Zadání příkladu z předmětu Programování prakticky

Řešení soustav diferenciálních rovnic s počáteční podmínkou

Na cvičení jsme se seznámili s řešením soustav obyčejných direneciálních rovnic. Skončili jsme s řešením problému oběhu lehké planety přitahované hmotným centrem.

```
program odr_planeta;

type tIndex = (ix, iy, ivx, ivy);
     tVektor= array[tIndex] of real;
     tPohybovaRovnice = function (U:tVektor;t:real): tVektor;


procedure krok_Euler(pohRovnice:tPohybovaRovnice;
                     var Y: tVektor;
                     var t:real; dt : real);
var i:tIndex;
    dYdt:tVektor;
begin
   dYdt := pohRovnice(Y,t);
   t := t+dt;
   for i:=low(Y) to high(Y) do Y[i] := Y[i] + dYdt[i]*dt;
end;

procedure krok_Midpoint(pohRovnice:tPohybovaRovnice;
                        var Y: tVektor;
                        var t:real; dt : real);
var i:tIndex;
    Ypul,dYdt:tVektor;
begin
   dYdt := pohRovnice(Y,t);
   for i:=low(Y) to high(Y) do Ypul[i] := Y[i] + dYdt[i]*dt*0.5;
   dYdt := pohRovnice(Ypul,t+0.5*dt);
   for i:=low(Y) to high(Y) do Y[i] := Y[i] + dYdt[i]*dt;
   t := t+dt;
end;

procedure Krok_RK4(fce_dUdt:tPohybovaRovnice; var U:tVektor; var t:real; dt:real);
var i:tIndex;
  Utmp,k1_dUdt,k234_dUdt : tVektor;
  dt2:real;
begin
  k1_dUdt := fce_dUdt(U,t);
  dt2 := 0.5*dt;
  t := t+dt2;

  for i:=low(tIndex) to high(tIndex) do Utmp[i] := U[i] + k1_dUdt[i]*dt2;
  k234_dUdt := fce_dUdt(Utmp,t);

  for i:=low(tIndex) to high(tIndex) do begin
      Utmp[i]    := U[i] + k234_dUdt[i]*dt2;
      k1_dUdt[i] := k1_dUdt[i]+2*k234_dUdt[i];
  end;
  k234_dUdt := fce_dUdt(Utmp,t);

  t += dt2;
  for i:=low(tIndex) to high(tIndex) do begin
      Utmp[i]    := U[i] + k234_dUdt[i]*dt;
      k1_dUdt[i] := k1_dUdt[i]+2*k234_dUdt[i];
  end;
  k234_dUdt := fce_dUdt(Utmp,t);
```

```
56
57    for i:=low(tIndex) to high(tIndex) do U[i] := U[i] +
          (k1_dUdt[i]+k234_dUdt[i])*dt2*0.333333333333333333;
58  end;
59
60  function pohybova_rovnice_planety(U:tVektor;t:real): tVektor;
61  const GM = 4*Pi*Pi;
62  var   r2,r_3 :real;
63  begin
64    pohybova_rovnice_planety[ix] := U[ivx];
65    pohybova_rovnice_planety[iy] := U[ivy];
66    r2 := sqr(U[ix])+sqr(U[iy]);
67    r_3 := 1/(r2*sqrt(r2));
68    pohybova_rovnice_planety[ivx] := -GM*U[ix]*r_3;
69    pohybova_rovnice_planety[ivy] := -GM*U[iy]*r_3;
70  end;
71
72  function jeNasobek(x,krok:real):boolean; // nepříliš spolehlivá verze stačí
73  begin
74    jeNasobek := abs(x-round(x/krok)*krok)<1E-9;
75  end;
76
77  var Y: tVektor;
78      t:real;
79
80  const dt=1/365;  // krok jeden den
81        tmax=2;    // dva roky
82
83  type tMetoda = procedure (fce_dUdt:tPohybovaRovnice; var U:tVektor; var t:real; dt:real);
84  const metoda: array [1..3] of tMetoda = (@krok_Euler,@krok_Midpoint,@Krok_RK4);
85  var i:integer;
86
87  begin
88    for i := low(metoda) to high(metoda) do begin
89      t := 0;
90      Y[ix] := 1;          // jednotka: AU
91      Y[iy] := 0;
92      Y[ivx] := 0;
93      Y[ivy] := 2*Pi-3;    // Au/rok, menší než kruhová rychlost
94
95      while t<tmax do begin
96        if jeNasobek(t,1/365) then writeln( t:7:4, '␣',Y[ix]:9:6,'␣',  Y[iy]:9:6);
97        metoda[i]( @pohybova_rovnice_planety, Y , t, dt);
98      end;
99
100     writeln; writeln;
101   end;
102
103 end.
```

