# Základy numerické matematiky

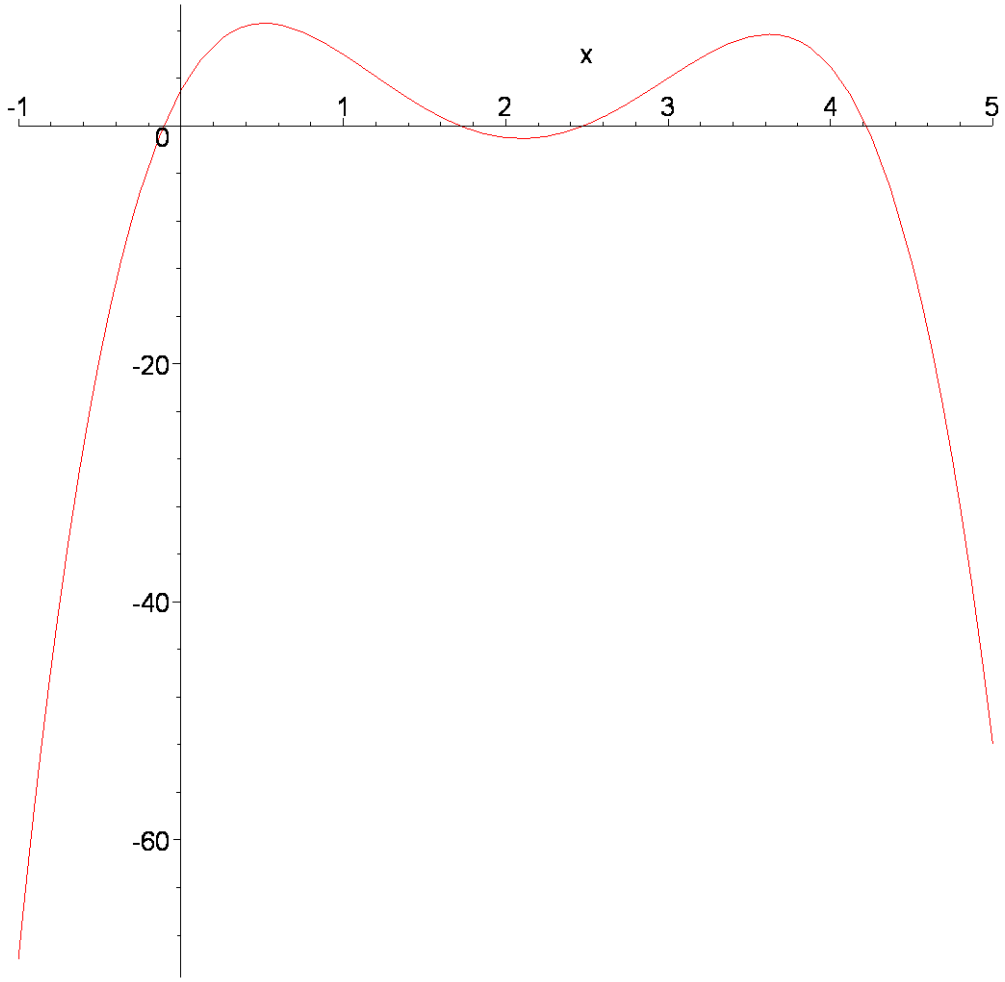
## Interpolace a aproximace funkcí

Nejdříve se podíváme na interpolaci. Lagrangeovu interpolaci počítá Maple pomocí funkce interp. Jejími parametry jsou - soubor uzlů, funkčních hodnot v těchto uzlech a název proměnné, ve které aproximační polynom počítáme.

```
> Lagrange := interp([0,1,2,3,4],[3,6,-1,4,5],x);
> f := x -> Lagrange;
> plot(f(x),x=-1..5);
```

$$Lagrange := -\frac{19}{12}x^4 + \frac{79}{6}x^3 - \frac{401}{12}x^2 + \frac{149}{6}x + 3$$

$$f := x \rightarrow Lagrange$$




Hermitův interpolační polynom neumí Maple počítat, museli bychom ho sami naprogramovat.

Interpolační polynomy často nejsou nejvhodnější aproximací funkce. Zvláště při ekvidistantních uzlech i přes zvyšování jejich počtů získáváme stále horší a horší aproximace. Proto se většinou na aproximaci funkce používá lepších funkcí, například

splinů. Maple umí počítat všechny možné druhy splinů, ale nejdříve musíme načíst knihovnu spline.

