```
> restart;
  Digits:=16;
```

$$Digits := 16$$

```
> with(numapprox);
```

[*chebdeg*, *chebmult*, *chebpade*, *chebsort*, *chebyshev*, *confracform*, *hermite_pade*, *hornerform*, *infnorm*, *laurent*, *minimax*, *pade*, *remez*, *taylor*]

```
> Int(sqrt(1+x^4),x=0..t);
```

$$\int_0^t \sqrt{1+x^4}\,dx$$

S tímhle integrálem si ani Maple ani matematika tak docela neporadí. Výsledek je totiž reáná funkce daná koplexním výrazem.

```
> W:=int(sqrt(1+x^4),x=0..t);
```

$$W := \left(\left(\frac{1}{12}-\frac{1}{12}I\right)\left(-4\sqrt{1-It^2}\sqrt{1+It^2}\,\text{EllipticF}\left(-\frac{1}{2}\sqrt{2}\,t-\frac{1}{2}I\sqrt{2}\,t, I\right)+I\sqrt{2}\,t+\sqrt{2}\,t+\sqrt{2}\,t^5+I\sqrt{2}\,t^5\right)\sqrt{2}\right)\Big/\sqrt{1+t^4}$$

Pokud bychom chtěli tuto hodnotu někde mimo Maple numericky počítat, museli bychom
1) sehnat knihovnu pro komplexní aritmetiku včetně eliptických funkcí
2) Počítat integrál přímo numerickou kvadraturou
3) Poprosit Maple aby nám vrátil aproximaci téhle funkce vhodnou pro další počítání

```
>
```

```
> W1:=expand(series(W,t,50));
```

Nejdříve zkusíme rozvoj (ale z tvaru W vidíme, že nemůžeme očekávat konvergenci pro |t|>1)

$$W1 := t+\frac{1}{10}t^5-\frac{1}{72}t^9+\frac{1}{208}t^{13}-\frac{5}{2176}t^{17}+\frac{1}{768}t^{21}-\frac{21}{25600}t^{25}+\frac{33}{59392}t^{29}-\frac{13}{32768}t^{33}+\frac{715}{2424832}t^{37}-\frac{2431}{10747904}t^{41}+\frac{4199}{23592960}t^{45}-\frac{4199}{29360128}t^{49}+O(t^{51})$$

```
> W2:=convert(W1,polynom):
```

Máme ale možnost konvertovat Taylor ---> Pade

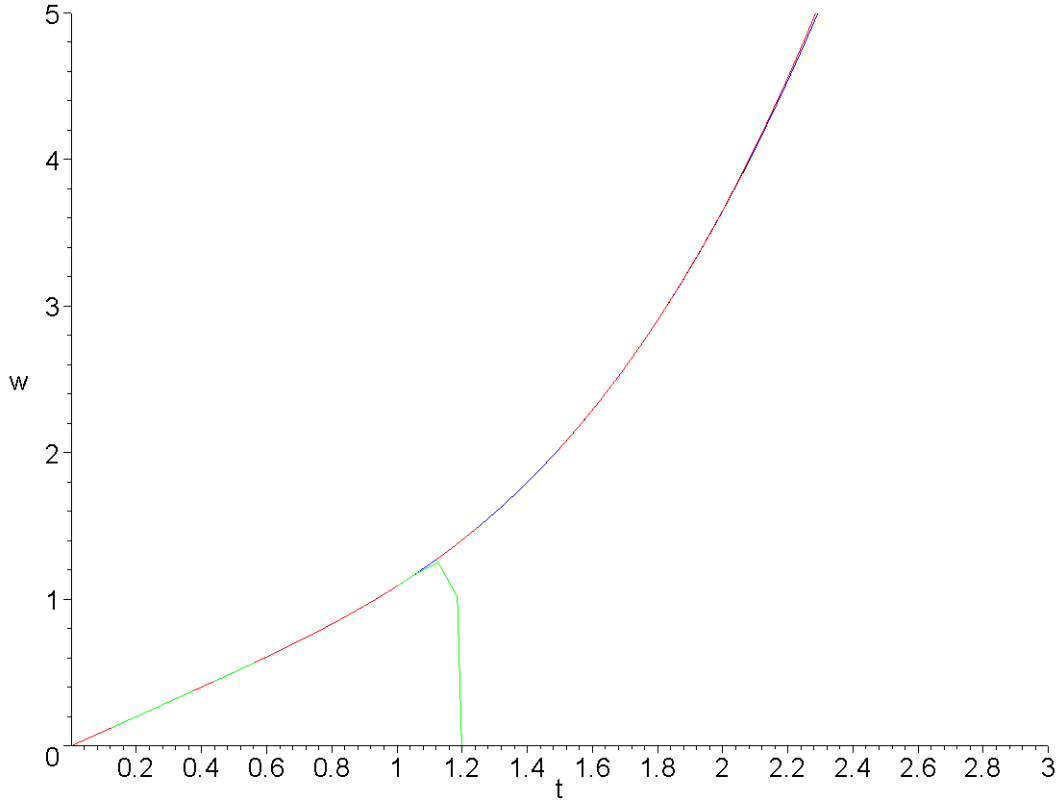
```
> W3:=convert(W1,ratpoly);
```

$$W3 := \left(\frac{8735911911935879245355997235 99}{16322337763153204217899597966336 00}t^{25}+\frac{16862844361743658792301746419401}{77502962413317111062233608274636 8}t^{21}+\frac{31065543320804399704015895560 4585}{12594231392164030547612961344628 48}t^{17}+\frac{35690148321297244920106042 53}{30899113309790257285748889 44}t^{13}+\frac{13250421808328828544055385 9}{52174862999645837818337712}t^9+\frac{18853942420101070656844871}{72465087499508108081024 60}t^5+t\right)\Big/\left(1+\frac{36258583090211979152069 25}{14493017499901621616204 92}t^4\right)$$

