

Využití programu *Mathematica*[®] při výuce matematiky na VŠCHT

RNDr. Miroslava Dubcová
Ústav matematiky VŠCHT Praha
Technická 5, Praha 6

Tento příspěvek se zabývá zařazením programu *Mathematica*[®] do výuky matematiky na VŠCHT.

Matematický software jako je *Mathematica*[®], Maple, Matlab, Derive přináší nové možnosti do výuky matematiky na technických školách. Jejich velké využití je možné především ve vyšších ročnících, kde již mají studenti osvojeny základy matematiky a jiných oborů a dokáží správně interpretovat výsledky, které získají řešením daných problémů pomocí těchto programů.

Na naší katedře využíváme především programy *Mathematica*[®] a Maple při cvičeních z předmětů:

- Fourierova Transformace pro infračervenou spektroskopii
- Soustavy obyčejných diferenciálních rovnic
- Numerické metody

V prvních dvou cvičeních seznámíme studenty s obsluhou programu, syntaxí příkazů a funkcemi, které budou v daném předmětu potřebovat. Tak například v předmětu Fourierova transformace se seznámí především s integrací a se standardní knihovnou `FourierTransform`, v předmětu Soustavy obyčejných diferenciálních rovnic především s řešením diferenciálních rovnic, jak s analytickým řešením, tak s numerickým řešením a se standardní knihovnou `FilledPlot`, v předmětu Numerická matematika s programovacími prostředky programu *Mathematica*[®]. Ve všech třech předmětech je naučíme využívat grafické možnosti programu a hlavně správně interpretaci získané výsledky. Kromě standardních knihoven používají studenti ještě programy napsané na katedře matematiky. Ty studentům blíže objasní daný problém a usnadní jim interpretaci daných výsledků.

1 Ukázka použití knihoven ústavu matematiky

1.1 Diferenciální rovnice - kvalitativní teorie

1.1.1 Fázové portréty

Student si pomocí vlastních čísel, vlastních vektorů a transformace souřadnic načrtne fázový portrét dané soustavy dvou lineárních rovnic. Výsledek si pak zkontroluje pomocí programu FazPort z knihovny našeho ústavu.

Student si zpřístupní naši knihovnu příkazem:

```
<<Matematika'FazPort'
```

Syntaxi příkazu pro vykreslení fázového portréту soustavy dvou lineárních rovnic zjistí student pomocí nápovědy:

```
?LFazPort
```

Jako odpověď dostane:

```
LFazPort[t1,t2,a11,a12,a21,a22,param] Fazovy portret soustavy obycejnych diferencialnich rovnic  $x'=Ax$ . t1 je pocatecni cas integrace, t2 je koncovy cas integrace, aij jsou prvky matice A. Param je nepovinne parametru pro vykresleni grafu.
```

Příklad 1:

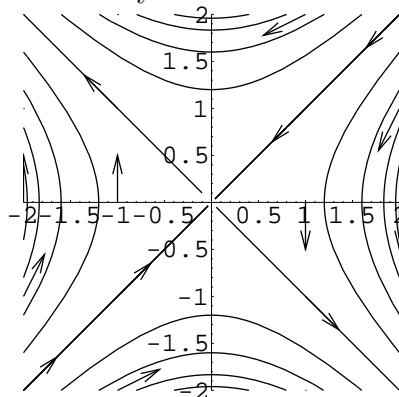
Je zadána soustava rovnic:

$$\begin{aligned}x' &= -y \\ y' &= -x\end{aligned}$$

Příkaz pro program *Mathematica*® má tvar:

```
LFazPort[0, 4, 0, -1, -1, 0]
```

Výsledný fázový portrét dané soustavy diferenciálních rovnic:



Příklad 2:

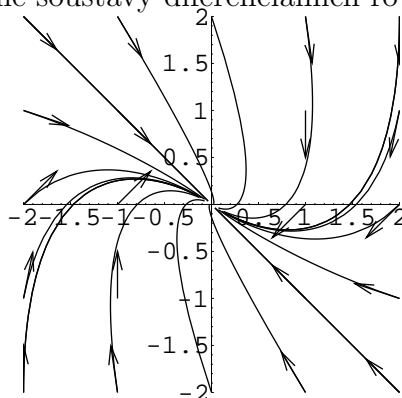
Soustava rovnic:

$$\begin{aligned}x' &= -x + y \\y' &= -x - 3y\end{aligned}$$

Příkaz pro program *Mathematica*® má tvar:

LFazPort[0, 2, -1, 1, -1, -3]

Výsledný fázový portrét dané soustavy diferenciálních rovnic:

**Příklad 3:**

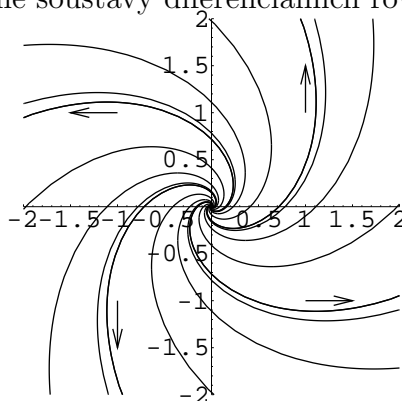
Soustava rovnic:

$$\begin{aligned}x' &= x - y \\y' &= x + y\end{aligned}$$

Příkaz pro program *Mathematica*® má tvar:

LFazPort[0, 6, 1, -1, 1, 1]

Výsledný fázový portrét dané soustavy diferenciálních rovnic:

