Numerické metódy pre obyčajné diferenciálne rovnice: Eulerova jednokroková metóda

Úvod:

V tomto dokumente ukážeme, ako je možné použiť systém ***Mathematica*** pri riešení obyčajných diferenciálnych rovníc (ODR) numerickými metódami. Budeme hľadať približné riešenie začiatočnej úlohy (IVP - Initial Value Problem) v tvare $y' = f(t, y),\ y(t_0) = y_0$. Riešením tejto začiatočnej úlohy bude približné riešenie definované pomocou ekvidištančných bodov v intervale $[t_0,\ t_N]$, i.e. $t_n = t_0 + n\,h$, pre $n = 0,\ ...,\ N$. Dĺžka $h = \frac{t_N - t_0}{N}$ sa nazýva **dĺžka kroku**. Približné riešenie y_n je aproximáciou presnej hodnoty $y(t_n)$ a budeme ho zapisovať $y_n = f(t_n, y_n)$.

1. Použitie NDSolve.

Je všeobecne známe, že nie je vždy možné ODR vypočítať presne. Ak systém ***Mathematica*** nedokáže vyriešiť ODR presne, príkaz DSolve vypíše presne to, čo ste vložili ako vstup:

```
ecu = DSolve[y'[t] == 1 + y[t]^2 - t^3, y[t], t]
```

```
DSolve[y'[t] == 1 - t^3 + y[t]^2, y[t], t]
```

Ak táto situácia nastane, ODR je možné vyriešiť približne numerickými metódami. Úloha však musí mať jediné riešenie a zadané začiatočné podmienky, teda musí ísť o začiatočný problém. Numerické riešenie takejto začiatočnej úlohy realizujeme pomocou príkazu **NDSolve** s nasledujúcou syntaxou:

```
NDSolve[{rovnica, počiatočné podmienky},
závisle premenná rovnice, {nezávisle premenná rovnice , interval}]
```

Všeobecne príkaz **`NDSolve[{rovnica, počiatočné podmienky},x[t],{t,a,b}]` vráti aproximáciu riešenia** *x(t)* začiatočného problému v intervale $[a, b]$. Konkrétne vráti objekt v Mathematice označený ako Interpolačná funkcia, ktorá v sebe zahŕňa približné riešenia začiatočného problému a umožňuje toto riešenie z objektu extrahovať.
Napríklad riešenie začiatočného problému $y' = 1 + y^2 - t^3,\ y(0) = 0$ na intervale [0, 4] vyzerá nasledovne:

```
solution = NDSolve[{y'[t] == 1 + y[t]^2 - t^3, y[0] == 0}, y[t], {t, 0, 4}]
```

```
{{y[t] → InterpolatingFunction[{{0., 4.}}, <>][t]}}
```

Ako je vidieť, výstupom z funkcie **NDSolve** je špeciálny objekt. Ide o objekt, ktorý tvorí tabuľka interpolačných hodnôt približného riešenia rovnice a preto je možné nakresliť graf približného riešenia. Postupujme nasledujúcim spôsobom. Najskôr z výsledku vyberme objekt "InterpolationFunction" a priraďme ho samostatnej funkcii.

```
yapprox[t_] = y[t] /. solution[[1]]
```

```
InterpolatingFunction[{{0., 4.}}, <>][t]
```

Teraz môžeme vypočítať funkčnú hodnotu v akomkoľvek bode zadaného intervalu:

```
yapprox[1]
```

```
1.16197
```

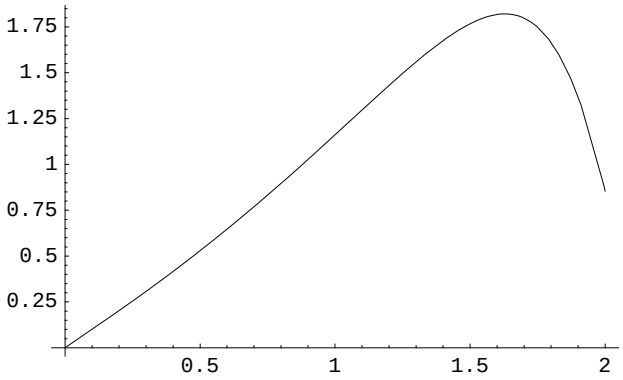
Pomocou príkazu Table môžeme zostaviť aj tabuľku hodnôt približného riešenia na intervale [0, 1] s krokom $h = 0.1$:

```
Table[{t, yapprox[t]}, {t, 0, 1, 0.1}] // TableForm // Chop
```

```
0       0
0.1     0.10031
0.2     0.202305
0.3     0.307248
0.4     0.416024
0.5     0.529206
0.6     0.647092
0.7     0.769722
0.8     0.896859
0.9     1.02794
1.      1.16197
```

