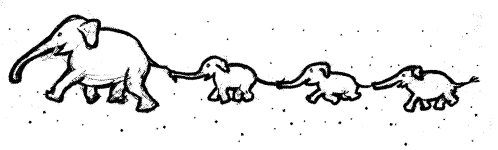
INFORMATIKA

P-čtveřice přirozených čísel

STANISLAV TRÁVNÍČEK

Přírodovědecká fakulta UP, Olomouc



Víme, co jsou to pythagorejské trojice přirozených čísel; např. známe trojici $(3; 4; 5)$; máme-li obdélník o stranách 3 cm a 4 cm, pak úhlopříčky tohoto obdélníku mají podle Pythagorovy věty celočíselnou velikost 5 cm, neboť $3^2 + 4^2 = 5^2$.

Vezměme si nyní čtveřici $(2; 3; 6; 7)$. Lehce se přesvědčíme, že $2^2 + 3^2 + +6^2 = 7^2$; máme-li tedy kvádr o hranách délky 2 cm, 3 cm a 6 cm, pak tělesové úhlopříčky tohoto kvádru mají (podle dvakrát použité Pythagorovy věty) celočíselnou velikost 7 cm. Čtveřice (a, b, c, d) přirozených čísel, pro něž platí

$$a^2 + b^2 + c^2 = d^2, \tag{1}$$

budeme pro stručnost dále nazývat p-čtveřice a v případě, že $D(a, b, c) = 1$, to budou *primitivní p-čtveřice*. (Je-li k společný dělitel čísel a, b, c, pak je samozřejmě také dělitelem čísla d.) Uveďme si nyní pro představu seznam primitivních p-čtveřic (pro $d \leq 35$, uspořádáno po řádcích podle rostoucího d, a, b).

1	2	2	3	2	3	6	7
1	4	8	9	4	4	7	9
2	6	9	11	6	6	7	11
3	4	12	13	2	5	14	15
2	10	11	15	1	12	12	17

8	9	12	17	1	6	18	19
6	6	17	19	6	10	15	19
4	5	20	21	4	8	19	21
4	13	16	21	8	11	16	21
3	6	22	23	3	14	18	23
6	13	18	23	9	12	20	25
12	15	16	25	2	7	26	27
2	10	25	27	2	14	23	27
7	14	22	27	10	10	23	27
3	16	24	29	11	12	24	29
12	16	21	29	5	6	30	31
6	14	27	31	6	21	22	31
14	18	21	31	1	8	32	33
4	7	32	33	4	17	28	33
7	16	28	33	8	8	31	33
8	20	25	33	17	20	20	33
1	18	30	35	6	10	33	35
6	17	30	35	15	18	26	35

Vidíme, že p-čtveřice vytvářejí mnohem bohatší svět než pythagorejské trojice. Např. existuje 7 různých primitivních p-čtveřic, kde $d = 33$, a k nim lze přidat ještě neprimitivní

6	18	27	33
11	22	22	33
18	18	21	33

Ale třeba pro $d = 99$ je primitivních p-čtveřic 19, pro $d = 355$ je primitivních p-čtveřic už 114.

Nyní se podívejme na to, k čemu lze p-čtveřic při výuce matematiky a informatiky využít.

1. Seznam primitivních p-čtveřic

Máme-li se zabývat užitím p-čtveřic, měli bychom mít k dispozici jejich seznam, např. seznam všech primitivních p-čtveřic v intervalu od urči-