**Příklad 4:**

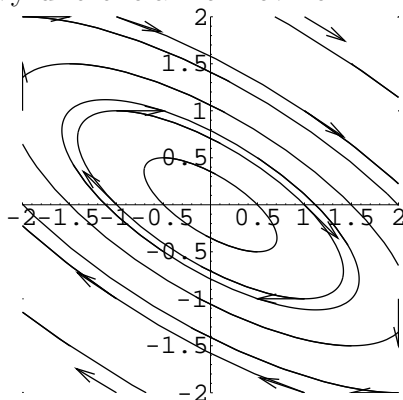
Soustava rovnic

$$\begin{aligned}x' &= x + 2y \\y' &= -x - y\end{aligned}$$

Příkaz pro program *Mathematica*® má tvar

```
LFazPort[0, 4, 1, 2, -1, -1]
```

Fázový portrét dané soustavy diferenciálních rovnic:



Podobně postupuje student i v případě soustavy nelineárních diferenciálních rovnic. Nejprve si zobrazí fázové portréty linearizované soustavy diferenciálních rovnic v okolí jednotlivých rovnovážných stavů dané nelineární soustavy diferenciálních rovnic (tedy lokální fázové portréty), potom se pokusí vykreslit globální fázový portrét soustavy nelineárních diferenciálních rovnic. K tomu mu poslouží příkaz NFazPort. Parametry příkazu získá opět pomocí nápovědy:

```
?NFazPort
```

Jako odpověď dostane:

```
NFazPort[f1,f2,t1,a1,a2,b1,b2,data,param] Fazovy portret soustavy obycejnych diferencialnich rovnic x'=f(x). f1 a f2 jsou slozky prave strany, t1 je koncovy cas integrace, a1, a2, b1, b2 jsou meze grafu a data jsou pocatecni podminky ve tvaru {{x0,y0},{x1,y1}...}. param je nepovinnny parametr pro funkci ParametricPlot.
```

Příklad 5:

Mějme soustavu nelineárních diferenciálních rovnic:

$$\begin{aligned}x' &= xy - x \\ y' &= -x - y\end{aligned}$$

Pravá strana soustavy diferenciálních rovnic má tvar:

```
f1[x_,y_]=x*y-x;  
f2[x_,y_]=-x-y;
```

Student pomocí programu *Mathematica*[®] nalezne rovnovážné stavy dané soustavy nelineárních diferenciálních rovnic:

```
Solve[{f1[x,y] == 0 , f2[x,y] == 0},{x,y}]
```

výsledek:

```
{{x->-1,y->1},x->0,y->0}}
```

Naše soustava má dva rovnovážné stavy. Student linearizuje soustavu v okolí těchto rovnovážných stavů a vykreslí si fázové portréty linearizovaného problému. K tomu potřebuje výpočet Jacobiho matice v rovnovážném stavu $[0,0]$. Obecnou Jacobiho matici vypočte pomocí příkazu `Outer` a dosadí hodnoty rovnovážného stavu:

```
Outer[D,{f1[x,y],f2[x,y]},{x,y] /. {x->0,y->0}
```

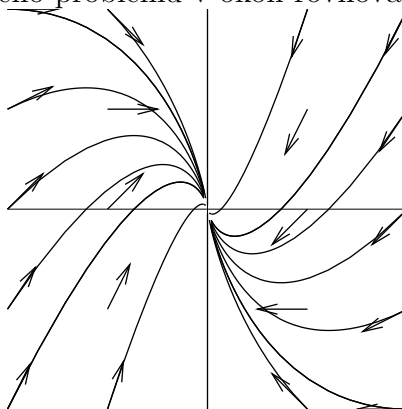
dostane výsledek:

```
{{-1,0},{-1,-1}}
```

Příkaz pro vykreslení fázového portréty linearizovaného problému v okolí rovnovážného bodu $[0,0]$:

```
LFazPort[0,4,-1,0,-1,-1,Ticks->None]
```

Fázový portrét linearizovaného problému v okolí rovnovážného stavu $[0,0]$:



Podobně postupuje student v případě druhého rovnovážného stavu. Výpočet Jacobiho matice v rovnovážném stavu $[-1,1]$:

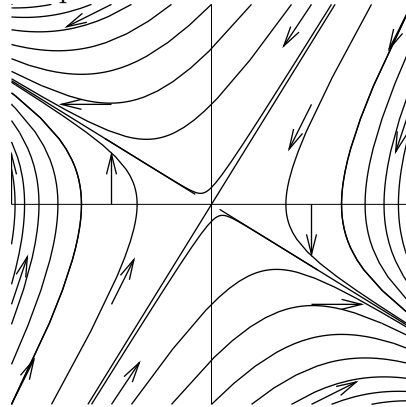
```
Outer[D,{f1[x,y],f2[x,y]},{x,y] /. {x->-1,y->1}
```

```
{{0,-1},{-1,-1}}
```

Vykreslení fázového portréty linearizovaného problému v okolí rovnovážného bodu $[-1,1]$:

```
LFazPort[0,6,0,-1,-1,-1,Ticks->None]
```

Fázový portrét linearizovaného problému v okolí rovnovážného stavu $[-1, 1]$:

