

# **Evolučné algoritmy**

## **Case study**

### **Téma 8: Evolučná stratégia (1+1)**

Vypracoval: Martin Kubačák  
Informatika, 4.ročník, MFF UK  
Zimný semester 1999/2000  
E-mail: 6kubacak@st.fmph.uniba.sk

## Zadanie

Metódu evolučnej stratégie (1+1) aplikujte na hľadanie globálneho minima funkcie

$$f(x) = 0,993851231 + e^{-0,01x^2} \sin(10x) \cos(8x)$$

pre  $\mathbf{x} \in [-10; 10]$ , ktorá je zovšeobecnená pre viacrozmerný prípad takto

$$F(x_1, x_2, \dots, x_n) = \sum_{i=1}^n f(x_i).$$

Použite pravidlo zmeny  $\sigma$  1/5 úspešnosti a nakreslite graf, kde na vodorovnej osi bude počet generácií a na zvislej aktuálna funkčná hodnota rodiča.

## Evolučná stratégia (1+1) - jeden rodič a jeden potomok

Evolučná stratégia patrí historicky medzi prvé úspešné pravdepodobnostné algoritmy. Bola navrhnutá už počiatkom 60-tych rokov Rechenbergom a Schwefelom. Vychádza zo všeobecných predstáv prirodzeného výberu, no o mnoho vážnejších ako je to napríklad u genetického algoritmu. Je založená na myšlienke náhodných zmien parametrov, podobne ako je tomu u mutácií. Na rozdiel od väčšiny ostatných stochastických metód, evolučná stratégia nie je založená na binárnej reprezentácii premenných, ale manipuluje priamo s "reálnou" reprezentáciou premenných.

Vo všeobecnosti v nej ide o minimalizáciu funkcie  $f(\mathbf{x})$ , kde  $\mathbf{x} \in \mathbf{R}^n$ , teda funkcia má ako nezávislé premenné zložky  $n$ -rozmerného vektora  $\mathbf{x}$  reálnych čísel, pričom prípustné hodnoty jednotlivých zložiek môžu byť obmedzené, napr.  $\mathbf{x} \in [a; b]^n$ . Formálne sa to dá zapísať

$$x_{opt} = \underset{x \in [a, b]^n}{\arg \min} f(x).$$

Základom evolučnej stratégie je predpis

$$\mathbf{x}' = \mathbf{x} + \mathbf{N}(0, \sigma),$$

ktorý "mutuje" aktuálne riešenie  $\mathbf{x}$  na nové riešenie  $\mathbf{x}'$ , kde  $\mathbf{N}(0, \sigma)$  je vektor nezávislých náhodných čísel distribuovaných podľa normálneho (Gaussovského) rozdelenia s nulovou strednou hodnotou a štandardnou odchýlkou  $\sigma$ , teda pravdepodobnosť, že sa objavia menšie čísla je vyššia ako pravdepodobnosť, že sa objavia väčšie čísla. Čím väčšia je štandardná odchýlka, tým väčšia bude najskôr vzdialenosť medzi rodičom a vygenerovaným potomkom.

Z rodiča sa teda generuje jeden potomok, ktorý sa od neho líši skôr málo ako veľa. Riešenie  $\mathbf{x}'$  je akceptované (úspešné), keď platí  $f(\mathbf{x}') < f(\mathbf{x})$ . Keď potomok odpovedá horšiemu riešeniu ako rodič, vymaže sa potomok a generuje sa iný. To sa robí tak dlho, až kým potomok neodpovedá lepšiemu riešeniu ako rodič a nezaujme jeho miesto, čo prakticky odpovedá gradientovému prístupu alebo hillclimbingu.

Stratégia, kedy sa do ďalšej generácie vyberú lepší jedinci (tu iba jeden jedinec) tak z populácie rodičov, ako aj z populácie potomkov, sa nazýva PLUS stratégia. Pre jedného rodiča a jedného potomka je to teda (1+1) stratégia (tiež sa jej hovorí dvojčlenná). Táto patrí medzi prvé evolučné stratégie, ktoré používali ako jediný rekombinačný operátor mutáciu. Rechenberg na základe svojich

pokusov na funkciách  $f(\mathbf{x}) = \sum_{i=1}^n x_i^2$  a  $f(\mathbf{x}) = -\sum_{i=1}^n x_i$  odvodil predpis na zmenu veľkosti "mutácie",

teda vzdialenosti potomka od rodiča, a na "zmenu tejto zmeny". Smerodajná odchýlka  $\sigma$  sa v priebehu evolučnej stratégie mení podľa pravidla 1/5 úspešnosti. Nech  $\varphi(k)$  je koeficient úspešnosti definovaný ako pomer počtu úspešných mutácií v priebehu posledných  $k$  iterácií k počtu  $k$  iterácií, z ktorých bola úspešnosť meraná, potom

$$\sigma' = \begin{cases} c_d \cdot \sigma & (\varphi(k) < 1/5) \\ c_i \cdot \sigma & (\varphi(k) > 1/5) \\ \sigma & (\varphi(k) = 1/5) \end{cases},$$

kde  $c_i > 1$  a  $c_d < 1$  riadia zväčšovanie, resp. zmenšovanie štandardnej odchýlky. V literatúre sú tieto koeficienty špecifikované  $c_d = 0.82$  a  $c_i = 1/c_d = 1.22$ . Intuitívne zdôvodnenie pravidla 1/5 úspechu je zvýšenie efektívnosti prehľadávania - pri úspešnosti by malo prehľadávanie pokračovať vo väčších krokoch, pri neúspešnosti by mali byť kroky kratšie. Toto pravidlo odvodené pre uvedené dve

funkcie sa všeobecne používa aj pre ďalšie neznáme funkcie, pretože pre mnohé problémy jeho používanie pomáhalo udržať najrýchlejší postup k optimu.

```

1  x:=náhodne generovaný vektor reálnych premenných;
2  t:=0;
3  sigma:=sigma_ini;
4  x_best:=x;
5  while t<t_max do begin
6    inc(t);
7    i:=0;
8    k:=0;
9    while i<i_max do begin
10     inc(i);
11     perturbation(x, x_new, sigma);
12     if f(x_new)<f(x) then begin
13       inc(k);
14       x:=x_new;
15       if f(x)<f(x_best) then
16         x_best:=x;
17     end;
18   end;
19   if k/i_max<0.2 then
20     sigma:=c_d*sigma
21   else if k/i_max>0.2 then
22     sigma:=c_i*sigma;
23 end;
```