Klíčovým krokem při volbě struktury programu bylo oddělit pohybovou rovnici (zde v `pohybova_rovnice_planety}`)a způsob, jímž ji řešíme (v programu jsou tři metody, dvě z nich jsme viděli na cvičení, třetí je nejlepší, ale nebyl čas na vysvětlení.)

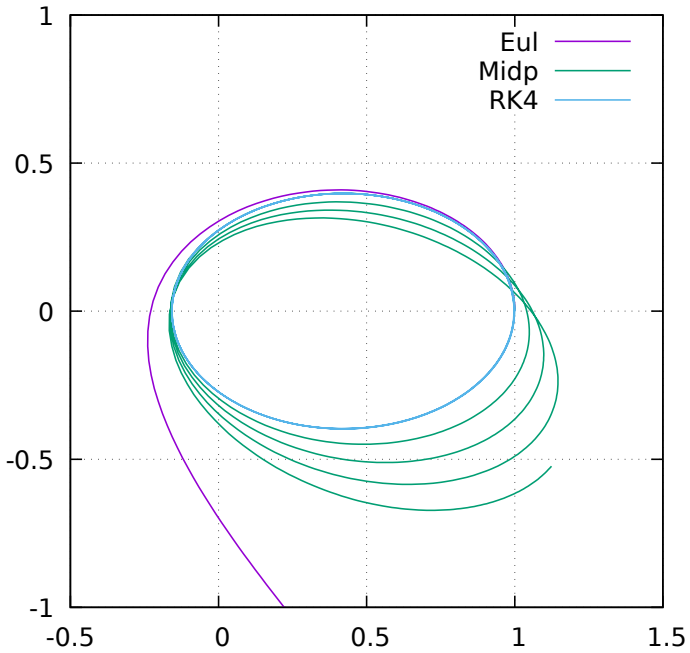
Problém, který jsme řešili je soustava dvou obyčejných diferenciálních rovnic druhého řádu, z níž jsme udělali soustavu čtyř rovnic prvního řádu. Výsledkem programu výše je sada ctyř různě přesných řešení v podobě tří sloupců čísel, které lze vykreslit příkazy gnuplotu

```
set size ratio -1
set grid
plot [-0.5:1.5][-1:1]'pl.txt' i 0 u 2:3 w l ti 'Eul', '' i 1 u 2:3 w l ti 'Midp', '' i 2 u 2:3 w l ti 'RK4'
```

Pro vykreslení do souboru pak ještě

```
set size ratio -1
set term pdf; set output 'planeta.pdf'; replot; unset term;
```

Výsledný obrázek vypadá takto:



Obrázek: Výstupr programu výše vykreslený uvedenými příkazy gnuplotu. Je vidět různá přesnost použitých metod.

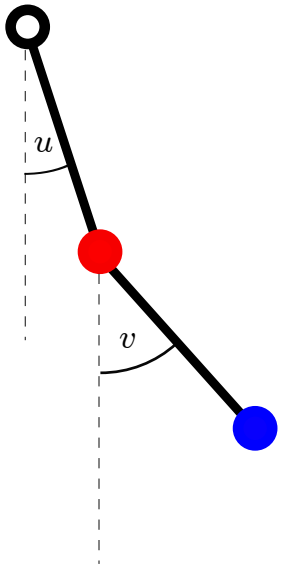
Aby program vyzkoušel všechny tři numerické metory řešení dif. rovnic, je v programu cyklus na ř. 88. Pokud chcete jen jednu metodu, je třeba cyklus zrušit a místo `metoda[i]` napsat na ř. 97 např. `Krok_RK4`.