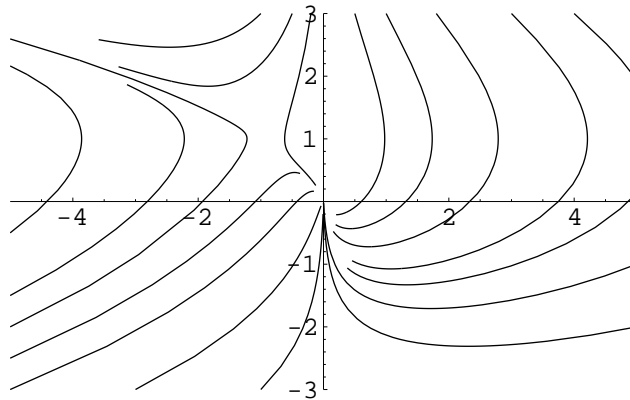


Nakonec si student zobrazí celý nelineární fázový portrét:

```
data1={{-3,-3},{1.8,3},{1,3},{0.5,3},{-5,-2.5},{-5,-3},{-1,-3}};
data2={{-1,3}};
data3={{-0.5,3},{-5,-0.5},{-5,-1.5},{3,3},{4,3}};
data4={{-0.2,3},{5,-2},{5,-1},{-5,-2}};
```

```
g1=NFazPort[f1,f2,2.5,-5,5,-3,3,data1,DisplayFunction->Identity];
g2=NFazPort[f1,f2,0.8,-5,5,-3,3,data2,DisplayFunction->Identity];
g3=NFazPort[f1,f2,1.7,-5,5,-3,3,data3,DisplayFunction->Identity];
g4=NFazPort[f1,f2,5.5,-5,5,-3,3,data4,DisplayFunction->Identity];
```

```
Show[{g1,g2,g3,g4},DisplayFunction->${DisplayFunction}]
```



1.1.2 První integrál

Použije-li student knihovnu GrCont3D, může si zobrazit fázový portrét pomocí prvního integrálu soustavy diferenciálních rovnic. Ukážeme si to na jednom lineárním příkladu a na pohybové rovnici matematického kyvadla $x'' = -k \sin x$, kterou převedeme na soustavu dvou nelineárních rovnic.

Příklad 1:

Funkce $f(x, y) = x^2 - y^2 + xy$ je první integrál soustavy diferenciálních rovnic

$$\begin{aligned}x' &= -x + 2y \\y' &= 2x + y\end{aligned}$$

Student si může zobrazit první integrál i s příslušným fázovým portrétem pomocí příkazu GrCont3D, který je součástí knihovny GrCont3D:

```
<<Matematika'GrCont3D
```

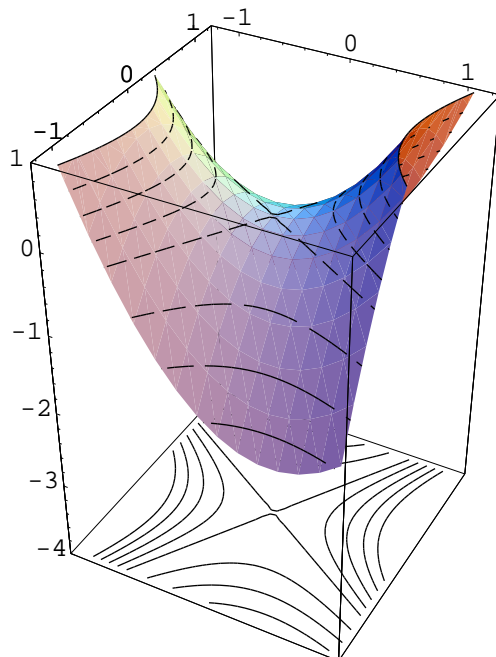
Pomocí nápovědy opět získá syntaxi příkazu:

```
?GrCont3D
```

```
"GrCont3D[funkce,{x,xmin,xmax},{y,ymin,ymax},list,z0,param]  
  3Dgraf plus jeho vrstevnice promítnute do roviny z=z0,  
  list je seznam bodu, v kterych bude proveden rez,  
  param je nepovinný parametr pro graf"
```

```
GrCont3D[x x-y y+x y,{x,-1.2,1.2},{y,-1.2,1.2},  
  {0.0,0.25,0.5,0.75,1.0,-0.5,-1,-1.5},-4,  
  PlotRange->{-4,1},BoxRatios->{1,1,1.5}]
```

jako výsledek dostane následující graf (fázový portrét se získá průmětem vrstevnic grafu prvního integrálu) :



Příklad 2:

Funkce $f(x, y) = \frac{1}{2}y^2 - k \cos x$ je první integrál soustavy diferenciálních rovnic

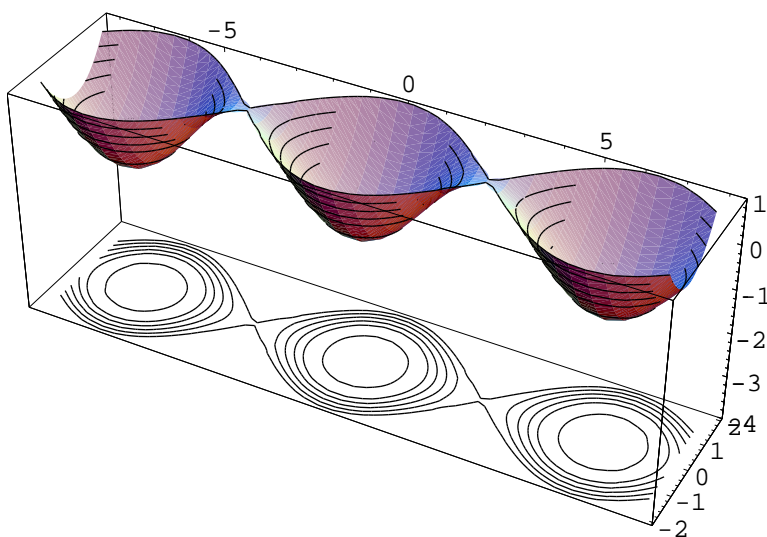
$$\begin{aligned}x' &= y \\y' &= -k \sin x\end{aligned}$$

Zobrazení prvního integrálu s příslušným fázovým portrétem:

$k = 1$

```
GrCont3D[y*y/2-k Cos[x],{x,-8,8},{y,-2,2},  
  {0.0,0.25,0.5,0.75,1.0,-0.5,-1,-1.5},-4,  
  PlotRange->{-4,1},BoxRatios->{4,1,1.5}]
```

Výsledný graf:



1.2 Fourierova transformace

Pro předmět Fourierova transformace máme vypracován na katedře matematiky program pro výpočet transmitance a absorbance k naměřenému signálu. Student získá data pomocí Michelsonova interferometru a zpracuje je pomocí programu Transm a Absorb, které jsou součástí knihovny Spektr.