tého zadaného $d = Od$ až po zadané $AzPo$, jako je výše pro $Od = 3$, $AzPo = 35$. Takový seznam pořídíme nejsnáze tím, že si jeho vytvoření naprogramujeme. Tedy na těch školách, kde se studenti učí programovat, je možno zadat studentům vyhotovení takového programu. Možností postupu je více a může být zajímavé, jaký postup zvolí vaši studenti. Pro čtenáře, kteří raději zadávají úpravy hotových programů, uvádíme *jedno z možných řešení* tisku p-čtveřic ve tvaru výše uvedeného seznamu.

```
program PCTV;
{vytvareni p-ctveric, trideno podle d,a,b
ukladano do souboru Ctverice.dat}

type Str5 = string[5];

var
  Pocet, Od, AzPo, Ind, L: Integer;
  Radek : string;              {radek tabulky}
  SI: array [1..4] of Integer; {cleny p-ctv}
  S2: array [1..4] of LongInt; {2. mocniny}
  SS: array [1..4] of Str5;    {cleny p-ctv jako texty}
  Fi: text;                    {vystupni soubor}

function NSD(U, V: Integer): Integer;
var Z: Integer;
begin  {nejvetsi spolecny delitel}
   Z := U mod V;
   if Z = 0 then NSD := V
  else
     NSD := NSD(V,Z)
end;   {NSD}

begin  {program}
  WriteLn('Primitivni p-ctverice (a,b,c,d)');
  Write('d od: '); ReadLn(Od);
  if Od < 3 then Od := 3;
  If (Od mod 2 = 0) then Inc(Od);
  Write('d do: '); ReadLn(AzPo); WriteLn;
  Assign(Fi, 'Ctverice.dat');
```

```
Rewrite(Fi);
Pocet := 1; SI[4] := Od;
repeat  {zakladni cyklus}
  S2[4] := SI[4] * SI[4];
  SI[1] := 1; S2[1] := 1;
  repeat
    SI[2] := SI[1]; S2[2] := S2[1];
    repeat
      S2[3] := S2[4] - S2[1] - S2[2];
      SI[3] := Trunc(Sqrt(S2[3]));  {mozne c}
      if (SI[3]*SI[3] = S2[3]) and (SI[3] >= SI[2]) then
      begin
        if NSD(NSD(SI[1], SI[2]), SI[3]) = 1 then    {*}
        begin
          if (Pocet MOD 2 = 1) then Radek := '';
          for Ind := 1 to 4 do
          begin
            Str(SI[Ind], SS[Ind]);
            L := Length(SS[Ind]);
            Radek := Radek + copy('    '+SS[Ind],L,5)
          end;
          if (Pocet mod 2 = 1) then
            Radek := Radek + '       '
          else
          begin
            WriteLn(Radek); WriteLn(Fi,Radek)
          end;
          if (Pocet mod 20 = 0) then
          begin ReadLn; WriteLn(Fi) end;
          Inc(Pocet)
        end
      end;
      Inc(SI[2]); S2[2] := SI[2] * SI[2]
    until S2[1] + S2[2] > 2 * S2[4] div 3;
    Inc(SI[1]); S2[1] := SI[1] * SI[1]
  until S2[1] > (S2[4] div 3);
  Inc(SI[4],2)
until SI[4] > AzPo;
```

```
  if (Pocet MOD 2 = 0) then
  begin
    WriteLn(Radek); WriteLn(Fi,Radek)
  end;
  WriteLn('Konec'); ReadLn;
  close(Fi)
end. {program}
```

Pokud bychom chtěli počítat všechny p-čtveřice, tj. nejen ty primitivní, stačí vynechat v programu řádek, kde je v komentáři hvězdička.

2. Seznam p-čtveřic jako zdroj výzkumné činnosti žáků

Seznam p-čtveřic se může stát zajímavým zdrojem pro skupinovou samostatnou práci; žáci mají objevit nějaké vlastnosti těchto čísel, vyslovit hypotézy a tyto hypotézy dokázat. Uveďme některé hypotézy, k nimž mohou žáci postupně dospět při své výzkumné činnosti.

H1. Je-li $(a; b; c; d)$ p-čtveřice, pak $(ka; kb; kc; kd)$, kde k je libovolné přirozené číslo, je také p-čtveřice.

H2. Ve všech primitivních p-čtveřicích $(a; b; c; d)$ je d liché.

H3. Ve všech primitivních p-čtveřicích $(a; b; c; d)$ je jedno z čísel a, b, c liché a další dvě jsou sudá.

H4. V každé primitivní p-čtveřici $(a; b; c; d)$ je dělitelné třemi buď jen d nebo právě dvě z čísel a, b, c.

Zajímavé také může být, zda a jak (matematicky vyspělejší) žáci dospějí k důkazu, že tyto hypotézy jsou vlastně matematickými větami.

Důkaz H1 je jednoduchý: Za uvedených předpokladů platí $(ka)^2+(kb)^2+$ $+(kc)^2 = k^2(a^2 + b^2 + c^2) = k^2d^2 = (kd)^2$, takže $(ka; kb; kc; kd)$ je p-čtveřice.

K dalším důkazům už žáci potřebují vědět něco o vlastnostech druhých mocnin čísel. Uveďme vlastnosti, které jsou tu potřebné:
1^0 Druhá mocnina sudého čísla je dělitelná čtyřmi.
2^0 Druhá mocnina lichého čísla dává při dělení čtyřmi zbytek 1.
3^0 Druhá mocnina čísla dělitelného třemi je dělitelná devíti (tedy i třemi).
4^0 Druhá mocnina čísla, které není dělitelné třemi dává při dělení třemi zbytek 1.