alebo môžeme riešenie reprezentovať graficky pomocou funkcie **Plot**. Napr. nakreslime graf na intervale [0, 2]:

```
Plot[yapprox[t], {t, 0, 2}]
```



Poznámky:

1.- Získané približné riešenie je definované len na intervale, v ktorom sme ho vypočítali. Ak potrebujeme získať približné riešenie diferenciálnej rovnice na inom intervale, je potrebné vhodne upraviť príkaz NDSolve. Ak chceme získať riešenie mimo zadaný interval, systém *Mathematica* más informuje, že hodnoty riešenia budú získané exptrapoláciou, a teda získané riešenia môžu byť zaťažené veľkou chybou:

```
yapprox[6]
```

```
InterpolatingFunction::dmval :
 Input value {6} lies outside the range of data in the interpolating
   function. Extrapolation will be used. More…
```

```
-14.5568
```

2.- Ak použijete help programu, nájdete ďalšie možnosti, nastavenia a numerické metódy použiteľné na výpočet približného riešenia diferenciálnej rovnice. Predprogramované sú metódy Adams-Bashford, Runge-Kutta, modifikovaný Euler, atď.

Príklad 1.

Určte približné hodnoty riešenia začiatočnej úlohy $y'=y(1-\sin t)$, $y(0)=1$, na intervale $[0,1]$. Počítajte s veľkosťou kroku $h = 0.05$.

Riešenie

Použime priamo príkaz **NDSolve**.

```
sol11 = NDSolve[{y'[t] == y[t] (1 - Sin[t]), y[0] == 1}, y[t], {t, 0, 1}]
```

```
{{y[t] → InterpolatingFunction[{{0., 1.}}, <>][t]}}
```

Tabuľku výsledných hodnôt je možné získať z vytvoreného objektu "InterpolatingFunction". Rovnako ako v predchádzajú·com príklade definujeme približné riešenie a následne zostavíme tabuľku približných hodnôt.

```
yap11[t_] = y[t] /. sol11[[1]]
```

```
InterpolatingFunction[{{0., 1.}}, <>][t]
```

```
Table[{t, yap11[t]}, {t, 0, 1, 0.05}] // TableForm
```

```
0       1.
0.05    1.04996
0.1     1.09966
0.15    1.14886
0.2     1.1973
0.25    1.24472
0.3     1.2909
0.35    1.33559
0.4     1.37859
0.45    1.4197
0.5     1.45875
0.55    1.4956
0.6     1.5301
0.65    1.56218
0.7     1.59176
0.75    1.61881
0.8     1.6433
0.85    1.66526
0.9     1.68474
0.95    1.70179
1.      1.71653
```

2. Eulerova metóda.

Eulerova metóda je najjednoduchšia jednokroková metóda. Pri bežnom zápise algoritmus aproximuje riešenie začiatočnej úlohy $y' = f(t, y)$, $y(t_0) = y_0$ na intervale $[t_0 = a, b]$ vzťahom:

$$y_{n+1} = y_n + h\, f(t_n, y_n) \qquad n = 0, \ldots, m-1$$

kde $h = \dfrac{b-a}{m}$ je dĺžka kroku a časový krok je určený vzťahom $t_{n+1} = t_n + h = a + n\,h$.