Příklad 1:

Načtení knihovny Spektr:

```
<<Matematika'Spektr'
```

Načtení naměřených dat:

```
dbeg=ReadList["b1.dat",Number];  
dvzor=ReadList["b5.dat",Number];
```

Parametry programu Transm a Absorb může získat student pomocí nápovědy:

```
?Transm
```

Jako odpověď dostane:

```
Transm[dvzor,dbeg,parametr] pomerove spektrum - transmitance.  
dvzor-interferogram vzorku, dbeg-interferogram pozadi.  
Jestlize parametr ZeroFilling -> True, provede se metoda zerofilling,  
v pripade False se neprovede.
```

```
?Absorb
```

```
Absorb[dvzor,dbeg,parametr] pomerove spektrum - absorbance.  
dvzor-interferogram vzorku, dbeg-interferogram pozadi.  
Jestlize parametr ZeroFilling -> True, provede se metoda erofilling,  
v pripade False se neprovede.
```

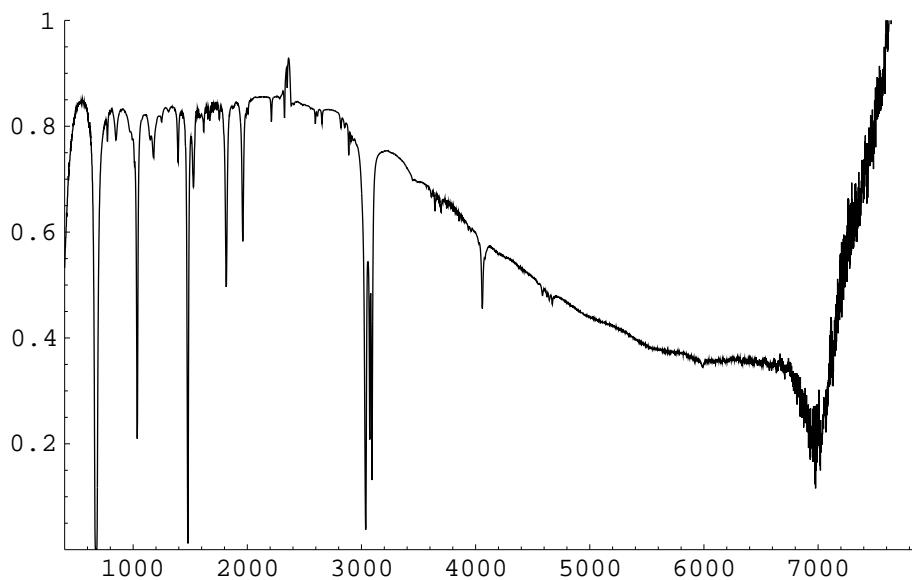
Nyní vypočte student transmitanci pomocí programu Transm a absorbanci pomocí programu Absorb:

```
dt=Transm[dvzor,dbeg,ZeroFilling->False];  
da=Absorb[dvzor,dbeg,ZeroFilling->False];
```

Nakonec si student vypočtenou transmitanci graficky zobrazí:

```
T = 0.5183520977;  
dv = 1/T;  
n=Length[dt];  
fr=n*dv/2;  
d1=Table[{dv*(i-1),dt[[i]]},{i=1,n}];  
ListPlot[d1,PlotJoined->True,PlotRange->{{400,fr},{0,1}},  
AxesOrigin->{400,0}]
```

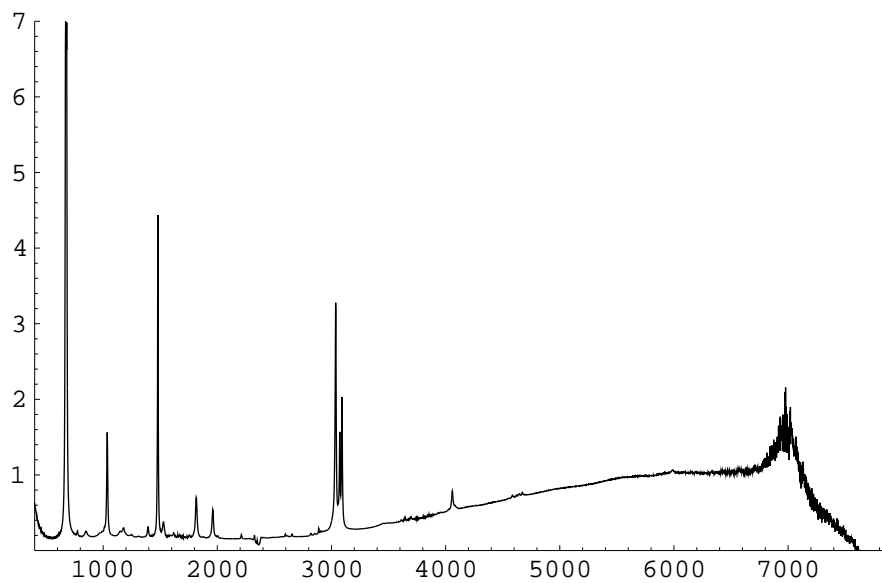
Výsledný graf:



Grafické zobrazení absorbance:

```
d2=Table[{dv*(i-1),da[[i]]},{i=1,n}];  
ListPlot[d2,PlotJoined->True,PlotRange->{{400,fr},{0,1}},  
  AxesOrigin->{400,0}]
```

Výsledný graf:



2 Numerická matematika

Při cvičeních z numerické matematiky používají program *Mathematica*[®] pouze studenti kteří již mají hlubší znalosti tohoto programu.