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

$$\textbf{proc}(XX, YY, z, d) \ \ldots \ \textbf{end proc}$$

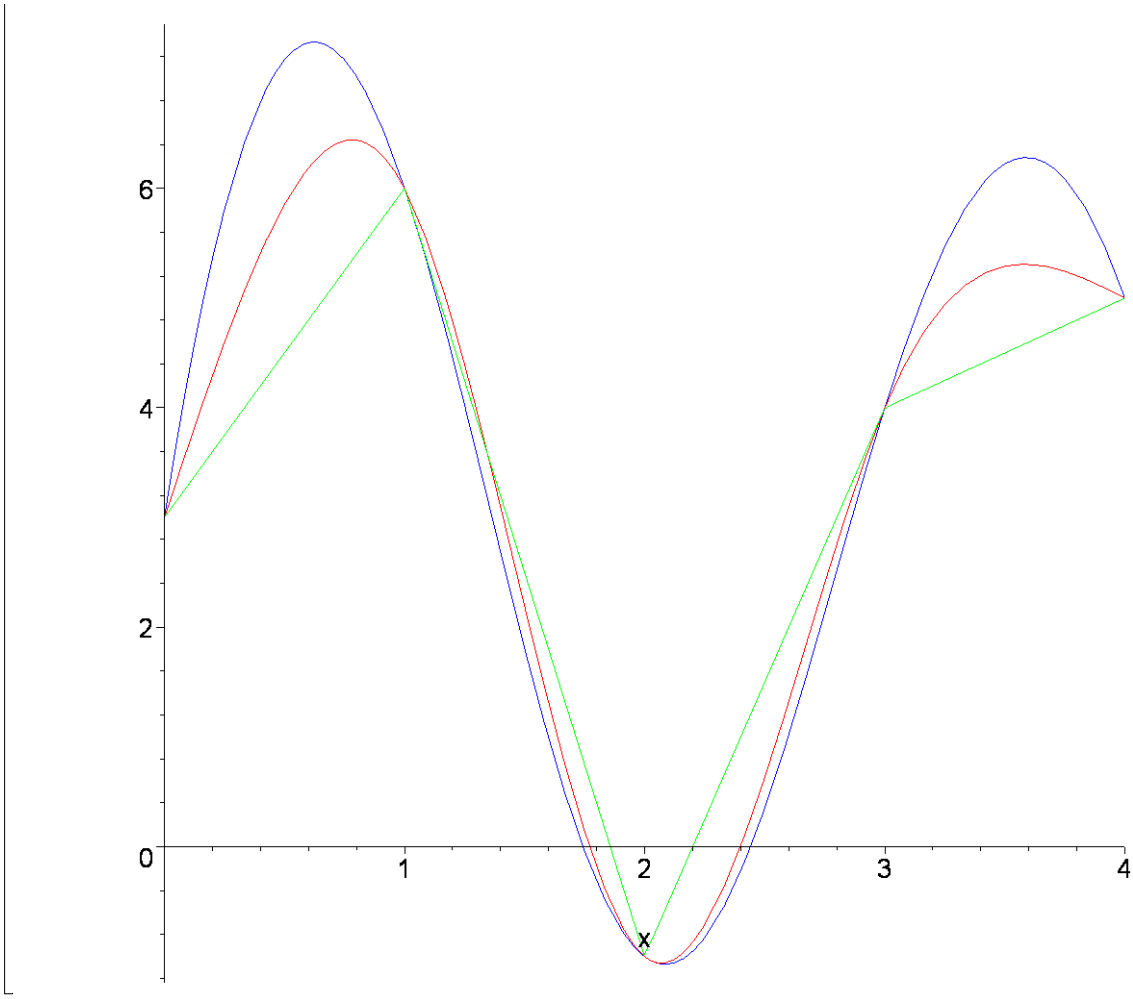
Maple počítá takzvané přirozené spliny. Například u kubického splinu to znamená, že spline je dán jednak uzly a funkčními hodnotami v nich a pak dvěmi dodatečnými podmínkami - nulovostí derivací aproximované funkce v krajních uzlech.

```
> linearni:= spline([0,1,2,3,4],[3,6,-1,4,5],x,linear);
> kubicky := spline([0,1,2,3,4],[3,6,-1,4,5],x,cubic);
> pateho_stupne := spline([0,1,2,3,4],[3,6,-1,4,5],x,5);
> plot([linearni,kubicky,pateho_stupne],x=0..4,
  color=[green,red,blue]);
```

$$linearni := \begin{cases} 3+3x & x<1 \\ 13-7x & x<2 \\ -11+5x & x<3 \\ 1+x & otherwise \end{cases}$$