Zo všeobecnej teórie numerických metód na riešenie diferenciálnych rovníc, z charakteristickej rovnice metódy $p(x) = x - 1$ je ľahké ukázať, že ide o konvergentnú numerickú metódu, pretože je stabilná a konzistentná. Iteračná schéma sa dá v systéme *Mathematica* naprogramovať pomocou nasledujúcej procedúry:

```
euler[f_, h_, ini_, a_, b_] := Module[{y, t, ytable, c}, c = (b - a) / h;
 y[0] = ini; t[n_] := a + n h; y[n_] := y[n] = y[n - 1] + h f[t[n - 1], y[n - 1]];
 ytable = Table[y[i], {i, 0, c}];
Table[{t[i], ytable[[i + 1]]}, {i, 0, c}] // TableForm]
```

kde **f** je funkcia asociovaná s pravou stranou riešeného začiatočného problému, **h** je dĺžka kroku, **ini** je hodnota začiatočnej podmienky a **a** a **b** sú hraničné body. Výstupom je tabuľka hodnôt približného riešenia ODR.

Definovanú procedúru použijeme na nájdenie riešenia začiatočnej úlohy $y' = -t\,y + \frac{4t}{y}$, $y(0)=1$ na intervale [0,1], s dĺžkou kroku 0.1. Funkcia z pravej strany začiatočného problému je definovaná nasledujúcim vzťahom

```
f[t_, y_] = -t y + 4 t / y;
```

a predchádzajúci algoritmus realizujeme s hodnotami $h = 0.1,\ ini = 1,\ a = 0,\ b = 1$.

```
euler[f, 0.1, 1, 0, 1]
```

0	1
0.1	1
0.2	1.03
0.3	1.08707
0.4	1.16485
0.5	1.25561
0.6	1.35211
0.7	1.44849
0.8	1.5404
0.9	1.6249
1.	1.70021

Aby sme mohli vypočítať presnosť približného riešenia, vypočítame presné riešenie začiatočnej úlohy a jeho hodnotu v jednotlivých bodoch porovnáme s približným riešením získaným Eulerovou metódou. Na výpočet presného riešenia použijeme príkaz **DSolve**

```
exactsol = DSolve[{y'[t] == -t y[t] + 4 t / y[t], y[0] == 1}, y[t], t]
```

$$\left\{\left\{y[t] \to e^{-\frac{t^2}{2}} \sqrt{-3 + 4\, e^{t^2}}\right\}\right\}$$

Definujeme funkciu yex(t) , do ktorej zapíšeme presné riešenie diferenciálnej rovnice

```
yex[t_] = exactsol[[1, 1, 2]]
```

$$e^{-\frac{t^2}{2}} \sqrt{-3 + 4\, e^{t^2}}$$

Pomocou príkazu Table zostavíme tabuľku presného riešenia na intervale [0, 1] s veľkosťou kroku 0.1.

```
Table[{t, yex[t]}, {t, 0, 1, 0.1}] // TableForm
```

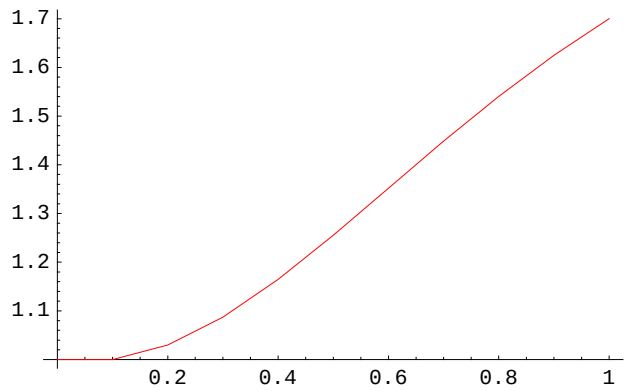
```
0       1
0.1     1.01482
0.2     1.05718
0.3     1.1217
0.4     1.20149
0.5     1.28981
0.6     1.38093
0.7     1.47042
0.8     1.55503
0.9     1.63261
1.      1.70187
```