$$+\frac{120175779045807054026619175}{52174862999645837818337712}t^{8}+\frac{204218135530335086411306087 5}{213916938298547935055184619 2}t^{12}$$
$$+\frac{122037160689891208977167579 25}{70354904151522431973705163648}t^{16}+\frac{186475895834677755841442828 05}{16885176996365383673689239275 52}t^{20}$$
$$+\frac{20686389381206793163735698025}{19586805315783845061479517559 6032}t^{24}\Bigg)$$

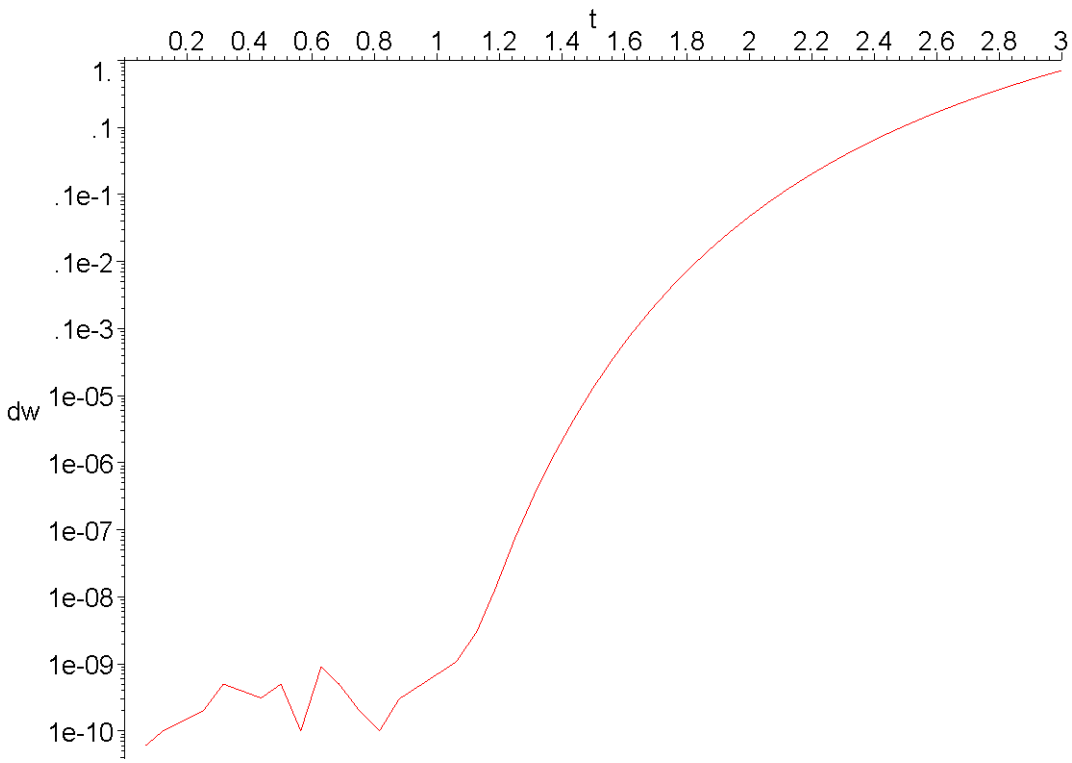
```
> plot([evalf(W),W2,W3],t=0..3,w=0..5,color=[red,green,blue]);
```

Padeova aproximace vypadá použitelně i za t>1




```
> plots[logplot](abs(evalf(W)-W3),t=0..3,dw);
```

Chyba je tam ale velká



Druhou možností je zkusit aproximovat funkční hodnoty přímo

```
> X:=[seq(i/5.0,i=0..15)];
  Y:=evalf(map(q->subs(t=q,Re(W)),X),25);
```

$X := [0, .2000000000000000, .4000000000000000, .6000000000000000, .8000000000000000,$
$1.000000000000000, 1.200000000000000, 1.400000000000000, 1.600000000000000,$
$1.800000000000000, 2.000000000000000, 2.200000000000000, 2.400000000000000,$
$2.600000000000000, 2.800000000000000, 3.000000000000000]$