$$kubicky := \begin{cases} 3+\frac{185}{28}x-\frac{101}{28}x^3 & x<1 \\ -\frac{121}{14}+\frac{1163}{28}x-\frac{489}{14}x^2+\frac{225}{28}x^3 & x<2 \\ \frac{1511}{14}-\frac{3733}{28}x+\frac{105}{2}x^2-\frac{183}{28}x^3 & x<3 \\ -\frac{878}{7}+\frac{2801}{28}x-\frac{177}{7}x^2+\frac{59}{28}x^3 & otherwise \end{cases}$$

$$pateho\_stupne := \begin{cases} 3+\frac{6497}{460}x-\frac{10701}{920}x^2+\frac{467}{920}x^5 & x<1 \\ \frac{571}{115}+\frac{1977}{460}x+\frac{7379}{920}x^2-\frac{452}{23}x^3+\frac{226}{23}x^4-\frac{1341}{920}x^5 & x<2 \\ -\frac{9917}{115}+\frac{106857}{460}x-\frac{202381}{920}x^2+\frac{2170}{23}x^3-\frac{859}{46}x^4+\frac{1281}{920}x^5 & x<3 \\ \frac{41356}{115}-\frac{234963}{460}x+\frac{253379}{920}x^2-\frac{1628}{23}x^3+\frac{407}{46}x^4-\frac{407}{920}x^5 & otherwise \end{cases}$$




Další často používanou aproximací funkcí je metoda nejmenších čtverců. Maple ji počítá příkazem leastsquare, který je obsažený v balíčku stats,fit.

```
> with(stats,fit);
```

$$[fit]$$

Teď už můžeme s chutí aproximovat. S lineární aproximací je to jednoduché. Jako parametry je potřeba zadat pouze proměnné, ve kterých bude aproximace počítána a příslušné body, kterými bude přímka prokládána.

```
> fit[leastsquare[[x,y]]]([[0,1,2,3,4],[2,3,-3,2,5]]);
```

$$y = \frac{4}{5} + \frac{x}{2}$$

Samozřejmě můžeme hledat metodou nejmenších čtverců aproximace jinými funkcemi než lineárními, například kvadratickými. Musíme pouze dodat dva další parametry. druh funkce, kterou budeme aproximovat a její parametry, které metodou nejmenších čtverců Maple dopočítá.

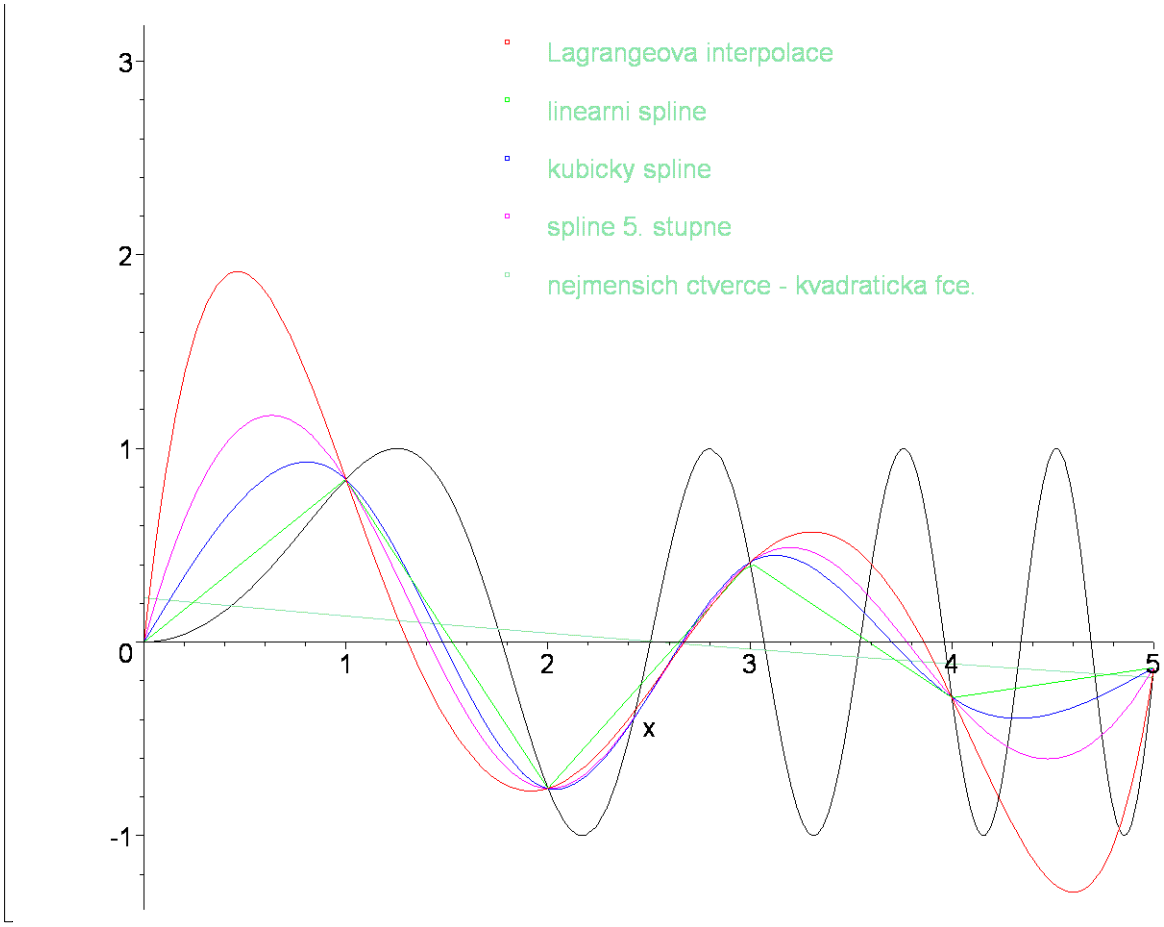
```
> fit[leastsquare[[x,y],y=a*x^2+b*x+c,
  {a,b,c}]]([[0,1,2,3,4],[2,3,-3,2,5]]);
```