Zápočtový problém

Zajímavým příkladem, kde nelze vystačit s analytickým řešením a je třeba rovnice řešit numericky, je *dvojkyvadlo*. To vznikne tak, že na matematické kyvadlo (to už umíme řešit) pověsíme ještě jedno. Pro jednoduchost bude mít stejně dlouhý závěs a stejně hmotné závaží.
Místo jedné obyčejné diferenciální rovnice $\ddot{\psi}+\sin\psi=0$ nyní budeme mít dvě navzájem neoddělitelně provázané

$$\ddot{u} = -\frac{2\sin(u-v)\left[\dot{v}^2+\dot{u}^2\cos(u-v)\right]+\sin(u-2v)+3\sin u}{3-\cos 2(u-v)},$$

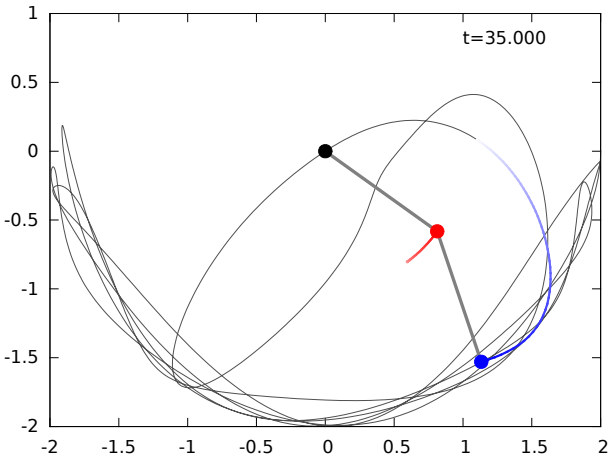
$$\ddot{v} = \frac{2\sin(u-v)\left(2\dot{u}^2+\dot{v}^2\cos(u-v)+2\cos u\right)}{3-\cos 2(u-v)}.$$



Odvodit tyto rovnice se naučíte příští rok v teoretické mechanice, kurs programování máme ale už letos. Stejně jako dříve u matematického kyvadla, i zde pro jednoduchost ve všech rovnicích uvažujeme „bezrozměrný“ čas $\tau = t\sqrt{g/l}$ a derivace podle něj. Pro kontrolu přesnosti výpočtu budeme ještě potřebovat vzorec pro celkovou (bezrozměrnou) energii obou závaží:

$$E = \dot{u}^2 + \frac{1}{2}\dot{v}^2 + \dot{u}\dot{v}\cos(u-v) - 2\cos u - \cos v.$$

Modifikujte program ze cvičení pro pohyb jednoduchého matematického kyvadla tak, aby řešil výše uvedené pohybové rovnice dvojkyvadla. Jako počáteční polohu obou závaží zvolte $u(0)=v(0)=\pi/2$, $\dot{u}(0)=\dot{v}(0)=0$.



Obr. 2. Trajektorie dolního závaží a poloha dvojkyvadla v $\tau = 35$. Rychlost je znázorněna délkou zanechané stopy. Návod jak z vašich dat vytvořit takovýto obrázek v pohybu najdete na `http://utf.mff.cuni.cz/~ledvinka/2kyv`

1a. Výstup programu nechť jsou hodnotu $\tau, u, v, E, x_1, y_1, x_2, y_2$ pro $\tau \in \langle 0, 35 \rangle$ vypsané s tabelačním krokem $\Delta\tau = 0.025$. Kartézské souřadnice obou závaží jsou $x_1 = \sin u, y_1 = -\cos u, x_2 = \sin u + \sin v, y_2 = -\cos u - \cos v$.