$Y := [0, .2000319928928243413905616, .4010203909858658475116466,$
$.6076419486590373465673711, .8311260060692207653644633,$
$1.089429413224822322411846, 1.404308681954214025042071,$
$1.797919137115700124142342, 2.291172421963997227437939,$
$2.903561424849032475448170, 3.653484493139718780938959,$
$4.558589837401343064114874, 5.636027823454942219360749,$
$6.902617905516293975778385, 8.374956299015460658185159,$
$10.06948564791735090118307]$

```
> readlib(spline);
```

proc(X, Y, z, d) ... **end**

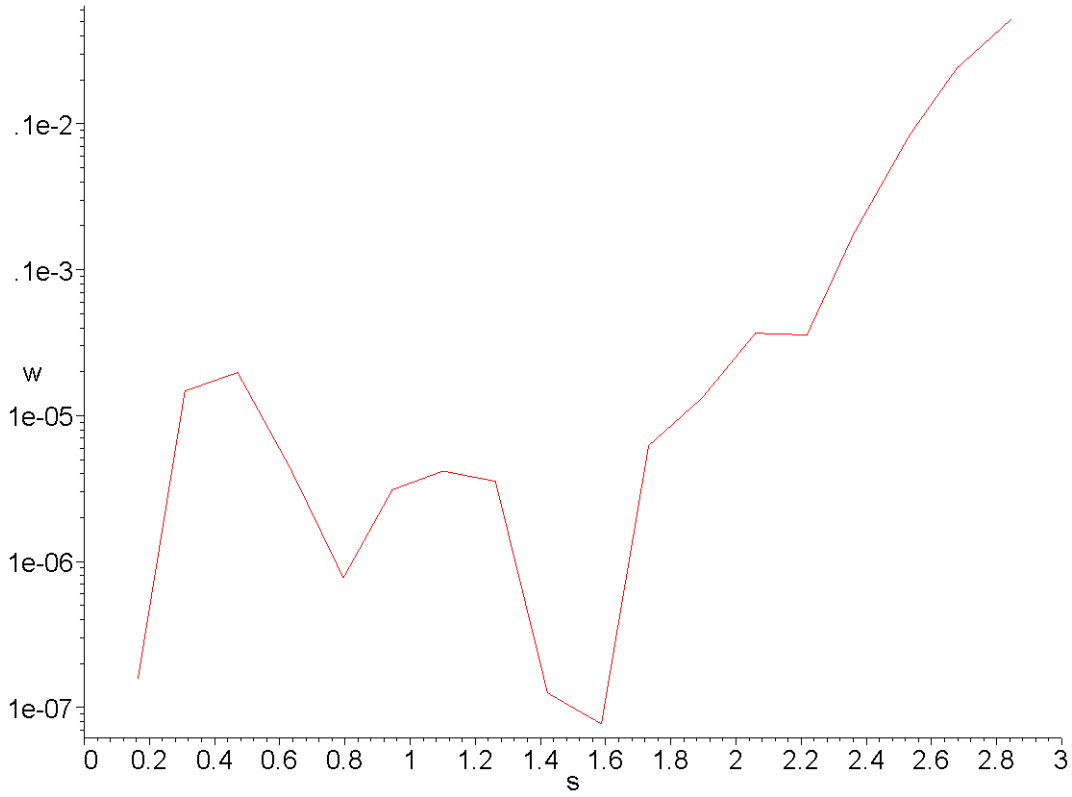
```
> Ws:=spline(X,Y,t):
> Wsf:=proc(s);
   if not type(s,float) then 'Wsf(s)' else evalf(subs(t=s,Ws))
  fi;
  end;
```

Wsf := **proc**(s) **if not** type(s, $float$) **then** 'Wsf(s)' **else** evalf(subs($t = s$, Ws)) **fi end**

Abychom vykrslili chybu splinové aproximace W, je nutné definovat výše uvedenou funkci a i pak

se musí kreslit velmi nepřehledným příkazem,
aby se obešli všechny nástrahy spojené s jedné fce dané po částech a druheé dané numerickou integrací

```
> plots[logplot](abs(evalf(subs(t=s,Re(W)))-Wsf(s)),s=0..3,w,color
  =[red,green,blue],numpoints=20,adaptive=false);
```



Někdy se analytické funkce podají při použití Thieleho vzorce

```
> readlib(thiele);
```

$$\mathbf{proc}(lx, ly, x) \ \ldots \ \mathbf{end}$$

```
> thiele([x1,x2],[y1,y2],t);
  thiele([x1,x2,x3],[y1,y2,y3],t);
  thiele([x1,x2,x3,x4],[y1,y2,y3,y4],t);
```