$$y = \frac{15}{14}x^2 - \frac{53}{14}x + \frac{103}{35}$$

Na závěr porovnáme všechny ukázané aproximace na průměrně "odporné" funkci $\sin(x^2)$.

```
> f:= t -> sin(t^2);
> lagrange :=
  interp([0,1,2,3,4,5],[f(0),f(1),f(2),f(3),f(4),f(5)],x):
> lin_spline :=
  spline([0,1,2,3,4,5],[f(0),f(1),f(2),f(3),f(4),f(5)],x,linear
  ):
> kub_spline :=
  spline([0,1,2,3,4,5],[f(0),f(1),f(2),f(3),f(4),f(5)],x,cubic)
  :
> pat_spline :=
  spline([0,1,2,3,4,5],[f(0),f(1),f(2),f(3),f(4),f(5)],x,5):
> least_kvadra :=
  fit[leastsquare[[t,z],z=a*t^2+b*t+c,{a,b,c}]]([[0,1,2,3,4,5],
  [f(0),f(1),f(2),f(3),f(4),f(5)]]):
> kvadra_func:=unapply(rhs(least_kvadra),t):
> plot1 :=
  plot([lagrange,lin_spline,kub_spline,pat_spline,kvadra_func(x
  ),sin(x^2)],x=0..5,color=[red,green,blue,magenta,aquamarine,b
  lack]):
> plot2 := plots[textplot]([[2,3,`Lagrangeova
  interpolace`],[2,2.7, `linearni spline`],[2,2.4,`kubicky
  spline`],[2,2.1,`spline 5. stupne`],[2,1.8,`nejmensich
  ctverce - kvadraticka fce.`]],align={ABOVE,RIGHT}):
> plot11 := plots[pointplot]([1.8,3.1],color=red,symbol=BOX):
> plot22 := plots[pointplot]([1.8,2.8],color=green,symbol=BOX):
> plot33 := plots[pointplot]([1.8,2.5],color=blue,symbol=BOX):
> plot44 :=
  plots[pointplot]([1.8,2.2],color=magenta,symbol=BOX):
> plot55 :=
  plots[pointplot]([1.8,1.9],color=aquamarine,symbol=BOX):
> plots[display]({plot1,plot2,plot11,plot22,plot33,plot44,plot5
  5});
```

$$f := t \to \sin(t^2)$$




Poučení z tohoto příkladu je jedno - na špatně zvolené uzly a jejich malý počet je kterákoliv apriximační metoda krátká. Přesto je na první pohled zjevný nedostatek Lagrangeovy interpolace. Samozřejmě aproximovat zvolené body kvadratickou funkcí je nesmysl.

# Lineární algebra v numerické matematice

Na většinu zde probíraných příkladů budeme potřebovat načtení knihovny linalg.

```
> with(linalg):
```

Této knihovně se věnuje převážně list Algebra .

