u − q*y + r, x, y, u, x0, y0, y1, n)
```

gives the Taylor polynomial solution of order n. The following are some examples

DERIVE input	DERIVE output after Expand
DSOLVE2_LINEAR_TAYLOR(2*x, x^2, 3*x, x, y, 0, 0, 1, 10)	$\frac{71x^9}{3240} + \frac{23x^7}{252} + \frac{3x^5}{10} + \frac{5x^3}{6} + x,$
DSOLVE2_LINEAR_TAYLOR(SIN(x), COS(x), 0, x, y, 0, 1, 2, 5)	$\frac{x^5}{15} + \frac{x^4}{12} + \frac{2x^3}{3} + \frac{x^2}{2} + 2x + 1,$
DSOLVE2_LINEAR_TAYLOR(1, 1, SIN(x), x, y, 0, 0, 1, 6)	$\frac{x^6}{72} + \frac{x^5}{20} + \frac{x^4}{6} + \frac{x^3}{2} + \frac{x^2}{2} + x.$

We remark that there is an obvious generalization of the given technique to explicit differential equations of higher order ($m \in \mathbb{N}$)

$$y^{(m)} = F(x, y, y', \ldots, y^{(m-1)})$$

with initial data

$$y(x_0) = y_0, y'(x_0) = y_1, \ldots, y^{(m-1)}(x_0) = y_{m-1}.$$

References

[1] W. Koepf, Power series in computer algebra, *Journal of Symbolic Computation* **13** (1992), 581–603.

[2] W. Koepf, Algorithmic development of power series. In: Artificial intelligence and symbolic mathematical computing, ed. by J. Calmet and J. A. Campbell, International Conference AISMC-1, Karlsruhe, Germany, August 1992, Proceedings, Lecture Notes in Computer Science 737, Springer, Berlin-Heilelberg, 1993, 195–213.

[3] W. Koepf, Examples for the algorithmic calculation of formal Puiseux, Laurent and power series, *SIGSAM Bulletin* **27** (1993), 20–32.

[4] W. Koepf, A new algorithm for the development of algebraic functions in Puiseux series, Preprint B-92-21 Fachbereich Mathematik der Freien Universität Berlin, 1992.

[5] A. Rich, J. Rich and D. Stoutemyer, DERIVE User Manual, Version 2, Soft Warehouse, Inc., 3660 Waialae Avenue, Suite 304, Honolulu, HI, 96816-3236.

[6] St. Wolfram, Mathematica. A system for doing mathematics by Computer. Addison-Wesley Publ. Comp., Redwood City, CA, 1991.