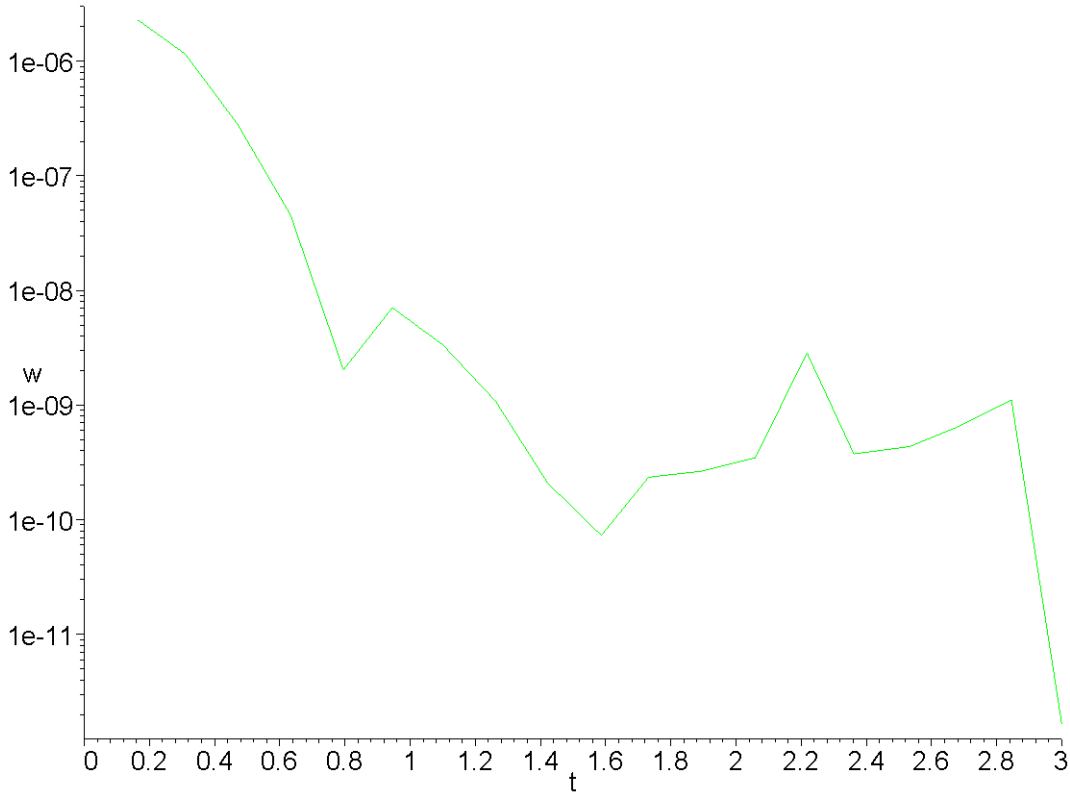
$$y1 + \frac{(t - x1)\,(y1 - y2)}{x1 - x2}$$

$$y1 + \frac{t - x1}{\dfrac{x1 - x2}{y1 - y2} + \dfrac{t - x2}{\dfrac{x1 - x3}{\dfrac{x1 - x2}{y1 - y2} - \dfrac{x2 - x3}{y2 - y3}} + y2 - y1}}$$

$$y1 + (t - x1) \Big/ \left(\frac{x1 - x2}{y1 - y2} + (t - x2) \Big/ \left(\frac{x1 - x3}{\dfrac{x1 - x2}{y1 - y2} - \dfrac{x2 - x3}{y2 - y3}} + y2 - y1 \right. \right.$$

$$+\frac{t-x3}{\dfrac{x1-x4}{\dfrac{x1-x3}{x1-x2}-\dfrac{x2-x3}{y2-y3}}+y2-\dfrac{x2-x4}{\dfrac{x2-x3}{y2-y3}-\dfrac{x3-x4}{y3-y4}}-y3}+\frac{x2-x3}{y2-y3}-\frac{x1-x2}{y1-y2}}\Bigg)\Bigg)$$

```
> Wt:=thiele(X,Y,t):
> plots[logplot](abs(evalf(W)-Wt),t=0..3,w,color=[green],numpoints
  =20,adaptive=false);
```




```
> 
> 
> 
```