## Gaussova eliminace a její numerické problémy, norma matice, podmíněnost

Jak bylo ukázáno v listu Algebra Maple umí upravovat matice Gaussovou eliminací pomocí příkazu gausselim.
Numericky se v Maplu provádí Gaussova eliminace s pivotací. Pokud matice obsahuje desetiná čísla provádí se eliminace v režimu pohyblivé desetinné čárky:

```
> A:=
  matrix([[2.111,3.15,1.23,4.33],[1.04,-1.3,0,2],[5.33,4.33,
```

```
   1.22,0],[2,-1,-1.44,-10.12]]);
> gausselim(A);
```

$$A := \begin{bmatrix} 2.111 & 3.15 & 1.23 & 4.33 \\ 1.04 & -1.3 & 0 & 2 \\ 5.33 & 4.33 & 1.22 & 0 \\ 2 & -1 & -1.44 & -10.12 \end{bmatrix}$$

$$\begin{bmatrix} 5.33 & 4.33 & 1.22 & 0. \\ 0 & -2.624765479 & -1.897786116 & -10.12 \\ 0 & 0 & 1.312763974 & 10.26975440 \\ 0 & 0 & 0 & 1.071824961 \end{bmatrix}$$

Pomocí funkce pivot se v Maplu provádí pivotace matice. Parametry jsou samozřejmě matice, její řádka a sloupec tak abychom po přičtení jistého násobku řádky k ostaním řádkám dosáhli v zadaném sloupci nuly (samozřejmě až na zadaný řádek):

```
> pivot(A,4,1);
```

$$\begin{bmatrix} 0. & 4.205500000 & 2.749920000 & 15.01166000 \\ 0. & -0.7800000000 & 0.7488000000 & 7.262400000 \\ 0. & 6.995000000 & 5.057600000 & 26.96980000 \\ 2 & -1 & -1.440000000 & -10.12000000 \end{bmatrix}$$

Norma matice se počítá úplně stejně jako norma vektoru, pouze s tím rozdílem, že Maple umí jen normu spektrální (2), maximální sloupcový součet (1), maximální řádkový součet (infinity) a Schurovu (frobenius). Takhle vypadá Schurova norma matice A - tedy odmocnina ze součtu druhých mocnin prvků matice A:

```
> A := matrix(3,3,[[5,-9,5],[1,-2,1],[2,3,3]]);
> norm(A,frobenius);
```

$$A := \begin{bmatrix} 5 & -9 & 5 \\ 1 & -2 & 1 \\ 2 & 3 & 3 \end{bmatrix}$$

$$\sqrt{159}$$

Podmíněnost matice zjistíme pomocí funkce cond:

```
> A := matrix(3,3,[[5,-9,5],[1,-2,1],[2,3,3]]);
> podminenost_A := cond(A);
```

$$A := \begin{bmatrix} 5 & -9 & 5 \\ 1 & -2 & 1 \\ 2 & 3 & 3 \end{bmatrix}$$

$$podminenost\_A := 988$$

## Rozklady matic

Trojúhelníkový rozklad najdeme v Maple pomocí funkce LUdecomp:

```
> A := matrix([[3,-9,-5],[1,5,3],[-5,0,3]]);
> LUdecomp(A);
```

$$A := \begin{bmatrix} 3 & -9 & -5 \\ 1 & 5 & 3 \\ -5 & 0 & 3 \end{bmatrix}$$

$$\begin{bmatrix} 3 & -9 & -5 \\ 0 & 8 & \frac{14}{3} \\ 0 & 0 & \frac{41}{12} \end{bmatrix}$$

Příkaz LUdecomp je silně variabilní, pomocí jeho parametrů můžeme velmi podrobně ovlivňova postup výpočtů, více viz. help na LUdecomp.

Maple ovládá i jiné druhy rozkladů, například Choleského rozklad pozitivně definitní symetrické matic A na dolní trojúhelníkové matice L a $L^t$ tak, aby platilo $A = L\,L^t$ .

```
> B:= matrix([[7,-1,4],[-1,5,1],[4,1,5]]);
> cholesky(B);
```

$$B := \begin{bmatrix} 7 & -1 & 4 \\ -1 & 5 & 1 \\ 4 & 1 & 5 \end{bmatrix}$$

$$\begin{bmatrix} \sqrt{7} & 0 & 0 \\ -\frac{\sqrt{7}}{7} & \frac{\sqrt{238}}{7} & 0 \\ \frac{4\sqrt{7}}{7} & \frac{11\sqrt{238}}{238} & \frac{5\sqrt{102}}{34} \end{bmatrix}$$