**Algoritmus 1.** Evolučná stratégia (1+1).

Premenná  $t$  je počítadlo epoch evolučnej stratégie. Algoritmus obsahuje dva WHILE-cykly, vonkajší a vnútorný. Vo vnútornom cykle sa pre dané  $\sigma$  opakuje elementárny krok evolučnej stratégie  $i_{max}$ -krát, pričom premenná  $k$  zaznamenáva úspešnosť mutácií v tomto vnútornom cykle. Vonkajší cyklus, s počítadlom  $t$ , aplikuje pre rôzne hodnoty odchýlky  $\sigma$  evolučnú stratégiu  $t_{max}$ -krát. Štandardná odchýlka  $\sigma$  je v 3. riadku inicializovaná hodnotou  $\sigma_{ini}$ . V 11. riadku sa vykonáva modifikácia riešenia  $x$  pomocou generátora náhodných čísel s nulovou strednou hodnotou a so štandardnou odchýlkou  $\sigma$  (pozri ďalej). Voľba základných parametrov evolučnej stratégie, teda  $t_{max}$ ,  $\sigma_{ini}$  a  $i_{max}$ , si vyžaduje určité experimentovanie, pomocou ktorého sa potom tieto konštanty nastaví. Zvyčajne je  $\sigma_{ini}$  blízke jednotke a konštanta  $i_{max}$  sa rovná rádovo tisícim. V 19. až 22. riadku sa vykonáva zmena  $\sigma$  podľa pravidla 1/5 úspešnosti. Je ale vždy potrebné ohraničiť zmeny smerodajnej odchýlky tak, aby nedosiahla nulovú hodnotu pre danú presnosť.

V nasledujúcom algoritme je zachytená mutácia vektora  $x$  po zložkách o náhodnú premennú s rozdelením  $N(0, \sigma)$  (v 5. riadku) a následne opravný proces, ak nová hodnota nespadá do svojho vymedzeného intervalu.

```

1  procedure perturbation(var x, x_new: vector; sigma: real);
2  var j: integer;
3  begin
4    for j:=1 to n do begin
5      x_new[j]:=x[j]+sigma*gauss_random;
6      while x_new[j]>x_max do
7        x_new[j]:=2*x_max-x_new[j];
8      while x_new[j]<x_min do
9        x_new[j]:=2*x_min-x_new[j];
10   end;
11 end;
```

**Algoritmus 2.** "Zrkadlenie hranice" - úprava hodnôt navrhovaného riešenia, keď algoritmus prekročí definičný obor optimalizovanej funkcie.

Algoritmus využíva tzv. zrkadlový obraz, kedy sa umiestňuje hodnota  $x_i$  na opačnú stranu hranice intervalu, v rovnakej vzdialenosti od hranice. Ak sa hodnota  $x_i$  dostane týmto presunom za hranice intervalu na opačnej strane, postupuje sa rovnako.

Stojí ešte za zamyslenie, prečo je Gaussovská distribúcia lepšia, resp. výhodnejšia ako prehadzovanie bitov v binárnom kóde. Pri štandardnom binárnom kódovaní nemajú mutované čísla dobrú distribúciu okolo nuly. Pokusy ukázali, že tento fenomén sa vyskytuje vo všeobecnosti aj pre veľké počty mutácií (možno by stálo za zamyslenie, ako to bude pri použití Grayovho kódu). Odpoveď na otázku, ako dostať náhodné čísla s normálnym rozložením podľa Gaussovskej distribúcie dáva nasledujúci algoritmus.

```

1 switch:=0;

2 function gauss_random: real;
3   var a, b: real;
4   begin
5     if switch=0 then begin
6       switch:=1;
7       a:=sqrt(-2*ln(real_random));
8       b:=2*pi*real_random;
9       gauss_random:=a*cos(b);
10      save_gauss_random:=a*sin(b);
11    end else begin
12      switch:=0;
13      gauss_random:=save_gauss_random;
14    end;
15  end;

```

**Algoritmus 3.** Generátor náhodnej premennej s normálnym rozdelením s nulovou strednou hodnotou a jednotkovou smerodajnou odchýlkou (podľa Schwefela).

Uvedený algoritmus vracia náhodne vygenerovanú premennú s normálnym (Gaussovským) rozdelením s nulovou strednou hodnotou  $\mu$  a jednotkovou smerodajnou odchýlkou  $\sigma$ . Používa pritom vygenerovanú premennú z intervalu  $(0;1]$  s uniformným rozložením, teda každé číslo z tohto intervalu sa vyskytne s rovnakou pravdepodobnosťou. Pokiaľ je k dispozícii oba generátory náhodných prirodzených čísel (prípade väčšiny programovacích jazykov Pascal), treba vygenerovať také číslo, previesť na reálne a vydeliť ho maximálnou možnou hodnotou dosiahnuteľnou generátorom, čo by malo odpovedať výsledku funkcie *real\_random*. Funkcia vlastne vyrába vždy dve náhodné premenné s normálnym rozložením, pričom prvú dáva ako výsledok hneď, druhú pri ďalšom volaní funkcie (prepínač *switch*). Existuje ešte jeden známy algoritmus na generovanie premennej s normálnym rozdelením s nulovou strednou hodnotou a jednotkovou smerodajnou odchýlkou, ktorý nepoužíva goniometrické funkcie (ten však nebudem uvádzať, v programe som použil hore popísaný).