Důkaz H2: Uvažujme primitivní čtveřici $(a; b; c; d)$. Kdyby byla všechna tři čísla a, b, c sudá, byla by sudá levá strana rovnice (1) a tedy i pravá strana, tj. bylo by sudé také d, což je ve sporu s tím, že daná p-čtveřice je primitivní. Tedy u primitivních p-čtveřic tento případ nenastane.

Případ, že právě jedno z čísel a, b, c by bylo sudé a dvě lichá dává při dělení čtyřmi na levé straně součet zbytků 2, tedy při platnosti (1) by to znamenalo, že pravá strana (1) dává při dělení čtyřmi zbytek 2, ale to podle 1^0 a 2^0 není možné.

Případ, že žádné z čísel a, b, c není sudé, tj. všechna jsou lichá, dává při dělení čtyřmi na levé straně součet zbytků 3, tedy při platnosti (1) by to znamenalo, že pravá strana (1) dává při dělení čtyřmi zbytek 3, ale to podle 1^0 a 2^0 také není možné.

Zbývá tedy případ, že jedno z čísel a, b, c je liché a dvě jsou sudá, takže levá strana (1) při dělení čtyřmi dává vždy zbytek 1, takže pravá strana d^2 je lichá a je tak liché i číslo d.

Vidíme, že jsme současně dokázali i H3.

Důkaz H4 je možno vést podobně. Uvažujme opět primitivní čtveřici $(a; b; c; d)$. Kdyby byla všechna tři čísla a, b, c dělitelná třemi, bylo by i d dělitelné třemi, takže daná p-čtveřice by nebyla primitivní.

Kdyby právě jedno z čísel a, b, c bylo dělitelné třemi, dávala by podle 4^0 levé strana (1) při dělení třemi zbytek 2, takže zbytek 2 by dávalo i d^2, což podle 3^0 a 4^0 není možné.

Kdyby nebylo dělitelné třemi žádné z čísel a, b, c, byl by podle 4^0 součet zbytků na levé straně rovnice (1) roven 3, takže pravá strana a tím i d by byly dělitelné třemi.

Jestliže jsou dělitelné třemi právě dvě z čísel a, b, c, je podle 4^0 zbytek při dělení levé strany rovnice (1) roven 1 a tedy levá strana a tím ani pravá strana a d nejsou dělitelné třemi.

Tím je důkaz H4 ukončen.

3. Využití v analytické geometrii

Jde o využití při sestavování metrických úloh v analytické geometrii v prostoru. Když nám jde o první seznamování s takovými úlohami, je vhodné, aby studentům při počítání neodváděly pozornost iracionální odmocniny nebo zlomky. Proto je vhodné volit celočíselná řešení. Chceme-li si tedy při probírání metrických úloh v prostoru připravit úlohy „aby to dobře vyšlo“, můžeme využít svých znalostí p-čtveřic.

Úloha 1. Vytvořme tuto úlohu: Jsou dány body A, P, Q. Vypočtěte vzdálenost bodu A od přímky $p = PQ$.

Samozřejmě jde nyní o to, jak vhodně zvolit souřadnice bodů A, P, Q. Při sestavování úlohy můžeme postupovat třeba takto: Nazveme V ten bod přímky PQ, pro nějž bude platit $AV \perp PQ$. Řešením úlohy bude $|AV|$, výsledek si zvolíme. Vybereme si vhodnou p-čtveřici, např. $(2, 3, 6, 7)$ a sestavíme pro AV směrový vektor, např. $\boldsymbol{u}(-6, 3, -2)$ tak, že bude $V = A + \boldsymbol{u}$. Dále si zvolíme souřadnice bodu A, třeba $A[5; -1; -2]$, pak vypočteme $V[-1; 2; -4]$. Nyní musíme zvolit směrový vektor $\boldsymbol{v}$ přímky PQ tak, aby $\boldsymbol{u} \cdot \boldsymbol{v} = 0$, např. $\boldsymbol{v}(1; 0; -3)$. Pomocí bodu V a směrového vektoru $\boldsymbol{v}$ pak vytvoříme souřadnice bodů P, Q. Např.

$$P = V - 2\boldsymbol{v} = [-3; 2; 2],$$
$$Q = V + 2\boldsymbol{v} = [1; 2; -10].$$

Tak jsme vytvořili úlohu:

Jsou dány body $A[5; -1; -2]$, $P[-3; 2; 2]$, $Q[1; 2; -10]$. Vypočtěte vzdálenost bodu A od přímky $p = PQ$.