Na závěr ukáží na dvou programech použití programu *Mathematica*[®] pro předmět Numerické metody.

1. Program pro řešení soustavy nelineárních rovnic Newtonovou metodou:

```
newtonsous[f_, z_, x0_, max_, eps_] :=
Module[{x1, xn, s, i, x, prs, res, j, jak, jak0, y}, {
  n = Length[z]; i = 0; Print["iterace xn          sn"];
  Print["  ", i, "          ", N[x0, 8]];
  x1 = x0;
  s = 999999;
  x = {x0};
  While[i < max && s > eps,
    jak = Outer[D, f, z] /. Thread[Rule[z, x1]];
    prs = f /. Table[z[[j]] -> x1[[j]], {j, 1, n}];
    res = LinearSolve[jak, -prs];
    xn = x1 + res;
    s = Sqrt[res . res];
    i++;
    Print["  ", i, "          ", N[xn, 8], "          ", N[s, 8]];
    x1 = N[xn, 8];
    x = Join[x, {x1}];];
  Print["V\{y}sledek"];
  Print[N[x1, 8]]; x}][[1]]
```

Použití programu:

```
f[x_, y_] = {x*x + x - y + 1, x + x*y - 2};
newtonsous[f[x, y], {x, y}, {1, 1}, 10, 0.0001];
```

Výsledek:

iterace	xn	Sn
0	{1., 1.}	
1	{0.6, 1.8}	0.89442719
2	{0.65436893, 2.0796117}	0.28484848
3	{0.6506358, 2.0739488}	0.0067826302
4	{0.65062919, 2.0739475}	0.0000067292947

```
V\{y}sledek:
{0.65062919, 2.0739475}
```

2. Výpočet soustavy obyčejných diferenciálních rovnic metodou Runge-Kutta:

```
RKStep[f_, y_, y0_, dt_] :=  
  Module[{ k1, k2, k3, k4 },  
    k1 = dt N[ f /. Thread[y -> y0] ];  
    k2 = dt N[ f /. Thread[y -> y0 + k1/2] ];  
    k3 = dt N[ f /. Thread[y -> y0 + k2/2] ];  
    k4 = dt N[ f /. Thread[y -> y0 + k3] ];  
    y0 + (k1 + 2 k2 + 2 k3 + k4)/6  
  ]  
RKSolve[f_List, y_List, y0_List, {t1_,n_}] :=  
  NestList[ RKStep[f, y, #, N[t1/n]]&, N[y0],n ];  
RKSolve[f_List, y_List, y0_List, {t_, t0_, t1_, n_}] :=  
  Module[{res},  
    res=RKSolve[Prepend[f,1],Prepend[y,t],Prepend[y0,t0],{t1-t0,n}];  
    res]
```

Použití programu:

```
f1[t_,x_,y_]=x x+y y-9 ;  
f2[t_,x_,y_]=x x-1;  
data=RKSolve[{f1[t,x,y],f2[t,x,y]},{x,y},{-0.5,1.2},{t,0,5,50}]
```

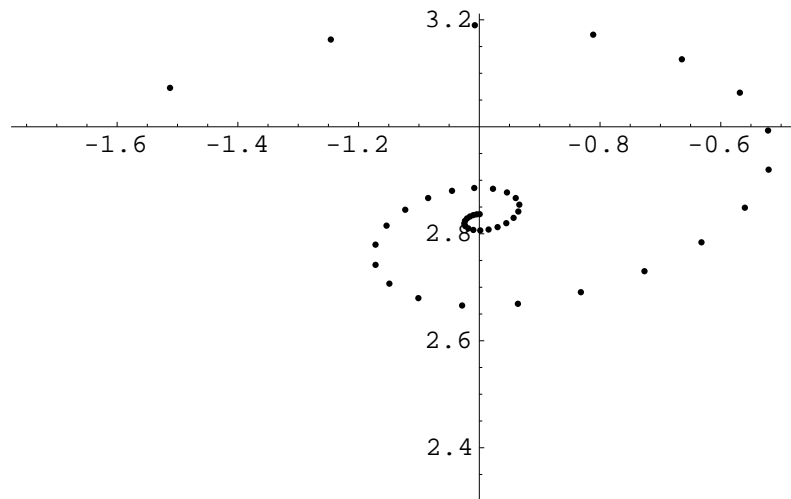
Výsledek:

```
{ {0,-0.5,1.2}, {0.1,-1.18464,1.17644}, {0.2,-1.71766,1.29391},  
  {0.3,-2.05396,1.55691}, {0.4,-2.19523,1.91505},  
  {0.5,-2.16874,2.29661}, {0.6,-2.01507,2.63805},  
  {0.7,-1.78112,2.90056}, {0.8,-1.5124,3.07272},  
  {0.9,-1.24601,3.16313}, {1.,-1.00744,3.18987},  
  {1.1,-0.811361,3.17218}, {1.2,-0.66446,3.1262},  
  {1.3,-0.568369,3.06376}, {1.4,-0.521654,2.99305},  
  {1.5,-0.520751,2.91985}, {1.6,-0.560105,2.84874},  
  {1.7,-0.631972,2.78403}, {1.8,-0.726395,2.73004},  
  {1.9,-0.831763,2.69077}, {2.,-0.936058,2.66909},  
  {2.1,-1.02848,2.66591}, {2.2,-1.10096,2.67971},  
  {2.3,-1.14903,2.70677}, {2.4,-1.17192,2.74191},  
  {2.5,-1.17197,2.77966}, {2.6,-1.15369,2.81518},  
  {2.7,-1.12269,2.84492}, {2.8,-1.08478,2.86683},  
  {2.9,-1.04523,2.88026}, {3.,-1.00837,2.88563},  
  {3.1,-0.977361,2.8841}, {3.2,-0.954181,2.87724},  
  {3.3,-0.939676,2.86677}, {3.4,-0.933718,2.85438},  
  {3.5,-0.935388,2.84162}, {3.6,-0.943196,2.82976},
```

```
{3.7,-0.955302,2.81982},{3.8,-0.969747,2.81244},  
{3.9,-0.984662,2.80794},{4.,-0.998448,2.80629},  
{4.1,-1.0099,2.80718},{4.2,-1.01825,2.81007},  
{4.3,-1.02321,2.81431},{4.4,-1.02488,2.81923},  
{4.5,-1.02371,2.82419},{4.6,-1.02032,2.82868},  
{4.7,-1.01547,2.83231},{4.8,-1.00994,2.83487},  
{4.9,-1.0044,2.8363},{5.,-0.999416,2.83667}}
```