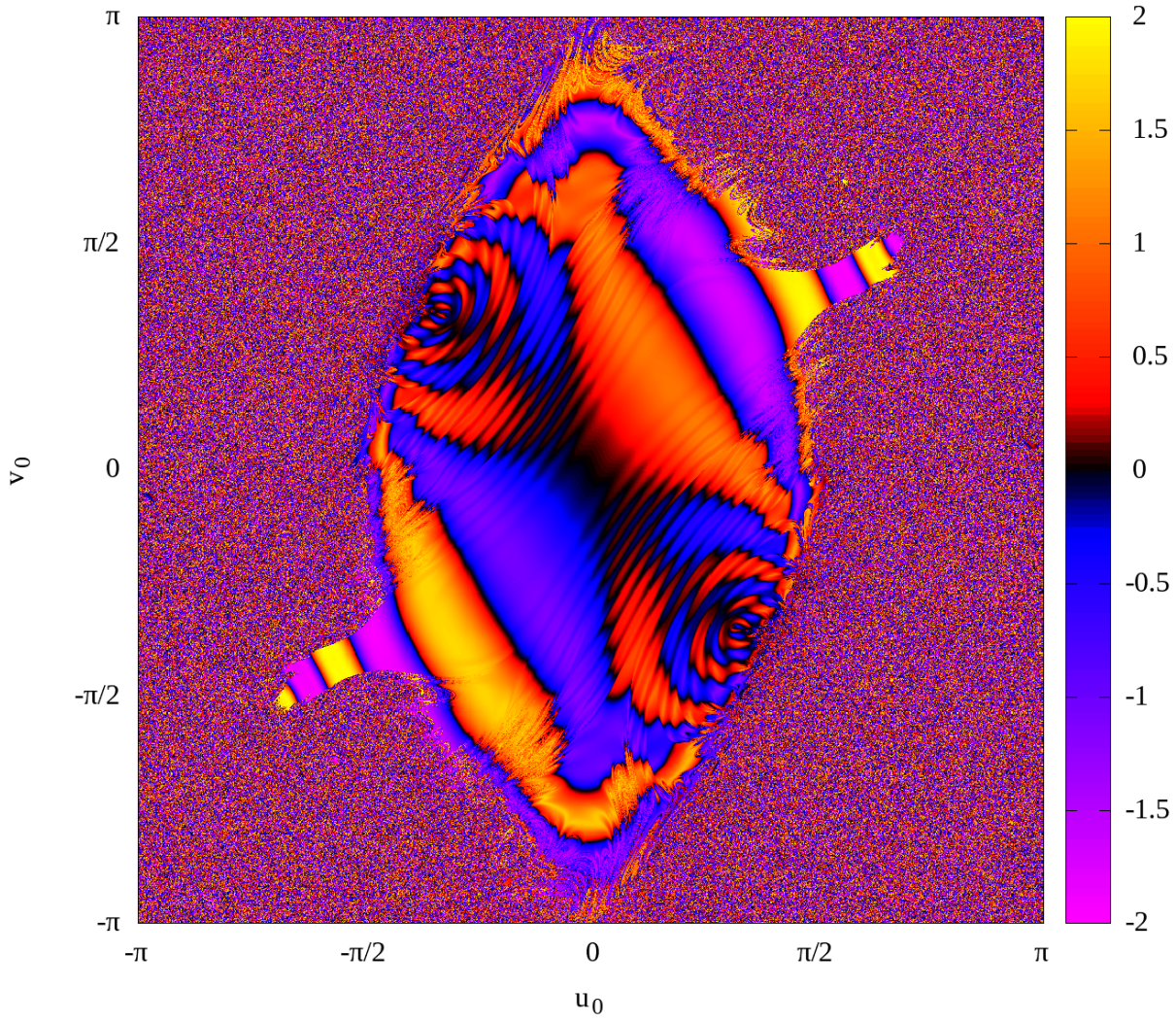
1b. Pro nejméně dvě metody uvedené výše nalezněte takovou hodnotu časového kroku při řešení diferenciální rovnice (zvoleného jako celočíselný zlomek tabelačního kroku), aby v uvažovaném časovém intervalu chyba energie E nepřesáhla 10^{-7}.

1c. Pro tuto hodnotu kroku nakreslete závislost $E(\tau)$ příkazem

```
plot 'data.txt' using 1:4 with lines
```

1d. Ze stejných dat vykreslete trajektorii druhého kyvadla

```
set size ratio -1
plot 'data.txt' using 7:8 with lines
```



Obrázek 2: Závislost výchylky spodního kyvadla v čase $\tau = 100$ na počátečních podmínkách u_0, v_0. Je vidět, že pro větší počáteční výchylky vykazuje dvojkyvadlo chaotické chování – výsledek závisí na počátečních podmínkách natolik silně, že i jen nejmenší myslitelná neznalost počátečních podmínek nám neumožní řešením pohybových rovnic určit, s jakou výchylkou kyvadlo *později* skončí.