Zastavím sa ešte nachvíľku pri tomto algoritme. Štandardne sa berie normálne rozdelenie  $\mathbf{N}(\mu, \sigma)$ , ktorého hustotnú funkciu možno opísať ako

$$f(x): y = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-0,5\left(\frac{x-\mu}{\sigma}\right)^2},$$

kde  $\mu$  je stredná hodnota premennej  $x$  a  $\sigma$  je jej smerodajná odchýlka. Po úpravách možno vyjadriť  $x$  nasledovne

$$x = \sigma \sqrt{-2 \ln(\sqrt{2\pi\sigma^2} y)} + \mu.$$

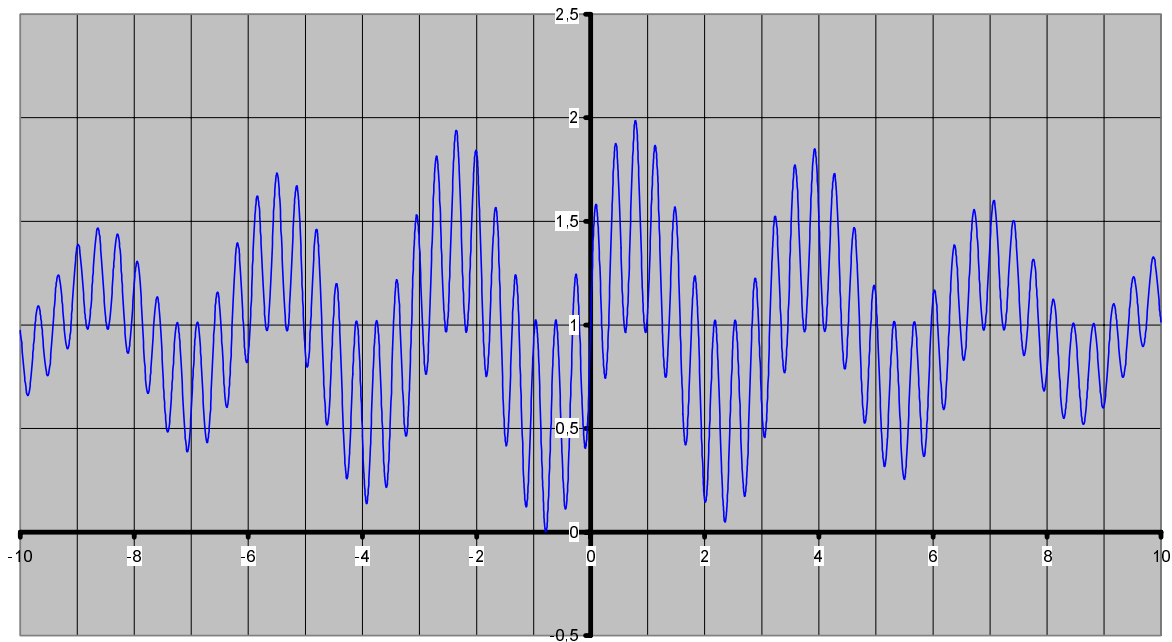
V argumente logaritmu netreba uvažovať  $2\pi\sigma^2$  pod odmocninou, pretože algoritmus pracuje s náhodne vygenerovanou premennou z intervalu  $(0;1]$  (v 7. riadku), a keďže pre potreby evolučnej stratégie sa kladie stredná hodnota rovná nule, teda  $\mu = 0$ , vzťah sa tým zjednoduší na

$$x = \sigma \sqrt{-2 \ln(y)},$$

kde  $y \in (0;1]$ . Teda stačí vygenerovať náhodnú premennú s normálnym rozdelením  $\mathbf{N}(0,1)$  a tú potom prenásobiť  $\sigma$ , čím získam premennú s rozdelením  $\mathbf{N}(0,\sigma)$  (pozri algoritmus 2). Nasledujúci

obrázok poskytuje predstavu o hustote náhodne generovaných premenných s normálnym rozdelením pre rôzne hodnoty odchýlky  $\sigma$ .

Graf funkcie  $f(x)$  na intervale  $[-10;10]$



Graf 1. Normálne (Gaussovské) rozdelenie pravdepodobnosti náhodnej premennej.

Pre evolučnú stratégiu je dokázané, že potenciálne poskytuje globálny extrém optimalizovanej funkcie. Za predpokladu regulárnosti optimalizovaného problému (spojitosť, uzatvorenosť množiny tvoriacej definičný obor funkcie apod.) je možné dokázať konvergenčný teorém, ktorý tvrdí, že globálne optimum bude dosiahnuté s jednotkovou pravdepodobnosťou, nehovorí však, za koľko generácií.

**Poznámka**

Pri vypracovaní tejto teoretickej časti som vychádzal z materiálov k prednáške Evolučné algoritmy, prednášanej v zimnom semestri 1999/2000 na MFF UK, ktoré sa nachádzajú na internetovej stránke [ftp://math.chtf.stuba.sk/vlado/evol\\_alq\\_mff/](http://math.chtf.stuba.sk/vlado/evol_alq_mff/).

**Výsledky**

Vyššie popísaný algoritmus evolučnej stratégie (1+1) som implementoval v programovacom jazyku Pascalu. Pre účely neskoršej sumarizácie som pri každej "pozitívnej" zmene funkčnej hodnoty vypisoval generáciu, vektor premenných a príslušnú funkčnú hodnotu.

Samotná funkcia  $f(x)$  s predpisom

$$f(x) = 0,993851231 + e^{-0,01x^2} \sin(10x)\cos(8x)$$

pre  $x \in [-10;10]$  je znázornená na grafe 2 na nasledujúcej strane.

Z grafu vidieť, že  $f(x)$  nadobúda globálne minimum blížiac sa k nule na zadanom intervale blízko hodnoty  $x = -1$ . Numerickými iteráciami som určil približnú hodnotu ako  $x_{opt} = -0,78530243234$  (upozorňujem však na možnú nepresnosť výpočtového zariadenia). Z charakteru zovšeobecnenej funkcie  $F(x)$  zadanej ako