Porovnanie presného a približného riešenia môžeme tiež ukázať aj graficky. Procedúra **grafeuler** v systéme *Mathematica* je naprogramovaná tak, aby nakreslila graf približného riešenia vypočítaného Eulerovou metódou červenou farbou. Nová procedúra grafeuler má rovnaký vstup, v ktorom použité parametre majú rovnaký význam ako mala predchádzajúca naprogramovaná procedúra.

```
grafeuler[f_, h_, ini_, a_, b_] :=
 Module[{y, t, ytable, c}, c = (b - a)/h; y[0] = ini; t[n_] := a + n h;
  y[n_] := y[n] = y[n - 1] + h f[t[n - 1], y[n - 1]]; ytable = Table[y[i], {i, 0, c}];
  ListPlot[Table[{t[i], ytable[[i + 1]]}, {i, 0, c}], Joined → True,
   PlotStyle → {RGBColor[1, 0, 0]}, PlotRange → All]]
```

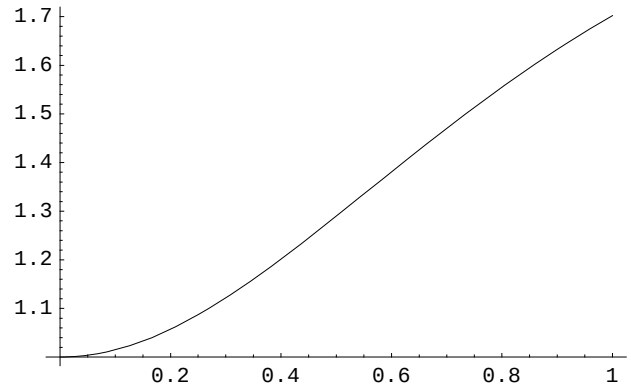
Približné aj presné riešenia sú nakreslené v jednom obrázku, aby ich bolo možné graficky porovnať

```
grafeuler[f, 0.1, 1, 0, 1]
```

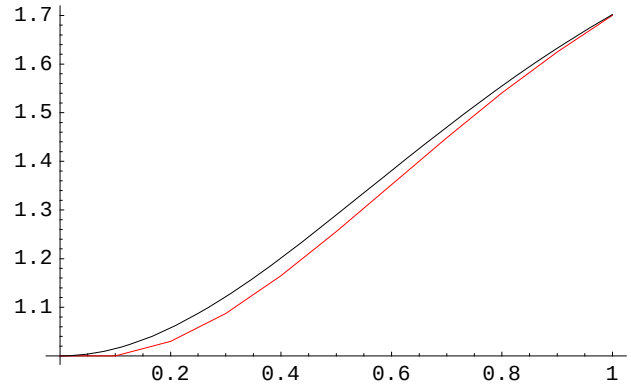



```
aproxi = %;
```

```
exact = Plot[yex[t], {t, 0, 1}]
```




```
Show[aproxi, exact]
```



Príklad 2.

Použitím Eulerovej metódy určte približné riešenie začiatočnej úlohy y' = 3 (y + t), y(0) = 1 na intervale [0,1], pre veľkosť kroku h = 0, 1. Získané riešenie nakreslite spolu s presným vypočítaným riešením pomocou príkazu ***DSolve****. Vzhľadom na veľkú chybu približného riešenia skúste veľkosť chyby redukovať zmenou veľkosti kroku vo výpočte.*

Riešenie

Presné riešenie začiatočnej úlohy získame použitím príkazu **DSolve.** Rovnako ako v predchádzajúcom príklade definujme najskôr funkciu z pravej strany DR a následne použime predprogramovanú procedúru.

```
f[t_, y_] := 3 (y + t)
```

```
euler[f, 0.1, 1, 0, 1]
```

```
0       1
0.1     1.3
0.2     1.72
0.3     2.296
0.4     3.0748
0.5     4.11724
0.6     5.50241
0.7     7.33314
0.8     9.74308
0.9     12.906
1.      17.0478
```

Presné riešenie zadaného začiatočného problému nájdeme pomocou príkazu **DSolve.**

```
solexacta22 = DSolve[{y'[t] == 3 (y[t] + t), y[0] == 1}, y[t], t]
```

$$\left\{\left\{y[t] \to \frac{1}{3}\left(-1 + 4\, e^{3\,t} - 3\,t\right)\right\}\right\}$$