Nakonec si ukážeme QR rozklad, tj. rozklad matice na horní trojúhelníkovou matici a matici unitární.

```
> QRdecomp(A);
```

$$\begin{bmatrix} \sqrt{35} & -\frac{22\sqrt{35}}{35} & -\frac{27\sqrt{35}}{35} \\ 0 & \frac{\sqrt{112910}}{35} & \frac{753\sqrt{112910}}{56455} \\ 0 & 0 & \frac{41\sqrt{3226}}{1613} \end{bmatrix}$$

# Diferenční rovnice

S diferenčními rovnicemi si Malpe poradí s pomocí příkazu rsolve. Taky takhle se například dá napsat n-faktoriál:

```
> rsolve({y(n) = n*y(n-1), y(0)=1}, y);
```

$$\Gamma(n+1)$$

Gama fce. ve výsledku si všímat nemusíte stačí vědět, že $\Gamma(n+1)=n!$. Více o gama-funkci najdete v listu Teorie míry a integrálu .

Bohužel Maple si nedokáže poradit s okrajovými podmínkami, které jsou zadané také v rekurentní formě. S následujícími rovnicemi si tak Maple neporadí:

```
> rsolve({y(0)=1/2*(1+1/2*y(0)+1/2*y(1)),y(j)=1+1/4*y(j-1)+1/2*
  y(j)+1/4*y(j+1),y(n)=1+1/2*y(n-1)+1/2*y(n)},{y});
Error, (in rsolve/single/process) more than one recurrence relation for
single function
```

Pokud ale rovnice trochu upravíme sečtením a získáme lepší vyjádření pro okrajové podmínky a pro rekurentní vyjádření, můžeme už diferenční rovnici v Maplu vyřešit:

```
> simplify(rsolve({y(0)=2*n,y(1)=6*n-2,y(j)-2*y(j-1)+y(j-2)=-4}
  ,{y}));
```

$$\{y(j)=2n+4nj-2j^2\}$$

# Numerické řešení diferenciálních rovnic

Maple obsahuje poměrně velkou knihovnu všemožných metod na numerické řešení všech možných diferenciálních rovnice. Základní používání těchto metod nastíníme na následující diferenciální rovnici:

$$y'(x) = (1+y(x)^2)\cos(x)$$

s okrajovou podmínkou $y(0)=\frac{\sqrt{3}}{3}$, na intervalu řešení [0,6]

Takhle Maple vypočítá řešení přesné:

```
> presne_reseni :=
  dsolve({diff(y(x),x)=(1+y(x)^2)*cos(x),y(0)=sqrt(3)/3},y(x));
```

$$presne\_reseni := y(x)=\tan\left(\sin(x)+\frac{\pi}{6}\right)$$

Teď se pokusíme dospět k něčemu podobnému numericky. Není na tom nic těžkého. Dokonce se i používá stejný příkaz - dsolve, jedině přidáme další parametry, které určí,

že budeme počítat numericky a jakou metodou to bude.

Takhle v Maplu vypadá nejzákladnější metoda na řešení diferenciálních rovnic - Eulerova :

```
> eulerova := dsolve({diff(y(x),x)=(1+y(x)^2)*cos(x),
  y(0)=sqrt(3)/3}, y(x), type=numeric,
  method=classical[foreuler]);
```

*eulerova* := **proc**(*x_classical*) ... **end proc**

Výsledek na pohled nic moc. Ale to zatím přejdeme, nakonec všechno srovnáme v grafu (nedočkavci ať se podívají hned ;-)). Teď si v Maplu ukážeme další metody:

Modifikovaná Eulerova:

```
> meulerova := dsolve({diff(y(x),x)=(1+y(x)^2)*cos(x),
  y(0)=sqrt(3)/3}, y(x), type=numeric,
  method=classical[impoly]);
```

*meulerova* := **proc**(*x_classical*) ... **end proc**

Runge-Kutta 2.stupně :

```
> runge_kutt2 := dsolve({diff(y(x),x)=(1+y(x)^2)*cos(x),
  y(0)=sqrt(3)/3}, y(x), type=numeric, method=classical[rk2]);
```

*runge_kutt2* := **proc**(*x_classical*) ... **end proc**

Z Runge-Kuttových metod umí Maple i metody třetího a čtvrtého stupně tj. rk3 a rk4. Maple ovládá i jen o málo složitější metody prediktor a prediktor korektor. Metoda prediktor:

```
> prediktor := dsolve({diff(y(x),x)=(1+y(x)^2)*cos(x),
  y(0)=sqrt(3)/3}, y(x), type=numeric,
  method=classical[adambash]);
```

*prediktor* := **proc**(*x_classical*) ... **end proc**

A konečně metoda prediktor-korektor:

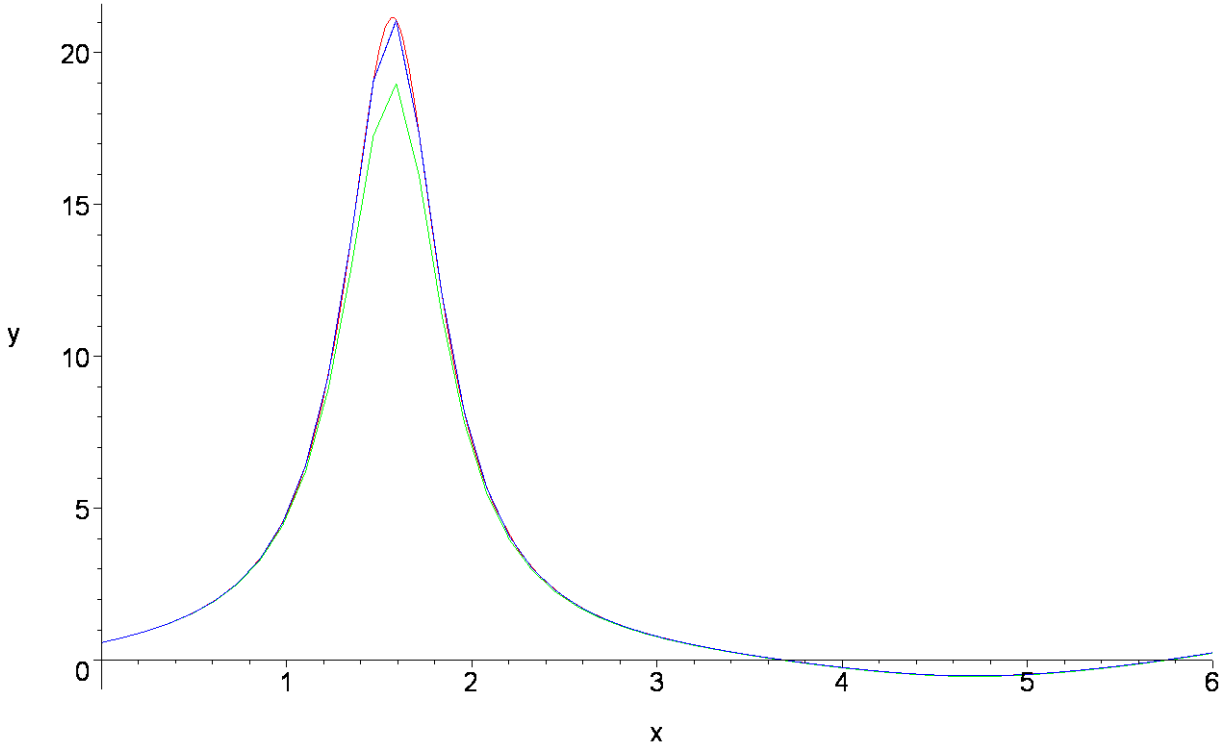
```
> pkorektor := dsolve({diff(y(x),x)=(1+y(x)^2)*cos(x),
  y(0)=sqrt(3)/3}, y(x), type=numeric,
  method=classical[abmoulton]);
```

*pkorektor* := **proc**(*x_classical*) ... **end proc**

Maple samozřejmě umí i další metody. Jak klasické tak i jiné jako např. rkdf 45 - Fehlbergova Runge-Kutta 4.-5. stupně, která se automaticky používá pokud zadáme pouze type=numeric a nezadáme method. Takovými metodami se tady ale nebudeme zabývat.

Teď se podíváme, jak vypočítané výsledky zobrazíme do grafu. Používá se na to zvláštní funkce odeplot z balíčku plots.. Zobrazíme všechny výsledky do jednoho grafu:

```
> plot1:=plots[odeplot](eulerova,[x,y(x)], 0..6, color=green):
> plot2:=plots[odeplot](meulerova,[x,y(x)], 0..6, color=brown):
> plot3:=plots[odeplot](runge_kutt2,[x,y(x)], 0..6,
  color=blue):
> plot4:=plots[odeplot](prediktor,[x,y(x)], 0..6, color=blue):
> plot5:=plots[odeplot](pkorektor,[x,y(x)], 0..6, color=blue):
> plot6:=plot(tan(sin(x)+1/6*Pi),x=0..6):
> plots[display]({plot1,plot2,plot3,plot4,plot5,plot6});
```




Jak je vidět, v daném případě všechny metody obstáli poměrně dobře až na nejjednodušší Eulerovu metodu (zeleně). Graf přesné funkce je samozřejmě červeně.