Řešením je $|AV| = \sqrt{(-1-5)^2 + (2+1)^2 + (-4+2)^2} = \sqrt{49} = 7$, což je 4. člen vybrané p-čtveřice.

Úloha 2. Vytvořme ještě tuto úlohu: Vypočtěte vzdálenost bodu M od roviny ϱ bez použití vzorce. Výsledek ověřte vzorcem.

Můžeme postupovat takto: Zvolíme p-čtveřici, např. $(6, 6, 7, 11)$ a první tři čísla (v libovolném uspořádání) vybavíme nějakými znaménky a zvolíme jako koeficienty rovnice roviny, např. $6x - 7y - 6z + D = 0$. V rovině zvolíme libovolný bod R (jako patu kolmice z bodu M), tj. např. $R[-2; 1; -4]$ a k němu jednoduchým výpočtem dostaneme D, zde $D = -5$. Protože vektor $\boldsymbol{n}(6; -7; -6)$ je směrový vektor normály, lehce si dopočítáme bod M, např. $M = R + 2\boldsymbol{n} = [10; -13; -16]$ Tak jsme vytvořili úlohu:

Vypočtěte vzdálenost bodu $M[10; -13; -16]$ od roviny ϱ: $6x - 7y - 6z - -5 = 0$ bez použití vzorce. Výsledek ověřte vzorcem.

Řešením je $|MR| = d = \sqrt{(-2-10)^2 + (1+13)^2 + (-4+16)^2} = \sqrt{484} = 22$, což je 4. člen dvojnásobku zvolené p-čtveřice $(12, 12, 14, 22) = 2 \cdot (6, 6, 7, 11)$.

Úloha 3. Jsou dány body A, B, C, D. Vypočtěte vzdálenost mimoběžek $a = AC$, $B = BD$.

Zvolíme p-čtveřici, např. 2, 6, 9, 11 (vzdálenost mimoběžek bude 11) a stanovíme směrový vektor osy o mimoběžek, např. $w = (2, -6, 9)$. Vytvoříme směrové vektory přímek a, b kolmé k w , např. $u = (3; 1; 0)$, $v = (3; -2; -2)$. Zvolíme libovolný průsečík P přímek a, o, např. $[-3; 2; -6]$, pak na b leží bod $Q = P + w = [-1; -4; 3]$. Nyní na přímce a zvolíme body A, C vyhovující vztahu $X = P + \lambda u$, např. $A[-6; 1; -6]$, $C[3; 4; -6]$ a podobně na přímce b body B, D vyhovující vztahu $X = Q + \mu v$, např. $B[5; -8; -1]$, $D[-4; -2; 5]$ a úloha je sestavena.

4. Generátor p-čtveřic

Čtenáře, kterému je známo, že pythagorejské trojice (a, b, c) lze generovat dvojicí přirozených parametrů m, n, $m > n$ vzorci

$$a = 2mn, \quad b = m^2 - n^2, \quad c = m^2 + n^2,$$

jistě napadne, zda existují podobné vzorce i pro generování p-čtveřic. Odpověď je „téměř kladná“. Uvažujme tři přirozené parametry m, n, p; pak užitím vzorců

$$a = 2mp, \quad b = 2np, \quad c = |p^2 - (m^2 + n^2)|, \quad d = p^2 + (m^2 + n^2) \qquad (2)$$

dostaneme p-čtveřici (a, b, c, d) (viz [1]). Toto je kladná část odpovědi. Ptáme-li se však, jestli pomocí vzorců (2) dostaneme všechny primitivní p-čtveřice, odpověď je záporná. Ze vzorců (2) je totiž zřejmé, že nutnou podmínkou je, aby bylo možno číslo d vyjádřit jako součet čtverců. A to nelze už u druhé p-čtveřice našeho seznamu, $(2, 3, 6, 7)$, neboť 7 nelze vyjádřit jako součet tří čtverců; v tomto je odpověď záporná.