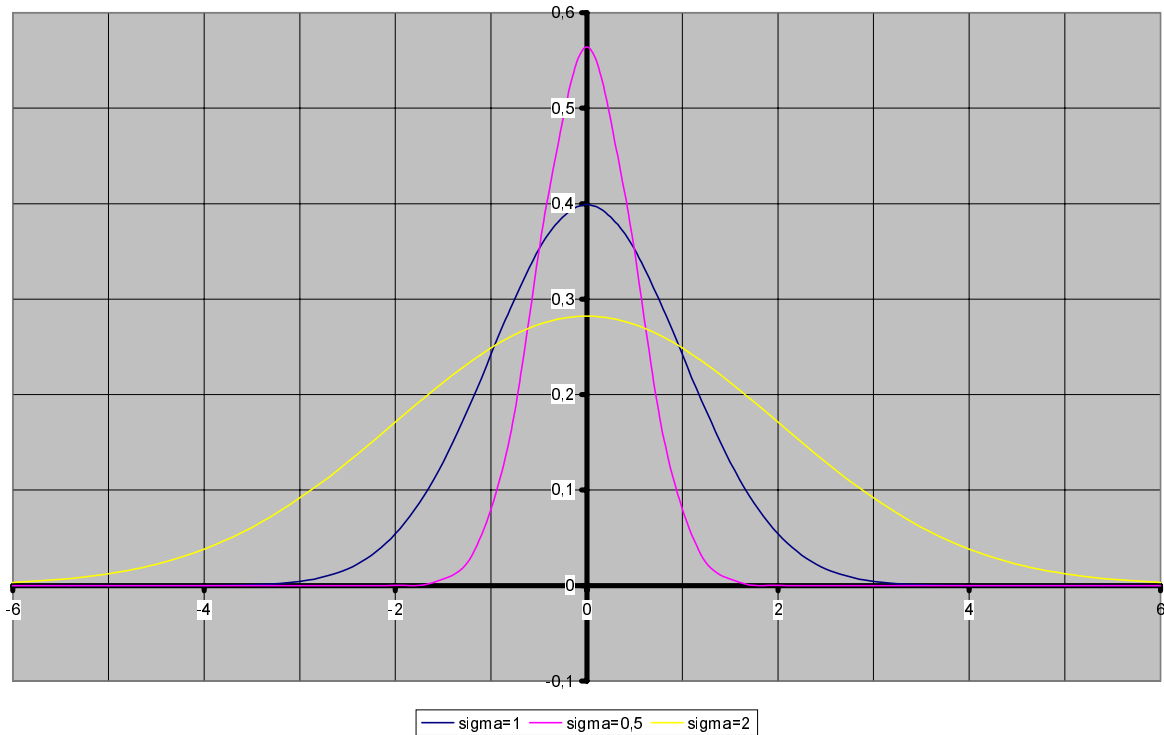
$$F(x_1, x_2, \dots, x_n) = \sum_{i=1}^n f(x_i)$$

vyplyva, že globálne optimum pre  $n$ -rozmerný vektor  $x$  bude mať taktiež funkčnú hodnotu blížiacu sa k nule pre  $x_{i\ opt} = -0,78530243234$  pre každú zložku  $i$  vektora  $x_{opt}$

Po určitom experimentovaní som nastavil koeficienty  $t_{max} = 127$  a  $i_{max} = 32767$  bez možnosti modifikácie, a koeficienty  $\sigma_{ini} = 2,19$  a  $c_d = 0,82$ , tie však možno pri spustení programu modifikovať.

Tieto nastavenia som použil vo všetkých popisovaných pokusoch s 1D, 2D a 3D vektormi (po 5 pokusov), a tiež v 3 pokusoch popísaných pre 4D, 5D a 10D vektory, pre zvyšné 2 pokusy so 4D, 5D a 10D veličinami som použil  $\sigma_{ini} = 3,78$  a  $c_d = 0,93$ . Hodnota koeficientu  $c_i$  sa nastaví ako  $c_i = 1/c_d$ . Maximálnu hodnotu  $n$  som stanovil na 32, aj keď pre väčšie hodnoty ako 10 som program vzhľadom sa obrovský počet operácií a veľkú výpočtovú a časovú náročnosť (pre väčšie  $n$  výpočet bežal aj niekoľko hodín) nespúšťal.

Normálne (Gaussovské) rozdelenie pravdepodobnosti so stredom v nule a odchýlkou sigma

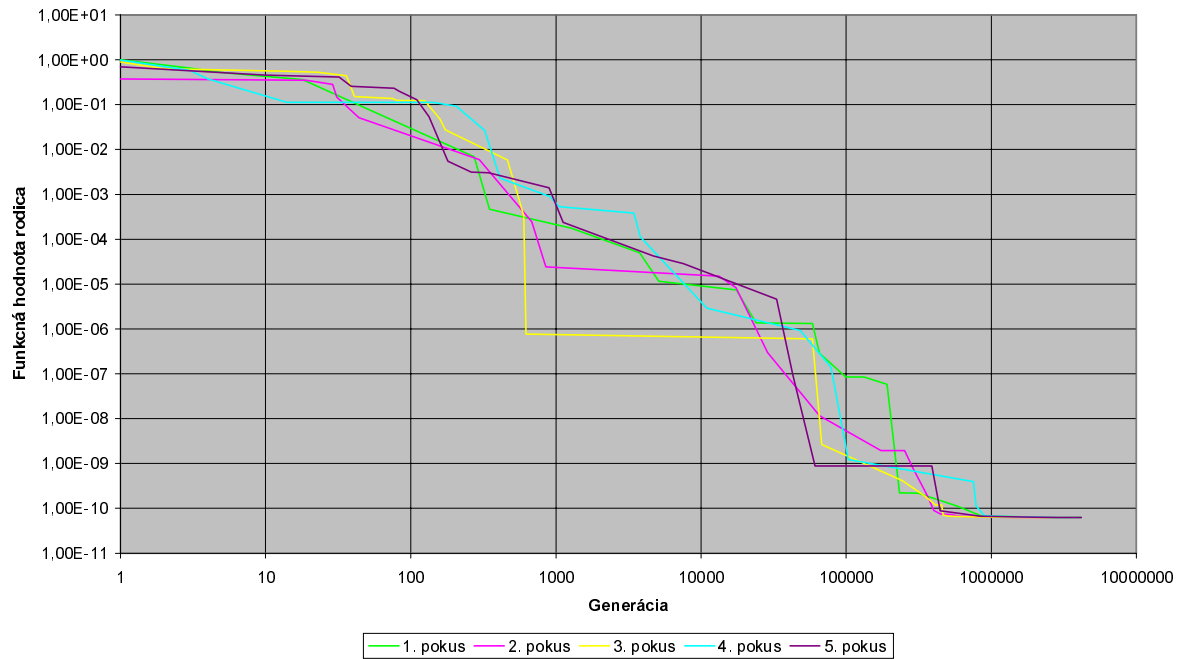


Graf 2. Funkcia  $f(x)$  na intervale  $[-10;10]$ .