2. Napište další verzi programu, která s nalezeným časovým krokem spočte stav dvojkyvadla v čase $\tau = 100$ vypuštěného z $u(0) = u_0, v(0) = v_0$ a vypíše hodnoty

$$u(\tau = 0) \quad v(\tau = 0) \quad u(\tau = 100) \quad v(\tau = 100) \quad E(\tau = 100)$$

stejným způsobem (viz `http://utf.mff.cuni.cz/~ledvinka/PrPrakticky/jupyter/Cviceni5.pdf`), jako jsme malovali Newtonův fraktál, nakreslí závislost výchylky spodního kyvadla $x_2 = sin(u) + sin(v)$ na počátečních podmínkách.

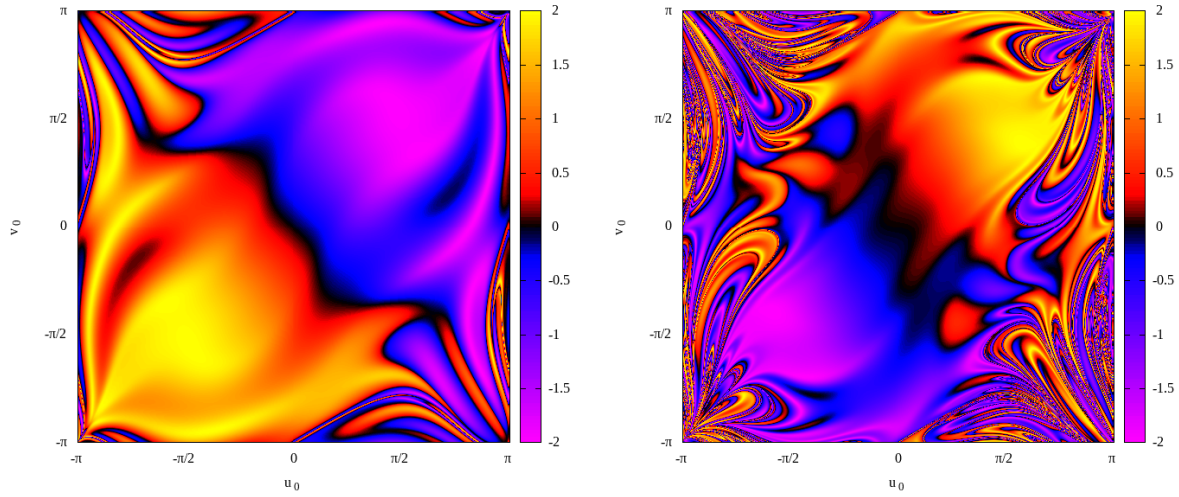
To znamená, že soubor vykreslíme příkazy

```
set size ratio -1
plot [-3.15:3.15][-3.15:3.15] 'uhly100.txt' using 1:2:(sin($3)+sin($4)) palette pt 5 ps 0.2
```

Pokud bychom chtěli hezčí, než automatickou paletu barev, je vhodné uložit následující sekvenci příkazů do souboru `obr2.gp`

```
set xlabel 'u0'
set ylabel 'v0'

set palette defined (-2 "magenta", -0.3 "blue", 0 "black", 0.3 "red", 2 "yellow")
set cbrange [-2:2]

set size ratio -1

set xtics ('-π' -pi, '-π/2' -pi/2, 0, 'π/2' pi/2, 'π' pi)
set ytics ('-π' -pi, '-π/2' -pi/2, 0, 'π/2' pi/2, 'π' pi)

plot [-pi:pi][-pi:pi] 'uhel100.txt' using 1:2:(sin($3)+sin($4)) palette pt 5 ps 0.2 notitle

#set term png size 1600, 1200; set output 'uvplot.png';replot; unset term
```

a v gunplotu ji vyvolat příkazem `load`'obr2.gp' , poslední příkaz pak po odstranění znaku # vytvoří soubor s výslednýcm obrázkem.



Obrázek 3: Závislost výchylky spodního kyvadla v čase $\tau = 5$ (vlevo) a $\tau = 10$ (vpravo) na počátečních podmínkách u_0, v_0. Je vidět, že s rostoucím časem se stále více projevuje komplikovaný průběh při větších výchylkách.