Výstup výrazů pro začlenění do jiného programu

```
> fortran(Wt,precision=double);
      t0 = t/(0.9998400611204157D0+(t-0.2D0)/(-0.8387334989536366D2+(t-0
     #.4D0)/(0.38716131687683D-2+(t-0.6D0)/(0.9854485689321408D2+(t-0.8D
     #0)/(-0.517955824630881D-1+(t-0.1D1)/(-0.2066169753305742D2+(t-0.12
     #D1)/(0.1093397160339938D1+(t-0.14D1)/(0.693336416938528D0+(t-0.16D
     #1)/(-0.1642724170156443D1+(t-0.18D1)/(-0.4824618632530334D1+(t-0.2
     #D1)/(-0.4512605207481688D0+(t-0.22D1)/(0.3924851087813006D2+(t-0.2
     #4D1)/(0.4758285892449502D-1+(t-0.26D1)/(0.3332880651777425D5-0.149
     #7673933197205D5*t))))))))))))))
> 
> readlib(C):
  C(Wt,precision=double);
      t0 = t/(0.9998400611204157+(t-0.2)/(-0.8387334989536366E2+(t-0.4)/(
0.38716131687683E-2+(t-0.6)/(0.9854485689321408E2+(t-0.8)/(
-0.517955824630881E-1+(t-0.1E1)/(-0.2066169753305742E2+(t-0.12E1)/(
0.1093397160339938E1+(t-0.14E1)/(0.693336416938528+(t-0.16E1)/(
-0.1642724170156443E1+(t-0.18E1)/(-0.4824618632530334E1+(t-0.2E1)/(
-0.4512605207481688+(t-0.22E1)/(0.392
```

```
0.4758285892449502E-1+(t-0.26E1)/(0.3332880651777425E5-0.1497673933197205E5*t))
))))))))))));
```

```
>
```

```
> fortran(Y);
```
Seznam přímo převést neumí

```
Error, (in fortran) invalid argument, [0, .20003199289282843413905616,
.40102039098586584751116466, .60764194865903734656737 11,
.83112600606922076536446 33, 1.08942941322482232241 1846,
1.40430868195421402504207 1, 1.79791913711570012414 2342,
2.29117242196399722743793 9, 2.90356142484903247544 8170,
3.65348449313971878093895 9, 4.55858983740134306411 4874,
5.63602782345494221936074 9, 6.90261790551629397577 8385,
8.37495629901546065818515 9, 10.06948564791735090118 307]
```

```
> fortran(Z(op(Y)));
```
Nejjednodušší je jeho prvky chápat jako parametry abstraktní funkce

```
    t0 = Z(0.E0,0.200032E0,0.4010204E0,0.6076419E0,0.831126E0,0.108942
   #9E1,0.1404309E1,0.1797919E1,0.2291172E1,0.2903561E1,0.3653484E1,0.
   #455859E1,0.5636028E1,0.6902618E1,0.8374956E1,0.1006949E2)
```

```
> J:=array(Y):
  C(J);
```

```
    J[0] = 0.0;
    J[1] = 0.2000319928928243;
    J[2] = 0.4010203909858658;
    J[3] = 0.6076419486590373;
    J[4] = 0.8311260060692208;
    J[5] = 0.1089429413224822E1;
    J[6] = 0.1404308681954214E1;
    J[7] = 0.17979191371157E1;
    J[8] = 0.2291172421963997E1;
    J[9] = 0.2903561424849032E1;
    J[10] = 0.3653484493139719E1;
    J[11] = 0.4558589837401343E1;
    J[12] = 0.5636027823454942E1;
    J[13] = 0.6902617905516294E1;
    J[14] = 0.8374956299015461E1;
    J[15] = 0.1006948564791735E2;
```

Podobně to lze učinit i s maticemi:

```
> J:=linalg[jacobian]([r*sin(theta)*cos(phi),r*sin(theta)*sin(phi)
  ,r*cos(theta)],[r,theta,phi]);
```

$$J := \begin{bmatrix} \sin(\theta)\cos(\phi) & r\cos(\theta)\cos(\phi) & -r\sin(\theta)\sin(\phi) \\ \sin(\theta)\sin(\phi) & r\cos(\theta)\sin(\phi) & r\sin(\theta)\cos(\phi) \\ \cos(\theta) & -r\sin(\theta) & 0 \end{bmatrix}$$

```
> C(J);
```

```
    J[0][0] = sin(theta)*cos(phi);
    J[0][1] = r*cos(theta)*cos(phi);
    J[0][2] = -r*sin(theta)*sin(phi);
    J[1][0] = sin(theta)*sin(phi);
    J[1][1] = r*cos(theta)*sin(phi);
    J[1][2] = r*sin(theta)*cos(phi);
    J[2][0] = cos(theta);
    J[2][1] = -r*sin(theta);
    J[2][2] = 0.0;
```

Výsledek lze i vylepšit, aby se neprováděly očividně se opakující operace

```
> C(J,optimized);
```

```
    t1 = sin(theta);
    t2 = cos(phi);
    t4 = cos(theta);
    t5 = r*t4;
```

```
      t7 = r*t1;
      t8 = sin(phi);
      J[0][0] = t1*t2;
      J[0][1] = t5*t2;
      J[0][2] = -t7*t8;
      J[1][0] = t1*t8;
      J[1][1] = t5*t8;
      J[1][2] = t7*t2;
      J[2][0] = t4;
      J[2][1] = -t7;
      J[2][2] = 0.0;
```

Lze generovat také funkce včetně hlavičky

```
> Wtf:=(unapply(Wt,x)):
> C(Wtf,ansi);
/* The options were    : operatorarrow */
#include <math.h>
double Wtf(double x)
{
  {
    return(t/(0.9998400611204157+(t-0.2)/(-0.8387334989536366E2+(t-0.4)/(
0.38716131687683E-2+(t-0.6)/(0.9854485689321408E2+(t-0.8)/(
-0.517955824630881E-1+(t-0.1E1)/(-0.2066169753305742E2+(t-0.12E1)/(
0.1093397160339938E1+(t-0.14E1)/(0.693336416938528+(t-0.16E1)/(
-0.1642724170156443E1+(t-0.18E1)/(-0.4824618632530334E1+(t-0.2E1)/(
-0.4512605207481688+(t-0.22E1)/(0.3924851087813006E2+(t-0.24E1)/(
0.4758285892449502E-1+(t-0.26E1)/(0.3332880651777425E5-0.1497673933197205E5*t))
))))))))))));
  }
}
```

```
> C(Wtf,ansi,filename="wt.c");
```
Zapíše výsledek do souboru "wt.c"

```
>
>
>
> restart;
```

Geodetiky na hyperboloidu

Samozřejm, že je lze počítat mnohem snáze, když uvážíme symetrie a zachování normy tečného vektoru.