Ani pokud bychom chtěli pouze funkční hodnoty není žádný problém. Takhle vypadá funkční hodnota řešení získaného Eulerovou metodou v bodě 1.6:

```
> eulerova(1.6);
```

$$[x = 1.6, y(x) = 18.9188271034394546]$$

Nakonec si ukážeme, co dělat pokud se nám dosažená numerická přesnost nelíbí. Nejjednoduší řešení takového problému je změnit velikost kroků. Dělá se to parametrem stepsize:

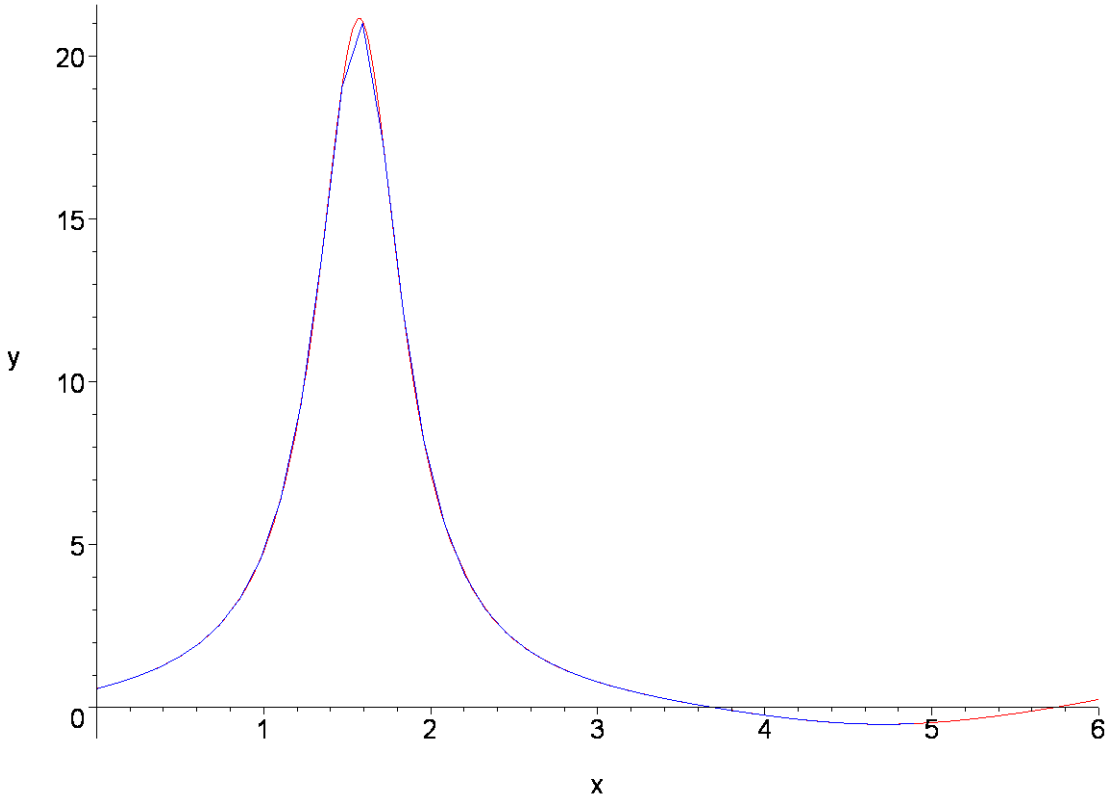
```
> euler2 := dsolve({diff(y(x),x)=(1+y(x)^2)*cos(x),
  y(0)=sqrt(3)/3}, y(x), type=numeric,
```

```
   method=classical[foreuler], stepsize=0.0001);
```

euler2 := **proc**(*x_classical*) ... **end proc**

A teď si to nakreslíme:

```
> plot7 := plots[odeplot](euler2,[x,y(x)], 0..6, color=blue):
> plots[display]({plot7,plot6});
Warning, could not obtain numerical solution at all points, plot may be
incomplete
```




```
>
>
```

Při dostatečném počtu kroků je poměrně přesná i tato jinak poměrně neefektivní metoda (modře).

Samozřejmě můžeme při numerickém řešení diferenciální rovnice měnit v Maplu mnohem více parametrů, více viz. help nadsolve[numeric] a dsolve[classical] .

## Použitá literatura


Jitka Segethová : Základy numerické matematiky
Jindřich Bečvář : Vektorové prostory I,II,III.
Jiří Anděl : Matematika náhody