Deriving Runge-Kutta-Heun parameters

We are trying to solve an initial value problem of the form $\frac{dx}{dt} = f(t,x)$ with with given initial value $x(0)$ at $t = 0$. Here we assume that f is a differentiable function of two variables.

```
var('t,y')
x = function('x',t)
f = function('f',t,y)
```

The Runge-Kutta-Heun method gives an approximate numerical solution to this equation. It works when f is a function that is differentiable upto sufficiently many orders. In this case, we will work with the order 4 method. To do this we need to work with the Taylor series of f upto terms of order 3.

```
g = taylor(f,(t,0),(y,x(t=0)),3)
var('P,Q,R,S,T,U,V,W,X,Y')
vals = [ f(t=0,y=x(t=0)) == P
       ,diff(f,t)(t=0,y=x(t=0)) == Q
       ,diff(f,y)(t=0,y=x(t=0)) == R
       ,diff(f,t,t)(t=0,y=x(t=0)) == S
       ,diff(f,t,y)(t=0,y=x(t=0)) == T
       ,diff(f,y,y)(t=0,y=x(t=0)) == U
       ,diff(f,t,t,t)(t=0,y=x(t=0)) == V
       ,diff(f,t,t,y)(t=0,y=x(t=0)) == W
       ,diff(f,t,y,y)(t=0,y=x(t=0)) == X
       ,diff(f,y,y,y)(t=0,y=x(t=0)) == Y
       ]
```

We will use h as the time interval for which we will do the numerical approximation. We will ignore all terms of order h^n, where n is at least 4. Since we will only be taking cubic powers of various terms, the highest power of h we might encounter is 13.

```
var('h')
hrel = [(h^p == 0) for p in range(4,13)]
```

The RK4 method depends on now taking the linear combination of $x(0)$ with the values of f at 4 points. The first is the starting point $k_1 = f(0, x(0))$

```
k1 = g.substitute({t:0,y:x(t=0)})
```

The second point is obtained by predicting the position at $t = Ah$ (for some A to be determined later). We thus use the value

$$k_2 = f(Ah, x(0) + a_1 h k_1)$$

```
var('A,a1')
k2 = expand(g.substitute({t:A*h,y:x(t=0)+a1*h*k1}))
k2 = k2.substitute(hrel)
```

The third point is obtained similarly at $t = Bh$ using the value

$$k_3 = f(Bh, x(0) + b_1 h k_1 + b_2 h k_2)$$

```
var('B,b1,b2')
k3 = expand(g.substitute({t:B*h,y:x(t=0)+b1*h*k1+b2*h*k2}))
k3 = k3.substitute(hrel)
```

The fourth and final point is obtained similarly at $t = Ch$ using the value

$$k_4 = f(Ch, x(0) + c_1 h k_1 + c_2 h k_2 + c_3 h k_3)$$

```
var('C,c1,c2,c3')
k4 =
expand(g.substitute({t:C*h,y:x(t=0)+c1*h*k1+c2*h*k2+c3*h*k3}))
k4 = k4.substitute(hrel)
```

We now compute the predicted translation from $x(0)$ of the points. We then break this up into terms depending on the powers of h to various orders.

```
var('a,b,c,d')
rhs = a*k1+b*k2+c*k3+d*k4
rhs = rhs.substitute(vals)
rhsl = []
for i in range(4):
    rhsl.append(rhs.coefficient(h,i))
```

This computes the "right-hand-side" of the required expression. On the other hand we would like this to match the Taylor series of $x(h)$ upto terms of order 4.

```
f1 = f(t=t,y=x)
eqn = diff(x,t) == f1
```

We use the differential equation to compute the i-th Taylor coefficients x_i in terms of values of f and its derivatives at $(0, x(0))$.

```
x1 = f1
x2 = diff(x1,t).substitute(eqn)
x3 = diff(x2,t).substitute(eqn)
x4 = diff(x3,t).substitute(eqn)
```

We now compute the Taylor coefficients in expanded form.

```
lhsl = [x1, x2/2, x3/6, x4/24]
lhsl = map(lambda e: (e(t=0)).substitute(vals), lhsl)
lhsl = map(expand, lhsl)
```

We now need to extract the coefficients of various monomials in P, Q, R, dots, as defined above. First some supplementary functions that find out which monomials occur in which terms.

```
def non_numeric(op):
    return not(op.is_numeric())
```

```
def extract_monom(term):
    return prod(filter(non_numeric,term.operands()))
```

```
def monomial_list(expr):
    l = expr.operands()
    if sum(l)==expr:
        return map(extract_monom,l)
    else:
        return [expr]
```

We can now produce the list of monomials which occur.

```
monoms = map(monomial_list,lhsl)
```

These are the terms that need to vanish

```
fleqns = [t[0]-t[1] for t in zip(rhsl,lhsl)]
```

We extract equations in the RKH parameters by extracting the coefficients of each of the monomials that occur. We also do a check to see that we have not missed anything.

```
check = [[(m,fleqns[i].coefficient(m,1)) for m in monoms[i]] for
i in range(4)]
recheck = [sum(expand(t[0]*t[1]) for t in l) for l in check]
rerecheck = [expand(t[0]-t[1]) for t in zip(fleqns,recheck)]
rerecheck
```

```
[0, 0, 0, 0]
```

If all is well, this last list should be made of 0's (at least after an additional *expand*).