Jde o příklad použití balíků **tensor** + **plots**

```
> with(tensor);
```

[*Christoffel1*, *Christoffel2*, *Einstein*, *Jacobian*, *Killing_eqns*, *Levi_Civita*, *Lie_diff*, *Ricci*, *Ricciscalar*, *Riemann*, *RiemannF*, *Weyl*, *act*, *antisymmetrize*, *change_basis*, *commutator*, *compare*, *conj*, *connexF*, *contract*, *convertNP*, *cov_diff*, *create*, *d1metric*, *d2metric*, *directional_diff*, *displayGR*, *display_allGR*, *dual*, *entermetric*, *exterior_diff*, *exterior_prod*, *frame*, *geodesic_eqns*, *get_char*, *get_compts*, *get_rank*, *init*, *invars*, *invert*, *lin_com*, *lower*, *npcurve*, *npspin*, *partial_diff*, *permute_indices*, *petrov*, *prod*, *raise*, *symmetrize*, *tensorsGR*, *transform*]

```
> a:=1;
  r:=sqrt(a^2+z^2);
  R:=unapply(r,z);
```

$$a := 1$$

$$r := \sqrt{1+z^2}$$

$$R := z \to \sqrt{1+z^2}$$

```
> ds^2 = r^2 * d(phi)^2 +(1+diff(r,z)^2)*d(z)^2;
```

Jen komentář, nijak se nepoužívá dále

$$ds^2 = (1+z^2)\,\mathrm{d}(\phi)^2 + \left(1+\frac{z^2}{1+z^2}\right)\mathrm{d}(z)^2$$

Jak pracovat s balíkem **tensor** se zjistí z nápovědy (zvolíme nápovědu k objektu který nás zajímá nakonec, **Christofell**,
a v nápovědě jsou pak všechny kroky potřebné k jeho spočtení)

```
> coord:=[z,phi]:
  g_compts:=array(symmetric,sparse,1..2,1..2):
  g_compts[1,1] := 1+z^2/(a^2+z^2):
  g_compts[2,2] := a^2+z^2:
  g := create( [-1,-1], eval(g_compts));
```

$g :=$ table([

$index\_char = [-1, -1]$

$$compts = \begin{bmatrix} 1+\frac{z^2}{1+z^2} & 0 \\ 0 & 1+z^2 \end{bmatrix}$$

])

```
> ginv:= invert (g, 'detg'):
  D1g:= d1metric (g, coord):
  Cf1:= Christoffel1 (D1g):
  Cf2:=Christoffel2 (ginv, Cf1):
```

Budeme řešit rovnici geodetiky, takže bod v němž počítáme konexe je na křivce v místě $[z(s), \phi(s)]$

```
> Cf2s:=subs(z=z(s),phi=phi(s),eval(Cf2)):
  displayGR(Christoffel2,Cf2s);
```

The Christoffel Symbols of the Second Kind

non-zero components :

$$\{1,11\} = \frac{z(s)}{(1+z(s)^2)\,(1+2\,z(s)^2)}$$

$$\{1,22\} = -\frac{(1+z(s)^2)\,z(s)}{1+2\,z(s)^2}$$

$$\{2,12\} = \frac{z(s)}{1+z(s)^2}$$

```
> Poloha:=array([z(s),phi(s)]);
  Rychlost:=map(diff,evalm(Poloha),s);
  Zrychleni:=map(diff,evalm(Rychlost),s);
```

$$Poloha := [z(s), \phi(s)]$$

$$Rychlost := \left[\frac{\partial}{\partial s} z(s), \frac{\partial}{\partial s} \phi(s)\right]$$

$$Zrychleni := \left[\frac{\partial^2}{\partial s^2} z(s), \frac{\partial^2}{\partial s^2} \phi(s)\right]$$

```
> U:=create([+1],evalm(Rychlost));
```
Teď z obyčejného vektoru uděláme kontravariantní objekt

$U :=$ table([

$index\_char = [1]$

$compts = \left[\frac{\partial}{\partial s} z(s), \frac{\partial}{\partial s} \phi(s)\right]$

])

Následující výraz spočte Gamma^k_ij U^i U^j

```
> xdd:=prod(Cf2s,prod(U,U),[2,1],[3,2]);
```

$xdd :=$ table([

$index\_char = [1]$

$$compts = \left[\frac{z(s)\left(\left(\frac{\partial}{\partial s} z(s)\right)^2 - \left(\frac{\partial}{\partial s} \phi(s)\right)^2 - 2\left(\frac{\partial}{\partial s} \phi(s)\right)^2 z(s)^2 - \left(\frac{\partial}{\partial s} \phi(s)\right)^2 z(s)^4\right)}{(1+z(s)^2)(1+2\,z(s)^2)}, \right.$$

$$\left. 2\frac{z(s)\left(\frac{\partial}{\partial s} z(s)\right)\left(\frac{\partial}{\partial s} \phi(s)\right)}{1+z(s)^2}\right]$$