Všimněme si nyní p-čtveřic $(4, 8, 19, 21)$, $(4, 13, 16, 21)$ a $(8, 11, 16, 21)$. Kromě zajímavosti, že p-čtveřice se mohou shodovat i ve dvou členech, uvažme, že $21 = 4^2 + 2^2 + 1^2$ a že rozklad čísla 21 na součet čtverců lze provést jedině tímto způsobem. Avšak parametry m, n, p zde lze volit jen třemi různými způsoby; počet permutací je sice 6, ale m a n jsou zaměnitelné, tj. po jejich záměně dostaneme tutéž p-čtveřici. Trojice $(m, n, p) = (2, 4, 1)$ tak dává stejný výsledek jako trojice $(4, 2, 1)$, a to první z uvedených p-čtveřic, trojice $(1, 4, 2)$ a $(4, 1, 2)$ dávají druhou p-čtveřici a $(1, 2, 4)$ s $(2, 1, 4)$ dávají třetí p-čtveřici. Tento výsledek

však není pravidlem, někdy všech 6 permutací parametrů vede na touž p-čtveřici; např. všechny permutace $(3, 4, 7)$ vedou na (neprimitivní) p-čtveřici $(24, 42, 56, 74)$.

Vraťme se však ještě k případu $d = 21$. V našem seznamu v úvodní části článku vidíme pro $d = 21$ ještě p-čtveřici $(4, 5, 20, 21)$; tu nelze dostat vzorci (2)? Lze, ale nepřímo. Násobíme-li tuto p-čtveřici číslem $k = 2$, dostaneme $(8, 10, 40, 42)$, kterou lze vygenerovat užitím parametrů $(1, 4, 5)$. Tedy: Některé primitivní p-čtveřice lze dostat jedině krácením jejich násobku vytvořeného vzorci (2), a toto je zmírnění výše uvedené záporné odpovědi. Již dříve uvedenou p-čtveřici $(2, 3, 6, 7)$ lze tak dostat pro $k = 2$ z parametrů $(1, 3, 2)$. Podobně p-čtveřici $(3, 8, 36, 37)$ zmíněnou v [1] lze dostat pro $k = 2$ a parametry $(3, 8, 1)$.

Na závěr si jako příklad ještě uveďme všech 6 primitivních p-čtveřic pro $d = 39$ a způsob jejich vytvoření užitím vzorců (2).

k	parametry	primitivní p-čtveřice
10	1, 17, 10	2, 19, 34, 39
5	1, 13, 5	2, 23, 29, 39
2	2, 5, 7	10, 14, 35, 39
10	13, 14, 5	13, 14, 34, 39
5	7, 11, 5	14, 22, 29, 39
10	11, 13, 10	19, 22, 26, 39

Čtenář zájemce (nebo jeho studenti) si nepříliš složitě může upravit program PCTV uvedený ve 2. části článku tak, aby dostával výsledky analýzy možnosti použití vzorců (2) pro generování příslušné p-čtveřice, jak je podává předchozí příklad (a jak to provedl i autor).

Děkuji *prof. Mgr. Radomíru Halašovi, Dr.* za cenný podnět k doplnění původně připraveného textu článku.

(Autorkou úvodní ilustrace je *Mgr. Jaroslava Čermáková* ze Suchých Lazců.)

Literatura

[1] *Weisstein, Eric W.*: Pythagorean Quadrille. From MathWorld A Wolfram Web Ressource. http://mathworld.wolfram.com/PythagoreanQuadruple.html

Využití software MAPLE při výuce fyziky 3

JAN HRDÝ

Přírodovědecká fakulta UP, Olomouc

(*Dokončení*)

6. Vytváření dynamických modelů s použitím programu MAPLE

Hned na úvod je třeba říci, že *Maple* nemá žádný „připravený formulář" pro zápis a vyhodnocení fyzikálního dynamického modelu. Vše co budeme pro vytvoření dynamického modelu potřebovat, si budeme muset sami naprogramovat [11]. Na druhé straně to má zase výhodu v tom, že nejsme nijak omezováni ze strany použitého software různými předdefinovanými postupy. *Maple* sice umí pomocí specializovaných příkazů také přímo provádět numerické řešení diferenciálních rovnic, ale to je v tomto okamžiku jiná kapitola. Na druhé straně je ovšem třeba podotknout, že programování v Maple je snadné a nepotřebuje rozsáhlé programátorské zkušenosti [12-14]. Co tedy potřebujeme zvládnout k tomu, abychom mohli úspěšně řešit dynamické modely? Kromě znalosti práce s matematickými funkcemi a poli hodnot (příkaz array) stačí pouze ovládat grafický výstup programu (v nápovědě programu část **Graphic/2-D**). Konkrétně jde o nakreslení grafu numericky získané závislosti pomocí jednotlivých bodů (parametr *style=point*), lomené čáry (parametr *style=patch*) nebo hladké křivky (parametr *style=line*) a kreslení libovolného počtu takových závislostí do jednoho grafu (příkaz *display*).