Finally, we have the list of expressions in the RKH parameters that must vanish in order the we have good approximation to order 4.

```
rkheqns = [[t[1] for t in l] for l in check]
for l in rkheqns:
    for t in l:
        view(t == 0)
```

$$a + b + c + d - 1 = 0$$
$$a_1 b + (b_1 + b_2)c + (c_1 + c_2 + c_3)d - \frac{1}{2} = 0$$
$$Ab + Bc + Cd - \frac{1}{2} = 0$$
$$a_1 b_2 c + (a_1 c_2 + b_1 c_3 + b_2 c_3)d - \frac{1}{6} = 0$$
$$\frac{1}{2}\, a_1^2 b + \frac{1}{2}\,\left(b_1^2 + 2\,b_1 b_2 + b_2^2\right)c + \frac{1}{2}\,\left(c_1^2 + 2\,c_1 c_2 + c_2^2 + 2\,c_1 c_3 + 2\,c_2 c_3 +\right.$$
$$Ab_2 c + (Ac_2 + Bc_3)d - \frac{1}{6} = 0$$
$$Aa_1 b + (Bb_1 + Bb_2)c + (Cc_1 + Cc_2 + Cc_3)d - \frac{1}{3} = 0$$
$$\frac{1}{2}\,A^2 b + \frac{1}{2}\,B^2 c + \frac{1}{2}\,C^2 d - \frac{1}{6} = 0$$
$$a_1 b_2 c_3 d - \frac{1}{24} = 0$$
$$\frac{1}{2}\,\left(a_1^2 b_2 + 2\,a_1 b_1 b_2 + 2\,a_1 b_2^2\right)c + \frac{1}{2}\,\left(a_1^2 c_2 + 2\,a_1 c_1 c_2 + 2\,a_1 c_2^2 + b_1^2 c_3 + 2\right.$$
$$\frac{1}{6}\,a_1^3 b + \frac{1}{6}\,\left(b_1^3 + 3\,b_1^2 b_2 + 3\,b_1 b_2^2 + b_2^3\right)c + \frac{1}{6}\,\left(c_1^3 + 3\,c_1^2 c_2 + 3\,c_1 c_2^2 + c_2^3 +\right.$$
$$Ab_2 c_3 d - \frac{1}{24} = 0$$
$$(Aa_1 b_2 + Ba_1 b_2)c + (Aa_1 c_2 + Ca_1 c_2 + Bb_1 c_3 + Cb_1 c_3 + Bb_2 c_3 + Cb_2$$

$\left(Ab_1b_2 + Ab_2^2\right)c + \left(Ac_1c_2 + Ac_2^2 + Bc_1c_3 + Ac_2c_3 + Bc_2c_3 + Bc_3^2\right)d -$
$\frac{1}{2}\,Aa_1^2b + \frac{1}{2}\left(Bb_1^2 + 2\,Bb_1b_2 + Bb_2^2\right)c + \frac{1}{2}\left(Cc_1^2 + 2\,Cc_1c_2 + Cc_2^2 + 2\,C\right.$
$\frac{1}{2}\,A^2b_2c + \frac{1}{2}\left(A^2c_2 + B^2c_3\right)d - \frac{1}{24} = 0$
$ABb_2c + (ACc_2 + BCc_3)d - \frac{1}{8} = 0$
$\frac{1}{2}\,A^2a_1b + \frac{1}{2}\left(B^2b_1 + B^2b_2\right)c + \frac{1}{2}\left(C^2c_1 + C^2c_2 + C^2c_3\right)d - \frac{1}{8} = 0$
$\frac{1}{6}\,A^3b + \frac{1}{6}\,B^3c + \frac{1}{6}\,C^3d - \frac{1}{24} = 0$

One of the "standard" solutions to this system of equations is given by taking the values of f at points as follows

$$k_2 = f(\frac{1}{2}h, x(0) + \frac{1}{2}hk_1); k_3 = f(\frac{1}{2}h, x(0) + \frac{1}{2}hk_2); k_4 = f(h, x(0) + hk_3)$$

These are combined in the form

$$x = x(0) + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

In other words, we are using the following dictionary for the parameters:

```
sold={A:1/2, a1: 1/2, B:1/2, b1:0, b2: 1/2, C:1, c1:0, c2: 0,
c3:1, a: 1/6, b:2/6, c:2/6, d: 1/6}
```

We can check that these satisfy the equations:

```
[[t.substitute(sold) for t in l] for l in rkheqns]
```

```
[[0], [0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

If all is well we should get only 0's!