V nasledujúcich riadkoch sa budem zaoberať interpretáciou výsledkov, ktoré som získal pri pokusoch s generovanými veličinami pre rôzne rozmery, konkrétne pre 1- až 5-rozmerné a 10-rozmerné vektory.

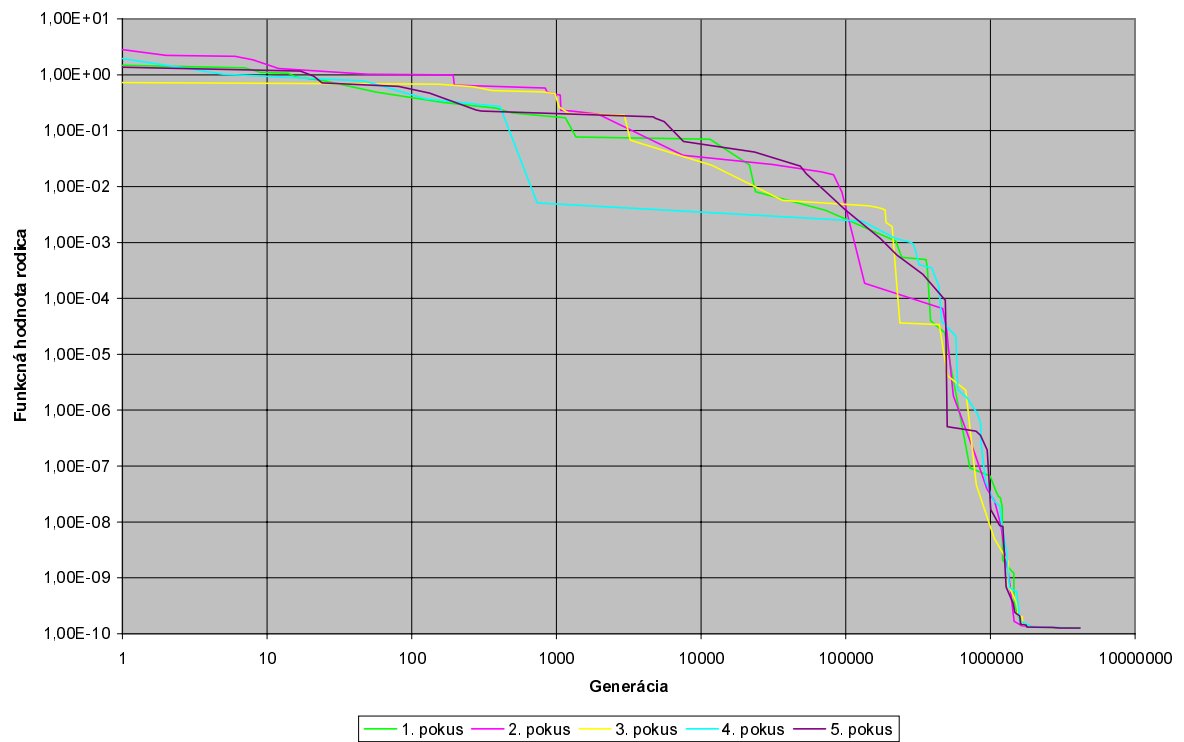
Jednorozmerné premenné predstavovali najjednoduchší prípad. Hoci z grafu funkcie  $f(x)$  sa zdá, že funkcia sa správa veľmi nepríjemne, ako vidieť z grafu 3 a priloženej tabuľky 1 (pozri časť **Tabuľky** na konci) algoritmus pre takéto veličiny nemal problémy v 5 zaznamenaných prípadoch (dalo by sa diskutovať o tom, či je to dostatočný počet krát) konvergovať k optimálnemu riešeniu s funkčnou hodnotou blížiacou sa k 0. Rovnako to dopadlo aj v prípade dvojrozmerných veličín, ako vidieť z priloženého grafu 4 a tabuľky 2. Pre tieto dva prípady bola úspešnosť algoritmu 100%. Pri trojrozmernej premennej v jednom pokuse z piatich program počas svojho behu nedokázal lokalizovať hodnotu  $x_{opt}$  (resp. jednej jeho zložky, pozri tabuľku 3), v ktorom funkcia  $f(x)$  nadobúda globálne minimum, v zvyšných štyroch pokusoch sa mu to už podarilo, čo je zachytené v grafe 5 i tabuľke 3. Program mal teda v tomto prípade úspešnosť v hľadaní optima 80%. Z zvyšných pokusov, teda pre 4-, 5- a 10-rozmerné premenné sa mu to však už nepodarilo ani raz, úspešnosť programu v týchto pokusoch bola 0%. Odpoveď na otázku, prečo to tak dopadlo treba hľadať jednak v správaní funkcií  $f(x)$  a  $F(x)$ , ale nezanedbateľný vplyv má zrejme aj znižovanie smerodajnej odchýlky pre generovanie nových potomkov pri neúspešnom hľadaní. Algoritmus "zamrzne" v bode pomerne vzdialenom od optima, odkiaľ sa pri generovaní nízkych náhodných čísel dostane niekde inde len s malou pravdepodobnosťou. Aj keď pre menší počet rozmerov tento prípad nastáva len sporadicky (spomínaný jeden pokus pri 3D vektoroch), zdá sa, že pri vektoroch s viac rozmermi je to ťažká prekážka brániaca vyššej úspešnosti algoritmu (je možné, že je to len dôsledok nepríjemného charakteru funkcie  $f(x)$ ). Možno by pomohlo iné nastavenie parametrov  $t_{max}$ ,  $\sigma_{ini}$  a  $i_{max}$  evolučnej stratégie. Na základe piatich pokusov sa však len ťažko dajú urobiť nejaké serioznejšie závery.