Vykreslení trajektorie:

```
data1=Drop[#,1]& /@ data;  
ListPlot[data1]
```



3 Přílohy

3.1 Knihovna FazPort

```
(* :Title:FazPort *)
(* :Author: Miroslava Dubcova, VSCHT *)

(* :Summary:
    Fazove portrety pro soustavu dvou obycejnych
    diferencialnich rovnic
*)
(* :Context: Mirka'FazPort' *)
(* :Package Version: 1.0 *)
(* :Mathematica Version: 2.2 3.0 *)
(* :Copyright: Copyright Miroslava Dubcova *)
(* :History:
*)
(* :Keywords:
    LFazPort, sol, NFazPort, nsol
*)
(* :Warning: *)

BeginPackage["Matematika'FazPort'"]
LFazPort::usage=LFazPort[t1,t2,a11,a12,a21,a22,param]
    Fazovy portrety soustavy obycejnych diferencialnich rovnic  $x'=Ax$ .
    t1 je pocatecni cas integrace, t2 je koncovy cas integrace,
    aij jsou prvky matice A.
    Param je nepovinnny parametr pro vykresleni grafu. "

NFazPort::usage="NFazPort[f1,f2,t1,a1,a2,b1,b2,data,param]
    Fazovy portrety soustavy obycejnych diferencialnich rovnic  $x'=f(x)$ .
    f1 a f2 jsou slozky prave strany, t1 je koncovy cas integrace,
    a1, a2, b1, b2 jsou meze grafu a data jsou pocatecni podminky
    ve tvaru  $\{x_0,y_0\},\{x_1,y_1\}...$ .param je nepovinnny parametr pro
    funkci ParametricPlot."

Begin["'Private'"]
Needs["Graphics'Arrow'"];
sol[t_,a_,b_,a11_,a12_,a21_,
    a22_] := {x[t],
    y[t]}/.DSolve[{x'[t]==a11* x[t]+a12* y[t],y'[t]==a21* x[t]+a22 *y[t],
    x[0]==a,y[0]==b},{x[t],y[t]},t];
LFazPort[t1_,t2_,a11_,a12_,a21_,a22_,param___] := Module[{z},
```

```

alfa=Eigenvalues[{{a11,a12},{a21,a22}}];
vec=N[Eigenvectors[{{a11,a12},{a21,a22}}]];
If[alfa[[1]]==0 || alfa[[2]]==0,Print["Vlastni cislo je nulove"],
  If[Re[alfa[[1]]]<0 && Re[alfa[[2]]]<0,
ParametricPlot[
  Evaluate[Join[Flatten[Table[sol[z,a,b,a11,a12,a21,a22],
    {a,-2,2,4},{b,-2,2,1}],2],
  Flatten[Table[sol[z,a,b,a11,a12,a21,a22],
    {a,-2,2,1},{b,-2,2,4}],2]],
{z,t1,t2},AspectRatio->Automatic,PlotRange->{{-2,2},{-2,2}},
Epilog->{Table[Arrow[{a,b},
  {a+0.5*(a11*a+a12*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2],
  b+0.5*(a21*a+a22*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2}],
  HeadCenter->0},{a,-2,-1},{b,-2,2}],
Table[Arrow[{a,b},
  {a+0.5*(a11*a+a12*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2],
  b+0.5*(a21*a+a22*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2}],
  HeadCenter->0},{a,1,2},{b,-2,2}],param]],
  If[Re[alfa[[1]]]>0 && Re[alfa[[2]]]>0,
ParametricPlot[
  Evaluate[Join[Flatten[Table[sol[z,a,b,a11,a12,a21,a22],
    {a,-0.01,0.01,0.02},{b,-0.01,0.01,0.005}],2],
  Flatten[Table[sol[z,a,b,a11,a12,a21,a22],
    {a,-0.01,0.01,0.005},{b,-0.01,0.01,0.02}],2]],
{z,t1,t2},AspectRatio->Automatic,PlotRange->{{-2,2},{-2,2}},
Epilog->Table[Arrow[{a,b},
  {a+0.5*(a11*a+a12*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2],
  b+0.5*(a21*a+a22*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2}],
  HeadCenter->0},{a,-1,1,2},{b,-1,1,2}],param]],
  If[Re[alfa[[1]]]==0,ParametricPlot[
  Evaluate[Join[Flatten[Table[sol[z,a,a,a11,a12,a21,a22],
    {a,-2,2,0.5}],1],
  Flatten[Table[sol[z,a,-a,a11,a12,a21,a22],{a,-2,2,0.5}],1]
  ],
  {z,t1,t2},AspectRatio->Automatic,PlotRange->{{-2,2},{-2,2}},
Epilog->{Table[Arrow[{a,b},
  {a+0.5*(a11*a+a12*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2],
  b+0.5*(a21*a+a22*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2}],
  HeadCenter->0},{a,-2,-1},{b,-2,2}],
Table[Arrow[{a,b},
  {a+0.5*(a11*a+a12*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2],
  b+0.5*(a21*a+a22*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2}],
  HeadCenter->0},{a,1,2},{b,-2,2}],param]],
ParametricPlot[