Ukázka možností realizace různých variant dynamických modelů principu použití radiouhlíkové metody při určování stáří rostlinných zbytků, přičemž jsou zde modely přesné (ale méně názorné) i modely hrubé, ale zato dobře pochopitelné i „zdravým selským rozumem" a realizovatelné i bez použití PC (pouze s kalkulačkou nebo jen s tužkou a papírem), což umožňuje lépe pochopit logiku tohoto způsobu modelování. Také je zde pozornost zaměřena na porovnávání přesnosti modelů v závislosti na počtu realizovaných aproximačních kroků.

Program 1: Program počítá počet nerozpadlých izotopů $^{14}C_6$ postupně na konci každého dílčího časového intervalu, který je v tomto

případě 1 rok, každá takto získaná hodnota se stává současně počáteční (vstupní) hodnotou pro další časový interval. Vypočítané údaje program ukládá do dvourozměrného pole (A1:2 × 10001 buněk).

```
n0:=1000000:
k:=0.000121:
A1:=array(1..2,0...10000):
   A1[1,0]:=0:
   A1[2,0]:=n0:
for i from 1 to 10000 do
   A1[1,i]:=i:
   A1[2,i]:=A1[2,i-1]*(1-k1):
end do:
t:=10000;n1:=round(A1[2,10000]); /tisk konečných hodnot
m1:=[[A1[1,100*m],A1[2,100*m]] $m=0..100]:/výstupní struktura
plot(m1,x=0..10000,y=0..1000000,style=line,colour=blue,
   axes=boxed);      /tisk grafu
```

I když je výpočet podrobný (10 000 kroků) a tedy velmi přesný, výstupní graf se z praktických důvodů tiskne pouze z rovnoměrně vybraných 100 bodů, čímž se samozřejmě nijak nesníží přesnosti zobrazení. Výsledný průběh je opět totožný s průběhem na obr. 7.

Program 2: Program umožňuje porovnávat výsledky získané pro různou délku základního časového intervalu (10 000 kroků o délce 1 rok – modře, 1000 kroků o délce 10 let – žlutě a 100 kroků o délce 100 let – zeleně) a zároveň je srovnat s výsledky získanými pomocí známého analytického vyjádření `n = n0*exp(-k*t)` – červeně (implicitní hodnota). Výběr grafických závislostí, které se mají vykreslit do společného grafu, závisí na argumentu funkce `display` (poslední řádek programu – možno upravovat). Ve všech čtyřech případech dostáváme prakticky stejnou křivku jako na obr. 7, protože všechny vzájemné odchylky jsou menší, než je tloušťka čáry v grafu (v rámci dosažitelné přesnosti zobrazení) na monitoru.

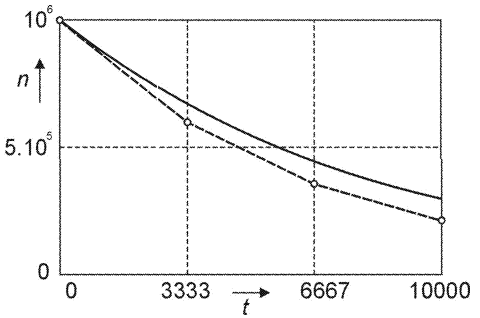
```
with(plots):    /grafická knihovna
n0:=1000000:

k1:=0.000121:
```

```
A1:=array(1..2,0...10000):
   A1[1,0]:=0:
   A1[2,0]:=n0:
for i from 1 to 10000 do
   A1[1,i]:=i:
   A1[2,i]:=A1[2,i-1]*(1-k1):
end do:
m1:=[[A1[1,100*m],A1[2,100*m]] $m=0..100]:
p1:=plot(m1,x=0..10000,y=0..1000000,style=line,colour=blue,
   axes=boxed):

k10:=0.00121: A2:=array(1..2,0...1000):
   A2[1,0]:=0:
   A2[2,0]:=n0:
for i from 1 to 1000 do
   A2[1,i]:=10*i:
   A2[2,i]:=A2[2,i-1]*(1-k10):
end do:
m2:=[[A2[1,10*m],A2[2,10*m]] $m=0..100]:
p2:=plot(m2,x=0..10000,y=0..1000000,style=line,colour=yellow,
   axes=boxed):

k100:=0.0121:
A3:=array(1..2,0...100):
   A3[1,0]:=0:
   A3[2,0]:=n0:
for i from 1 to 100 do
   A3[1,i]:=100*i:
   A3[2,i]:=A3[2,i-1]*(1-k100):
end do:
m3:=[[A3[1,m],A3[2,m]] $m=0..100]:
p3:=plot(m3,x=0..10000,y=0..1000000,style=line,colour=green,
   axes=boxed):

p4:=plot(1000000*exp(-0.000121*x),x=0..10000,y=0..1000000,
    style=line,axes=boxed):

display({p1,p2,p3,p4});
```