Zmeny funkčnej hodnoty rodiča v závislosti od generácie pre 1D vektory



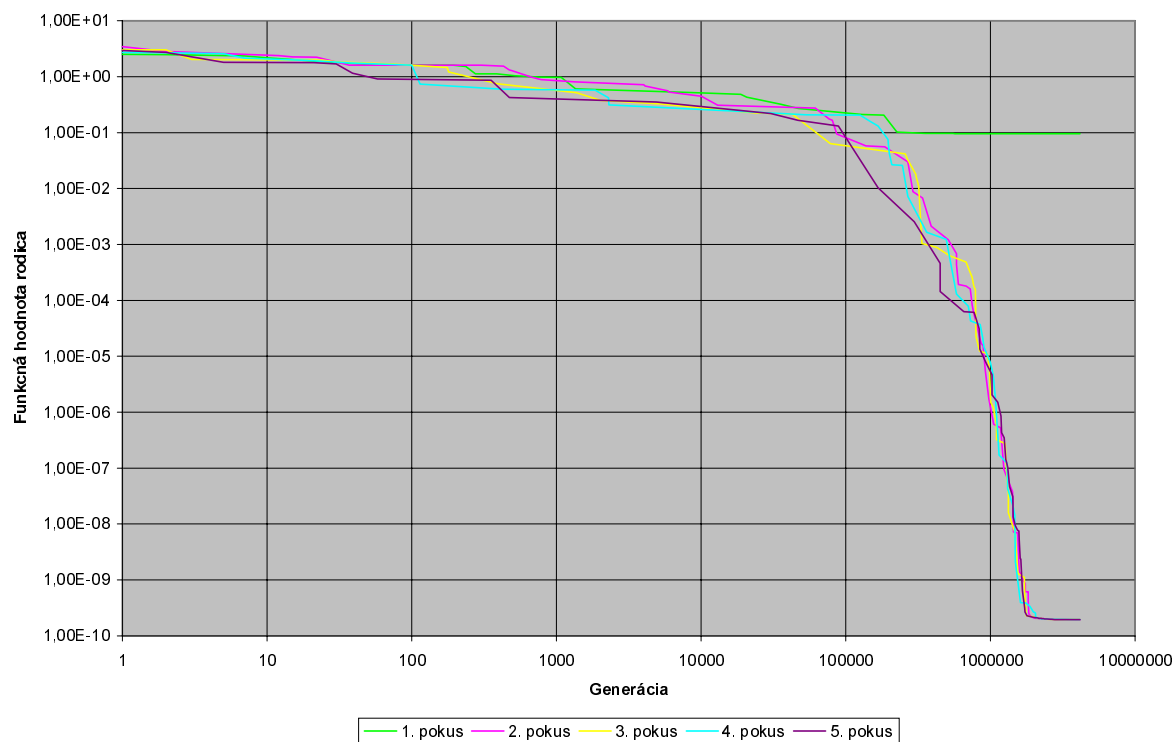
Graf 3..Zmeny funkčných hodnôt rodiča pre 1-rozmernú premennú.

Zmeny funkčnej hodnoty rodiča v závislosti od generácie pre 2D vektory



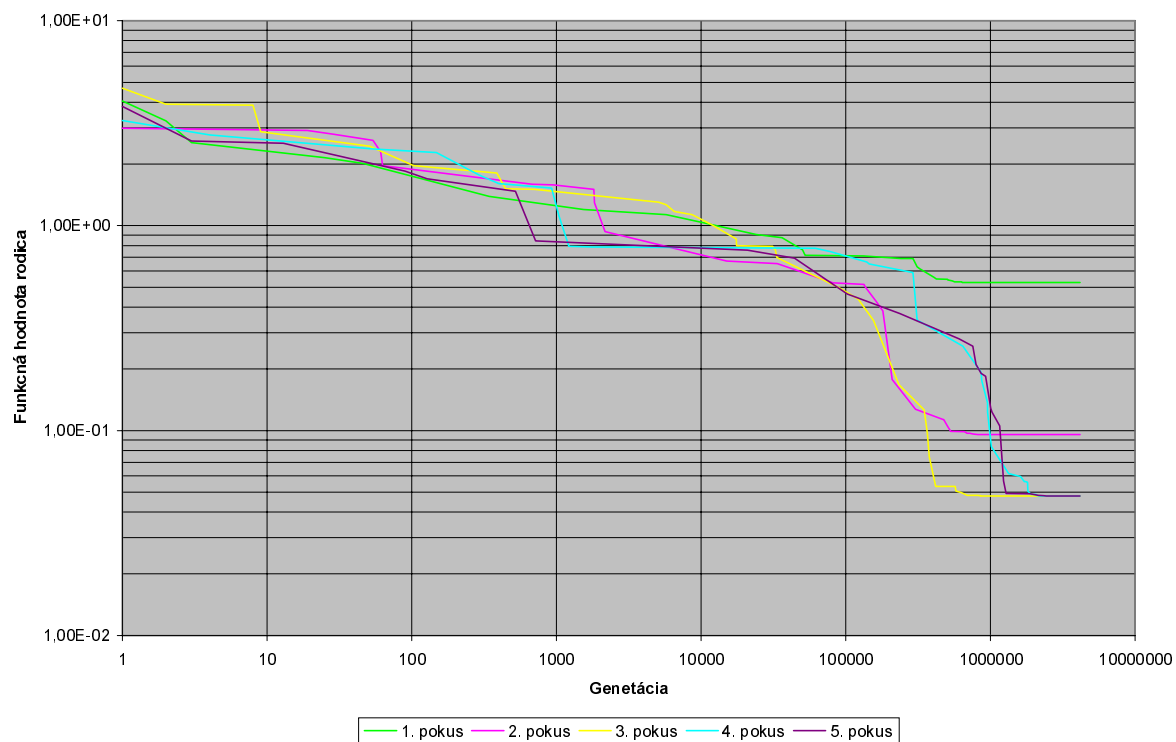
Graf 4. Zmeny funkčných hodnôt rodiča pre 2-rozmernú premennú.

Zmeny funkcej hodnoty rodica v závislosti od generácie pre 3D vektory



Graf 5. Zmeny funkčných hodnôt rodiča pre 3-rozmernú premennú.