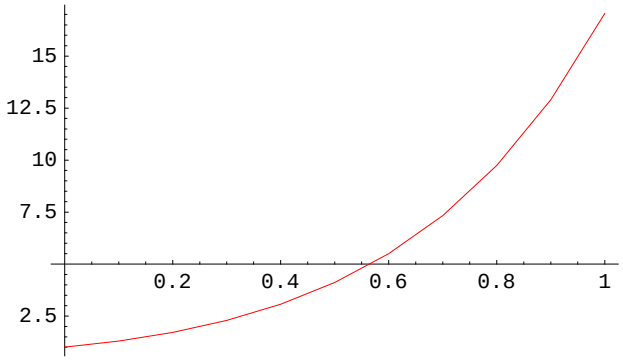
Z výsledku osamostatníme presné riešenie a obe riešenia graficky znázorníme

```
yex22[t_] = solexacta22[[1, 1, 2]]
```

$$\frac{1}{3}\left(-1 + 4\, e^{3\,t} - 3\,t\right)$$

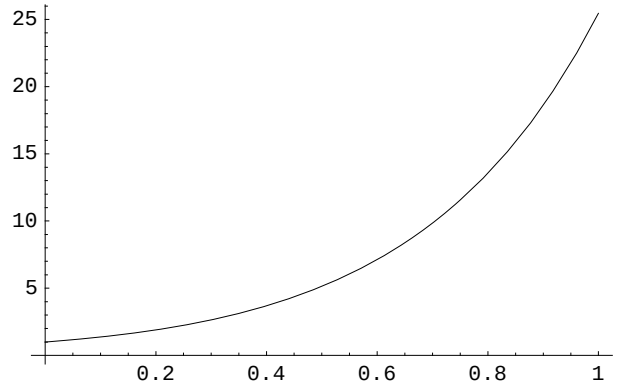
Graf obidvoch riešení vypočítame ako v predchádzajúcom príklade

```
grafeuler[f, 0.1, 1, 0, 1]
```

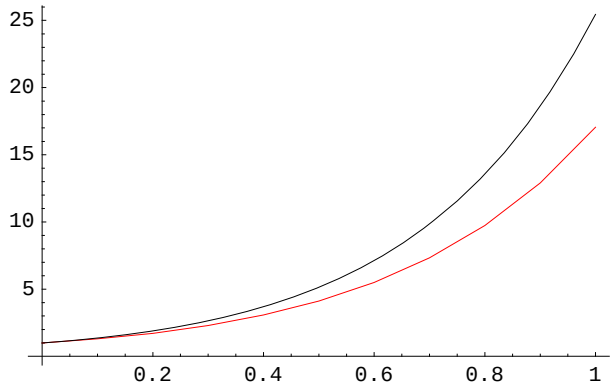



```
aproximada22 = %;
```

```
exacta22 = Plot[yex22[t], {t, 0, 1}]
```

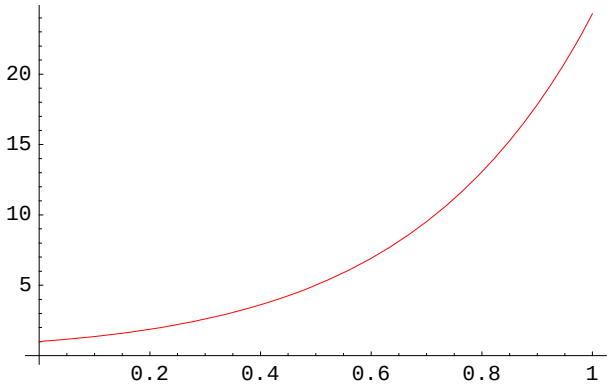



```
Show[aproximada22, exacta22]
```



Ako je vidieť z grafu, približné riešenie získané Eulerovou metódou nie je dostatočne presné. Je to preto, lebo presné riešenie rýchlo rastie vďaka exponenciálnemu členu $4\,e^{3t}$. Lepšiu aproximáciu presného riešenia získame zmenšením dĺžky kroku v Eulerovej metóde. Zlepšenie výsledku nám garantujú splnené podmienky riešiteľnosti tejto rovnice.

```
grafeuler[f, 0.01, 1, 0, 1]
```




```
aproximada22bis = %;

Show[aproximada22bis, exacta22]
```