```

```

Evaluate[Join[Flatten[Table[sol[z,a,b,a11,a12,a21,a22],
    {a,-2,2,4},{b,-2,2,0.4}],2],
    Flatten[Table[sol[z,a,b,a11,a12,a21,a22],
    {a,-2,2,0.4},{b,-2,2,4}],2],
    sol[z,vec[[2,1]]*0.1,vec[[2,2]]*0.1,a11,a12,a21,a22],
    sol[z,-vec[[2,1]]*0.05,-vec[[2,2]]*0.05,a11,a12,a21,a22],
    sol[z,2*vec[[1,1]]/vec[[1,2]],2,a11,a12,a21,a22],
    sol[z,-2*vec[[1,1]]/vec[[1,2]],-2,a11,a12,a21,a22]],
{z,t1,t2},AspectRatio->Automatic,PlotRange->{{-2,2},{-2,2}},
Epilog->{Table[Arrow[{a,b},
    {a+0.5*(a11*a+a12*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2],
    b+0.5*(a21*a+a22*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2}],
    HeadCenter->0],{a,-2,-1},{b,-2,2}],
    Table[Arrow[{a,b},
    {a+0.5*(a11*a+a12*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2],
    b+0.5*(a21*a+a22*b)/Sqrt[(a11*a+a12*b)^2+(a21*a+a22*b)^2}],
    HeadCenter->0],{a,1,2},{b,-2,2}],param]]]
]]]
nsol[t_,f_,g_,a_,b_,
    t0_]:= {x[t],
    y[t]}/.NDSolve[{x'[t]==f[x[t],y[t]],y'[t]==g[x[t],y[t]],x[0]==a,
    y[0]==b},{x[t],y[t]},{t,0,t0}]
NFazPort[f1_,f2_,t1_,a1_,a2_,b1_,b2_,data_,param___]:=Module[{},{
    ParametricPlot[
        Evaluate[Flatten[Apply[nsol,Apply[{z,f1,f2,#1,#2,t1}&,data,-1],
            {1}],1]],{z,0,t1},
        AspectRatio->(b2-b1)/(a2-a1),PlotRange->{{a1,a2},{b1,b2}},
        param]]]
End[]
EndPackage[]

```

3.2 Knihovna GrCont3D

(* :Title: GrCont3D *)

(* :Author: Miroslava Dubcova, VSCHT *)

(* :Summary: Program zobrazí třírozměrný graf, vrstevnice v zadanych bodech a promítne vrstevnice do roviny $z=z_0$.

Parametry jsou funkce $f[x,y]$, hranice pro promennou x ve tvaru $\{x,xmin,xmax\}$, hranice pro promennou y ve tvaru $\{y,ymin,ymax\}$, seznam z -tovych souradnic pro vrstevnice, z_0 , parametry pro třírozměrný graf (nemusi byt zadany).

*)

```

(* :Context: Mirka'GrCont3D' *)
(* :Package Version: 1.0 *)
(* :Mathematica Version: 2.2 3.0*)
(* :Copyright: Copyright Miroslava Dubcova *)
(* :History:
*)
(* :Keywords:
    GrCont3D
*)
(* :Warning:  *)

BeginPackage["Matematika'GrCont3D'"]
GrCont3D::usage=
    "GrCont3D[funkce,{x,xmin,xmax},{y,ymin,ymax},list,z0,param]
3Dgraf plus jeho vrstevnice promitnute do roviny z=z0,
list je seznam bodu, v kterych bude proveden rez,
param je nepovinnny parametr pro graf"

Begin["Private'"]
    ZTG[d_List, z_] := Map[ ZTG[#, z]& , d ];
    ZTG[Point[{x_, y_}], z_] := Point[{x, y,z}];
    ZTG[Line[d:{{_,_}...}], z_] := Line[ Map[Insert[#, z, 3]&, d] ];
    ZTG[Polygon[d:{{_,_}...}], z_] := Polygon[ Map[Insert[#, z, 3]&,d]];
    ZTG[Text[d_String, {x_, y_}, dd___], z_] := Text[d,{x,y,z},dd];
    ZTG[expr_, z_] := expr;

    Graph2DTo3D[Graphics[primitives_,options___],z_] :=
    Graphics3D[ZTG[primitives, z], BoxRatios->{1,1,1},
    Axes->{True, True, True},
    PlotRange->{Automatic, Automatic, Automatic}
    ];

    g1[Fce_,xm_List,ym_List,zl_List]:=ContourPlot[Fce,xm,ym,
    Contours->zl,
    ContourShading->False,DisplayFunction->Identity,PlotPoints->40];

    gc1[Fce_,xm_List,ym_List,zl_List]:=Table[ContourPlot[Fce,xm,ym,
    Contours->{zl[[i]]},
    ContourShading->False,DisplayFunction->Identity,PlotPoints->40],
    {i,Length[zl]};

    g2[Fce_,xm_List,ym_List]:=Plot3D[Fce,xm,ym,Mesh->False,
    DisplayFunction->Identity,PlotPoints->{40,40}];