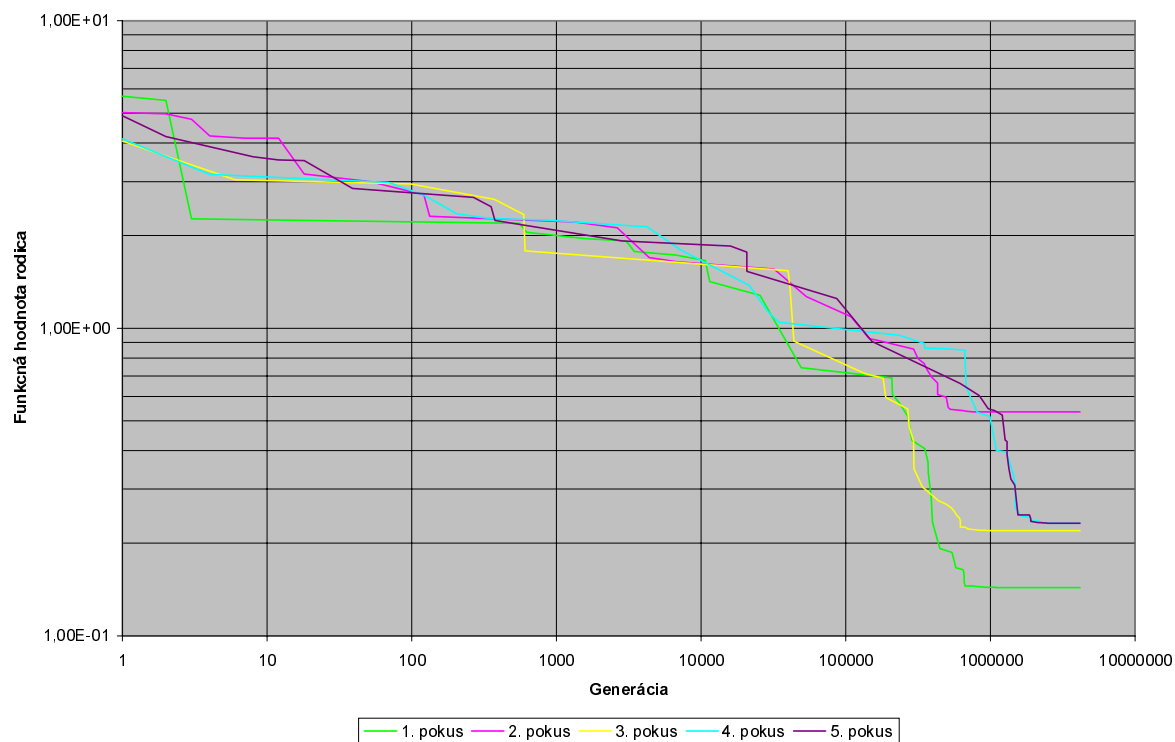
Zmeny funkcej hodnoty rodica v závislosti od generácie pre 4D vektory



Graf 6. Zmeny funkčných hodnôt rodiča pre 4-rozmernú premennú.

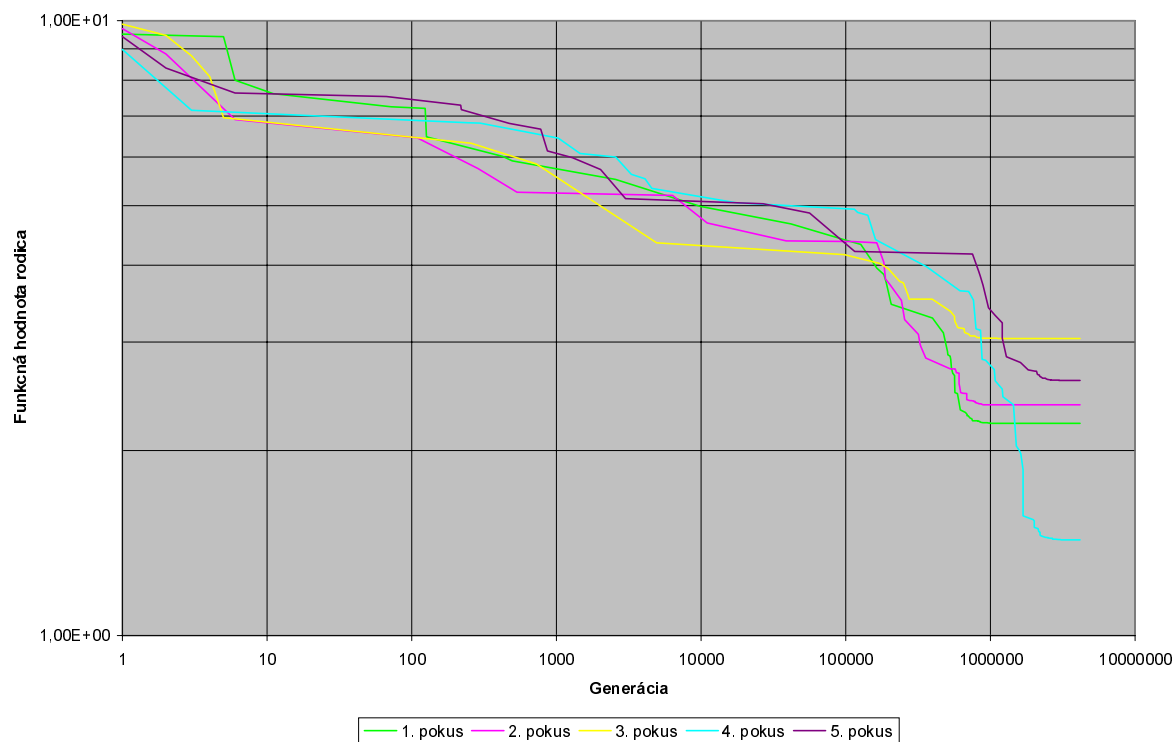


Zmeny funkcej hodnoty rodiča v závislosti od generácie pre 5D vektory



Graf 7..Zmeny funkčných hodnôt rodiča pre 5-rozmernú premennú.

Zmeny funkcej hodnoty rodiča v závislosti od generácie pre 10D vektory



Graf 8. Zmeny funkčných hodnôt rodiča pre 10-rozmernú premennú.