Program 3: Algoritmy použité v předchozích dvou programech jsou sice dostatečně přesné, ale vzhledem k použitému velkému počtu kroků se při jejich použití neobejdeme bez počítače. Pro vytvoření uceleného obrazu o problematice dynamického modelování však můžeme použít také algoritmy s malým počtem kroků, které mají sice o něco menší přesnost, ale jsou názornější, lépe korespondují s našimi praktickými zkušenostmi a vzhledem k malému počtu kroků vystačíme při jejich použití třeba jen s kalkulačkou (dají se použít třeba k rychlému odhadu průběhu sledovaného děje). Postup spočívá v tom, že sledované časové období, které je v našem případě 10000 let, rozdělíme pouze na několik málo (např. 1 ... 10) stejných intervalů, pro které provedeme výpočet stejně jako v předcházejícím případě (tj. vždy počítáme počet nerozpadlých atomů na konci určitého intervalu a takto získanou hodnotu bereme jako vstupní hodnotu pro následující časový interval, postupně se tak dopracujeme až ke konečné hodnotě počtu nerozpadlých atomů izotopu $^{14}C_6$. Počet dílčích časových intervalů je v tomto programu volitelný a je uložen v proměnné q (v prvním řádku programu). Program po spuštění vypíše počet zvolených časových úseků q, délku trvání jednoho tohoto úseku t v rocích, počet rozpadlých atomů k za dobu trvání jednoho úseku (vztažený na 1 milion vstupních atomů) a hlavně konečný počet nerozpadlých atomů N na konci celého sledovaného období (10000 let) získaný zvolenou aproximací a pro porovnání také konečný počet atomů získaný přesným výpočtem `n = 1000000*exp(-0.000121*10000)`. Pak program nakreslí příslušný graf, kde hledaná závislost je aproximována lomenou čarou. Body zlomu jsou označeny barevným kroužkem. Pro vizuální posouzení dosažené přesnosti aproximace je zakreslena také závislost daná analyticky (`n = n0*exp(-k*t)`). Ukázka grafického výstupu z tohoto programu pro $q = 3$ je na obr. 9.

```
q:=3;
with(plots):
t:=10000/q;
k:=1-t*121/1000000;
B2:=array(1..2,0...q):
   B2[1,0]:=0:
   B2[2,0]:=1000000:
for i from 1 to q do
   B2[1,i]:=t*i:
```

```
   B2[2,i]:=B2[2,i-1]*k:
end do:
N:=round(evalf(B2[2,q]));
n:=round(evalf(1000000*exp(-0.000121*10000)));
m2:=[[B2[1,m],B2[2,m]] $m=0..q]:
p2:=plot(m2,x=0..10000,y=0..1000000,style=patch,colour=green,
   axes=boxed);
p6:=plot(1000000*exp(-0.000121*x),x=0..10000,y=0..1000000,
   style=line,axes=boxed):

display({p2,p6});
```



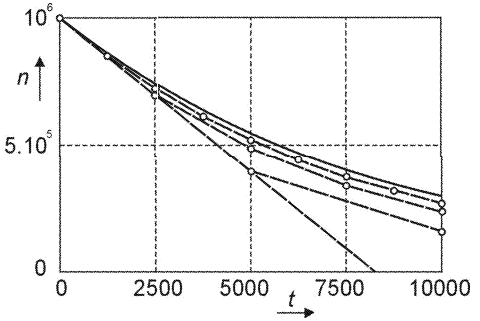
Obr. 9 Zjednodušená aproximace hledaného průběhu lomenou čarou