])

```
> DR:=evalm(Zrychleni+rhs(op(op(op(xdd)))[2]));
```
To jsou diferenciální rovnice geodetiky na 2-ploše

$$DR := \left[\left(\frac{\partial^2}{\partial s^2} z(s)\right) + \frac{z(s)\left(\left(\frac{\partial}{\partial s} z(s)\right)^2 - \left(\frac{\partial}{\partial s} \phi(s)\right)^2 - 2\left(\frac{\partial}{\partial s} \phi(s)\right)^2 z(s)^2 - \left(\frac{\partial}{\partial s} \phi(s)\right)^2 z(s)^4\right)}{(1+z(s)^2)(1+2\,z(s)^2)}, \right.$$

$$\left. \left(\frac{\partial^2}{\partial s^2} \phi(s)\right) + 2\frac{z(s)\left(\frac{\partial}{\partial s} z(s)\right)\left(\frac{\partial}{\partial s} \phi(s)\right)}{1+z(s)^2}\right]$$

```
> DRIF:=(p)->{op(convert(DR,set)),
z(0)=0,phi(0)=0,D(z)(0)=sin(p),D(phi)(0)=cos(p)};
```
A knim se musí přidat počáteční podmínky

Využívá se faktu, že na počátku je "jednotková" metrika

$DRIF :=$

$$p \to \{\mathrm{op}(\mathrm{convert}(DR, set)), z(0) = 0, \phi(0) = 0, \mathrm{D}(z)(0) = \sin(p), \mathrm{D}(\phi)(0) = \cos(p)\}$$

```
> DRIF(Pi/2);
```
Příklad, co to vrátí

$$\left\{ D(z)(0) = 1,\ D(\phi)(0) = 0,\ \left(\frac{\partial^2}{\partial s^2}\phi(s)\right) + 2\,\frac{z(s)\left(\frac{\partial}{\partial s}z(s)\right)\left(\frac{\partial}{\partial s}\phi(s)\right)}{1+z(s)^2},\right.$$

$$\left(\frac{\partial^2}{\partial s^2}z(s)\right) + \frac{z(s)\left(\left(\frac{\partial}{\partial s}z(s)\right)^2 - \left(\frac{\partial}{\partial s}\phi(s)\right)^2 - 2\left(\frac{\partial}{\partial s}\phi(s)\right)^2 z(s)^2 - \left(\frac{\partial}{\partial s}\phi(s)\right)^2 z(s)^4\right)}{(1+z(s)^2)\,(1+2\,z(s)^2)},$$

$$\left. z(0) = 0,\ \phi(0) = 0 \right\}$$

```
> W11:=dsolve(DRIF(0.0025),{z(s),phi(s)},type=numeric);
```
Příklad jak se to řeší

$$W11 := \mathbf{proc}(rkf45\_x)\ \ldots\ \mathbf{end}$$

```
> W:=W11(1);
```
Příklad co to je to řešení

$$W := \left[s = 1,\ \phi(s) = .9999943330602962,\ \frac{\partial}{\partial s}\phi(s) = .9999882433403452,\right.$$

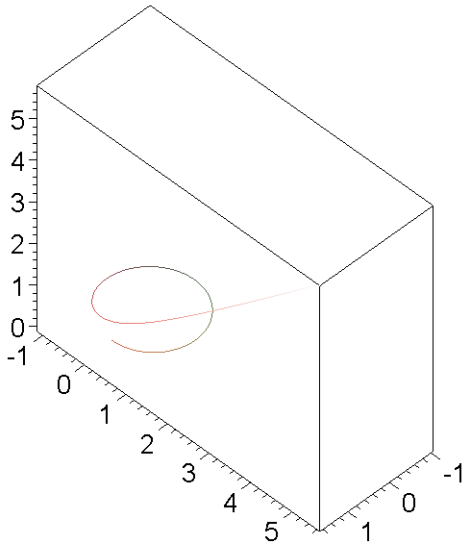
$$\left. z(s) = .002937988053371267,\ \frac{\partial}{\partial s}z(s) = .003857653642077863 \right]$$

```
> U1:=create([1],array(subs(W,convert(Rychlost,list)))):
```
A že se při řešení zachová norma

```
U2:=op(prod(g,prod(U1,U1),[1,1],[2,2]))[1][2]:
subs(z=subs(W,z(s)),U2);
```

$$compts = 1.000000000$$

```
> plots[odeplot](W11,
[R(z(s))*cos(phi(s)),R(z(s))*sin(phi(s)),z(s)], 0..14,
numpoints=250,scaling=constrained,axes=boxed);
```

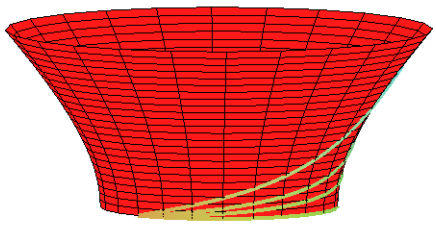



```
> Plocha:=plots[cylinderplot](R(z),
  phi=0..2*Pi,z=0..1.5,style=PATCH, color = 0):
> GG:=[];
  for a in [0.0,0.05,0.1,0.2] do
    GG:=[op(GG),
      plots[odeplot](
                    dsolve(DRIF(a),{z(s),phi(s)},type=numeric),

  [R(z(s))*cos(phi(s)),R(z(s))*sin(phi(s)),z(s)],
                      0..3,
  numpoints=50,scaling=constrained,thickness=4)];
  od:
```

$$GG := [\ ]$$

```
> plots[display](GG,Plocha);
```



>

>

Vstup a výstup

Následující procedury nají na starosti vstup, výstu a související operace.

```
> readline
> writedata
> readdata
> readbytes writebytes
> fopen fclose fflush open
> filepos
> feof
> printf sscanf
```