# Tabuľky

<b>1. pokus</b>	
X_opt=(-7, 8530243236E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	9,9746322835E-01
3	6,5952184208E-01
5	5,1983163990E-01
18	3,6426204878E-01
64	5,6206925759E-02
274	6,8513481619E-03
348	4,6716813813E-04
1272	1,7575696620E-04
3769	5,0013558393E-05
5124	1,1509927390E-05
17497	7,4086919994E-06
23985	1,3501494323E-06
58592	1,3089202184E-06
65626	2,8591421142E-07
99634	8,5052306531E-08
132328	8,5051397036E-08
191156	5,8179466578E-08
233533	2,2009771783E-10
312966	2,1827872843E-10
578123	1,1867848832E-10
862931	6,6393113229E-11
1203631	6,5483618528E-11
1704785	6,4574123826E-11
1944596	6,3664629124E-11
2586866	6,2755134422E-11
4161410	6,2755134422E-11

<b>2. pokus</b>	
X_opt=(-7, 8530243236E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	3,7138166330E-01
19	3,5099208409E-01
29	2,8061030982E-01
31	1,4565563737E-01
44	5,0804701224E-02

<b>3. pokus</b>	
X_opt=(-7, 8530243236E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	8,9592404284E-01
2	6,3345218426E-01
14	5,5686030943E-01
23	5,2259569540E-01
36	4,3966641115E-01
38	2,9142678202E-01
41	1,5228719694E-01
75	1,3741210475E-01
80	1,2394881951E-01
125	1,2292430242E-01
159	4,7695126586E-02
173	2,7098972579E-02
464	5,8089118065E-03

<b>4. pokus</b>	
X_opt=(-7, 8530243236E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	9,8463587147E-01
3	5,8084345801E-01
4	3,7306258094E-01
14	1,1271992686E-01
142	1,1133389599E-01
206	9,3040492322E-02
323	6,298209886E-02
413	2,2852140555E-03
906	8,8859924199E-04
1054	5,2655875970E-04
3454	3,8650888564E-04
3807	1,1321434613E-04
10997	2,8940803531E-06
48283	9,2215668701E-07
78770	1,3741919247E-07
103345	1,2123564375E-09
751941	3,8926373236E-10
783135	1,1550582713E-10
891689	6,9121597335E-11
1059550	6,7302607931E-11

Tabuľka 1. Zmeny funkčných hodnôt rodiča pre 1-rozmernú premennú.

<b>1. pokus</b>	
X_opt=(-7, 8530238551E-01, -7, 8530241625E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	1,4890073122E+00
7	1,3234791223E+00
9	1,1134068702E+00
14	1,0640735047E+00
16	9,0785539452E-01
24	7,9801626549E-01
57	4,8784831915E-01
164	3,1991436275E-01
384	2,5302431482E-01
495	2,0564959645E-01
1151	1,6942210396E-01
1365	7,6879300116E-02
11537	7,0176500262E-02
21619	2,4329989773E-02
23764	8,0766627061E-03
72907	3,7602795501E-03
221585	1,0557779099E-03
244404	5,4406311938E-04
358353	4,9484192732E-04
365770	3,4982072426E-04
387251	3,9998173634E-05
482545	2,3310710614E-05
550228	0,116819946E-06
724960	9,2401023701E-08
998083	6,616279544E-08
1099381	3,4616277844E-08
1136185	2,8499016480E-08
1173842	7,7042005968E-08
1209479	1,8539140001E-08
1215528	1,072982240E-09
1396694	1,331500244E-09
1450918	1,2287273421E-09
1461218	3,7562111833E-10
1478357	3,119568271E-10
1558443	2,0372681320E-10
1675555	1,5916157281E-10
1686224	1,4733814169E-10
1767588	1,3460521586E-10
2012366	1,3369572116E-10
2164811	3,278622646E-10
2257735	1,3187673176E-10
2316180	1,3096723706E-10
2466814	1,3005774235E-10
2700046	1,2914824765E-10
2855322	1,2823875295E-10
4161410	1,2823875295E-10

<b>2. pokus</b>	
X_opt=(-7, 8530238576E-01, -7, 8530240533E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	2,8380331177E+00
2	2,237832667E+00
6	1,1337134839E+00
8	1,8468963965E+00
12	1,3084136546E+00
50	1,0241269013E+00
193	9,9764314378E-01

<b>3. pokus</b>	
X_opt=(-7, 8530240130E-01, -7, 8530232142E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	7,2656102904E-01
151	6,7834451652E-01
266	6,0805397980E-01
362	5,2145466738E-01
792	4,9685409345E-01
802	4,9416689947E-01
978	4,6017310278E-01
1046	2,6567216102E-01
1237	1,9887144571E-01
2947	1,9189765252E-01
3236	6,8006799142E-02
11941	2,3868388455E-02
36711	5,7101651719E-03
144413	4,5383966890E-03
165606	4,2009923036E-03
178346	3,9938110340E-03
188076	3,7823428548E-03
190620	2,2827655730E-03
209609	1,9242405187E-03
236633	3,6413868656E-03

<b>4. pokus</b>	
X_opt=(-7, 8530235319E-01, -7, 8530238621E-01)	
<b>Gener.</b>	<b>Funk.hodnota</b>
1	1,9496220112E+00
3	1,2699443351E+00
5	1,0174988458E+00
47	7,682073221E-01
125	3,6917706546E-01
407	2,6766997706E-01
735	5,0692482146E-03
130516	2,4237189755E-03
221064	1,1725665436E-03
286833	9,9335369214E-04
297870	8,6398085386E-04
320995	4,0073942637E-04
393470	3,5062366260E-04
441410	1,6431166023E-04
461732	3,9024231228E-05
578399	2,1247339646E-05
959511	2,338372536E-05
693043	1,6414960555E-06
811545	8,6691852630E-07
854208	5,9827016230E-07
860929	2,4726523407E-07
927373	5,1885763241E-08
1060883	2,3804204830E-08
1192677	1,8850187189E-08
1184480	1,1245901987E-08
1261731	3,9153746911E-09
1286619	3,3669493860E-09
1351110	6,7848304752E-10
1485385	5,7843863033E-10

Tabuľka 2. Zmeny funkčných hodnôt rodiča pre 2-rozmernú premennú.

<b>1. pokus</b>	
X_opt=(2, 3559071105E+00, -7, 8530243236E-01, 1, 5995368793E+00)	

<b>2. pokus</b>	
X_opt=(2, 3559072260E+00, -7, 8530243236E-01, 1, 5995368793E+00)	

<b>3. pokus</b>	
X_opt=(2, 4857015384E+00, 7, 2430178490E+00, 10, 21824022341E+00)	

<b>4. pokus</b>	
X_opt=(1, 9534399741E+00, 56, 1, 6267840709E+00, 194, 1, 5995368793E+00)	