```

```

gc[Fce_,xm_List,ym_List,zl_List]:=
  Table[Graph2DTo3D[Graphics[gc1[Fce,xm,ym,zl][[i]]],zl[[i]]],
  {i,Length[zl]}];

g[Fce_,xm_List,ym_List,zl_List,z_]:=
  Graph2DTo3D[Graphics[g1[Fce,xm,ym,zl]],z];

GrCont3D[Fce_,xm_List,ym_List,zl_List,z_,param___]:=
  Show[{g[Fce,xm,ym,zl,z],g2[Fce,xm,ym],gc[Fce,xm,ym,zl]},param];
End[]
EndPackage[]

```

3.3 Knihovna Spektr

```

(* :Title: Spektr *)
(* :Author: Miroslava Dubcova, VSCHT *)

(* :Summary: Vypocet absorbance a transmitace z interferogramu vzorku
a z interferogramu pozadi s moznosti pouzit metodu
zerofilling.
Parametry jsou interferogram vzorku,interferogram pozadi
a logicka promenna, ktera urcuje zda se provede
zerofilling.

*)
(* :Context: Mirka'Spektr' *)
(* :Package Version: 1.0 *)
(* :Mathematica Version: 2.2 3.0*)
(* :Copyright: Copyright Miroslava Dubcova *)
(* :History:
*)
(* :Keywords:
Absorb, Transm
*)
(* :Warning: *)

BeginPackage["Matematika'Spektr'"]
Absorb::usage=
  "Absorb[dvzor,dbeg,parametr] pomerove spektrum - absorbance.
dvzor-interferogram vzorku, dbeg-interferogram pozadi.
Jestlize parametr ZeroFilling -> True, provede se metoda
zerofilling, v pripade False se neprovede."

Transm::usage=
  "Transm[dvzor,dbeg,parametr] pomerove spektrum - transmitance.

```

dvzor-interferogram vzorku, dbeg-interferogram pozadi.
 Jestlize parametr ZeroFilling -> True, provede se metoda
 zerofilling, v pripade False se neprovede."

ZeroFilling::usage=

"parametr pouzity v funkcich Absorb a Transm. Jestlize
 Zerofilling->True, pouzije se metoda zerofilling."

Begin["Private"]

Options[Absorb]={ZeroFilling->False};

Options[Transm]={ZeroFilling->False};

dim="Nesouhlasí dimenze!"

Transm::dim=dim

Absorb::dim=dim

Absorb[dvzor_,dbeg_,options___]:=

Module[{n,nz,n1,nn,dvzf,dbzf,fazkorv,fazkorb,Fvzor,Fbeg,zerof},

zerof=ZeroFilling /. {options} /. Options[Absorb];

If[zerof!=True,zerof=False];

n=Length[dvzor];

If[n==Length[dbeg],

{n1=n+1;

If[zerof,

{nz=2 n;

dvzf=Table[0.0,{i,1,nz}];

dbzf=Table[0.0,{i,1,nz}];

For[i=1,i<n1,i++,dvzf[[i]]=dvzor[[i]]];

For[i=1,i<n1,i++,dbzf[[i]]=dbeg[[i]]];},

{nz=n;

dvzf=dvzor;

dbzf=dbeg;}]];

Fvzor=System'InverseFourier[N[dvzf]];

Fbeg=System'InverseFourier[N[dbzf]];

fazkorv=Table[N[Abs[Fvzor[[i]]]],{i,1,nz}];

fazkorb=Table[N[Abs[Fbeg[[i]]]],{i,1,nz}];

absor=-Log[fazkorv/fazkorb];},

Message[Absorb::dim];

absor

]

Transm[dvzor_,dbeg_,options___]:=

Module[{n,nn,dvzf,dbzf,Fvzor,Fbeg,nz,n1,fazkorv,fazkorb,zerof},

zerof=ZeroFilling /. {options} /. Options[Transm];

If[zerof!=True,zerof=False];

```

        Print[zerof];
n=Length[dvzor];
nn=Length[dbeg];
If[n==nn,
  {n1=n+1;
  If[zerof,
    {nz=2 n;
    dvzf=Table[0.0,{i,1,nz}];
    dbzf=Table[0.0,{i,1,nz}];
    For[i=1,i<n1,i++,dvzf[[i]]=dvzor[[i]]];
    For[i=1,i<n1,i++,dbzf[[i]]=dbeg[[i]]];},
    {nz=n;
    dvzf=dvzor;
    dbzf=dbeg;}}];
Fvzor=System'InverseFourier[N[dvzf]];
Fbeg=System'InverseFourier[N[dbzf]];
fazkorv=Table[N[Abs[Fvzor[[i]]]],{i,1,nz}];
fazkorb=Table[N[Abs[Fbeg[[i]]]],{i,1,nz}];
trans=fazkorv/fazkorb},
Message[Transm::dim]];
trans
]
End[]
EndPackage[]

```