Program 4: Tento program umožňuje rychlé porovnání výsledků zjednodušené aproximace pro pevně zvolené hodnoty $q = 1, 2, 4, 8$ a 16 s přesným analytickým průběhem `n = n0*exp(-k*t)` závislosti počtu ještě nerozpadlých atomů n na čase t, viz obr. 10 (pro přehlednost byla vynechána lomená čára pro $q = 16$).

```
with(plots):
t1:=10000:
k1:=1-t1*121/1000000:
A1:=array(1..2,0...1):
   A1[1,0]:=0:
   A1[2,0]:=1000000:
```

```
for i from 1 to 1 do
   A1[1,i]:=t1*i:
   A1[2,i]:=A1[2,i-1]*k1:
end do:
m1:=[[A1[1,m],A1[2,m]] $m=0..1]:
p1:=plot(m1,x=0..10000,y=0..1000000,style=line,colour=green,
   axes=boxed):
p10:=plot(m1,x=0..10000,y=0..1000000,style=point,
    symbol=circle,colour=green,axes=boxed):

t2:=10000/2:
k2:=1-t2*121/1000000:
A2:=array(1..2,0...2):
   A2[1,0]:=0:
   A2[2,0]:=1000000:
for i from 1 to 2 do
   A2[1,i]:=t2*i:
   A2[2,i]:=A2[2,i-1]*k2:
end do:
m2:=[[A2[1,m],A2[2,m]] $m=0..2]:
p2:=plot(m2,x=0..10000,y=0..1000000,style=line,colour=green,
   axes=boxed):
p20:=plot(m2,x=0..10000,y=0..1000000,style=point,
    symbol=circle,colour=green,axes=boxed):

t3:=10000/4:
k3:=1-t3*121/1000000:
A3:=array(1..2,0...4):
   A3[1,0]:=0:
   A3[2,0]:=1000000:
for i from 1 to 4 do
   A3[1,i]:=t3*i:
   A3[2,i]:=A3[2,i-1]*k3:
end do:
m3:=[[A3[1,m],A3[2,m]] $m=0..4]:
p3:=plot(m3,x=0..10000,y=0..1000000,style=line,colour=green,
   axes=boxed):
p30:=plot(m3,x=0..10000,y=0..1000000,style=point,
```

```
    symbol=circle,colour=green,axes=boxed):

t4:=10000/8:
k4:=1-t4*121/1000000:
A4:=array(1..2,0...8):
   A4[1,0]:=0:
   A4[2,0]:=1000000:
for i from 1 to 8 do
   A4[1,i]:=t4*i:
   A4[2,i]:=A4[2,i-1]*k4:
end do:
m4:=[[A4[1,m],A4[2,m]] $m=0..8];
p4:=plot(m4,x=0..10000,y=0..1000000,style=line,colour=green,
   axes=boxed):
p40:=plot(m4,x=0..10000,y=0..1000000,style=point,
    symbol=circle,colour=green,axes=boxed):

t5:=10000/16:
k5:=1-t5*121/1000000:
A5:=array(1..2,0...16):
   A5[1,0]:=0:
   A5[2,0]:=1000000:
for i from 1 to 16 do
   A5[1,i]:=t5*i:
   A5[2,i]:=A5[2,i-1]*k5:
end do:
m5:=[[A5[1,m],A5[2,m]] $m=0..16]:
p5:=plot(m5,x=0..10000,y=0..1000000,style=line,colour=green,
   axes=boxed):
p50:=plot(m5,x=0..10000,y=0..1000000,style=point,
   symbol=circle,colour=green,axes=boxed):
p6:=plot(1000000*exp(-0.000121*x),x=0..10000,y=0..1000000,
   style=line,axes=boxed):

display({p1,p2,p3,p4,p5,p10,p20,p30,p40,p50,p6});
```



Obr. 10 Porovnání přesnosti různých „stupňů“ zjednodušené aproximace

Je zajímavé, že i tento zjednodušený dynamický model poskytuje již pro q = 8 poměrně slušnou představu o průběhu hledané závislosti.

7. Závěr

Cílem toho článku bylo srozumitelným způsobem poskytnout ucelený přehled možností programu *Maple* při dynamickém modelování vybrané skupiny fyzikálních problémů tak, aby vytvořené modely mohly býti přímo využity ve výuce fyziky na SŠ. Teoretický úvod k zvolené problematice může být využit také ve specializovaných fyzikálních seminářích nebo při přípravě studentů na FO. Případné konzultace nebo zdrojové kódy zde prezentovaných nebo i jiných programů si lze vyžádat u autora tohoto příspěvku (hrdy@prfnw.upol.cz).

Literatura je uvedena za 1. částí článku.

Oprava: Správný tvar vztahu (37) ve 2. části (č. 3, s. 176) je

$$s_0 = \lim_{t\to\infty} s = \lim_{t\to\infty} \frac{mv_0}{\mu}\left(1 - \mathrm{e}^{-\frac{\mu t}{m}}\right) = \lim_{t\to\infty} \frac{mv_0}{\mu}\left(1 - \frac{1}{\mathrm{e}^{\frac{\mu t}{m}}}\right) = \frac{mv_0}{\mu}.$$