>

>

Obvyklý cyklus: otevřít-zapsat-zavřít

```
> soubor := open("tst.txt",WRITE);
  fprintf(soubor,`%a`,series(1/sqrt(x^2+1),x,10));
  close(soubor);
```

$$soubor := 0$$

$$58$$

Úplně stejně můžeme použít funkce **fopen** a **fclose**

>

Z textových souborů můžeme načíst anásledně rozebrat i čitelně zapsané záznamy

```
> readline(soubor);
```

```
> NactenyRetezec:="Teplota[17] = 1/(x^2-sin(x))":
> sscanf(NactenyRetezec,"%[a-zA-Z][%d] =%a");
```

$$\left[\text{"Teplota"}, 17, \frac{1}{x^2-\sin(x)}\right]$$

Cvičení s lanem

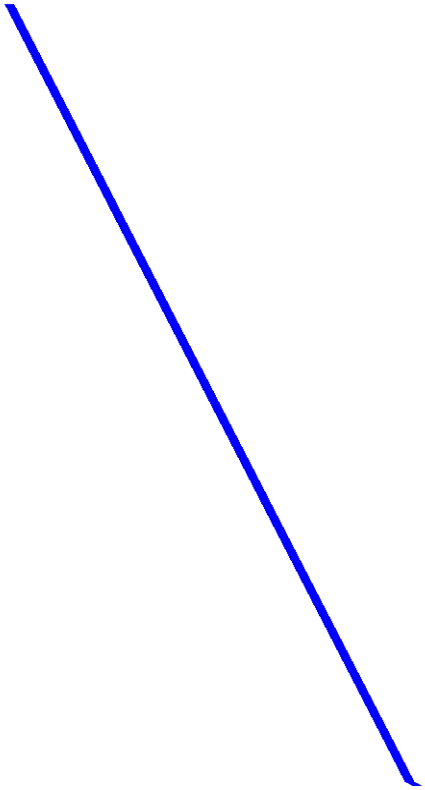
```
>
> restart;
  Digits:=35;
```

$$Digits := 35$$

```
> with(linalg):
  with(plots):
Warning, new definition for norm
Warning, new definition for trace
>
> N:=29:
> h:=1.0/N;
> T:=sum(v[i]^2/2,i=1..N);
> x[0]:=0;
  y[0]:=0;
  for i in [$1..N] do
   y[i]:=y[i-1]+sqrt(h^2-(x[i]-x[i-1])^2);
  od:i:='i':
  V:=-sum(y[i],i=1..N):
>
> W:=hessian(V,[seq(x[i],i=1..N)]):
  W:=subs({seq(x[i]=0,i=1..N)},evalm(W));
> #kontrola
  #hessian(T,[seq(v[i],i=1..N)]);
>
> Wd:=evalf(jordan(W,'Q')):
  sort([seq(Wd[i,i],i=1..N)]);
  Q1:=evalm(Q^(-1)):
```

[1.42142020984669574622888879940488 67, 7.49381163964457883156990284096756 67,
18.43666925821972788029300508845150 3, 34.28383061312044559076778338945982 8,
55.08162844207778912715634061175810 7, 80.89154261233246224891314826315109 2,
111.79140858766347811090061634609157, 147.87680257225941576805637272104238,
189.26278367059427467402714340730735, 236.08609226346792937032637555528709,
288.50791791541165848372735417710248, 346.71738637723651315477499363509412,
410.93596945629668737322838664279953, 481.42310001715437921598825658036442,
558.48338948079107586833608453296479, 642.47601761127614549190184362268582,
733.82712912845446056721378620595594, 833.04648973272626268323469887139113,
940.75033552740710402146623351814599, 1057.69350169554485448108402153657 68,

```
    1184.815947765145185848661121212107407, 1323.312563337056623851893036713159 5,
    1474.742553963715341595472695441495 2, 1641.210424003195340254732475008838 9,
    1825.687126533328745238269608892773 1, 2032.635943296301086202908976390976 6,
    2269.405742162902785899717717189711 4, 2550.045215646597688796193036582855 3,
    2910.657256480231263622956096223447 0 ]
> #kontrola
  #evalm(Q&*Wd&*Q1-W);
> CosSqrt_Wd_t:=matrix(N,N,(i,j)-> if i=j then
  cos(sqrt(Wd[i,i])*t) else 0 fi):
> x0:=[$1..N];
x0 := [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
    27, 28, 29 ]
> x0d:=evalm(Q1&*x0);
> x_t:=evalm(Q&*evalm(CosSqrt_Wd_t&*x0d)):
>
> LANO:=unapply(PLOT((evalf(CURVES([[0,0],seq([x_t[i],-i],i=1..N)]
  )),COLOR(RGB,0,0,1))),t):
>
> display([seq(LANO(i/20),i=0..400)],insequence=true);
```



```
> N;
>
```

29

```
>
>
```