Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Generovanie realizácií rovnomerného rozdelenia pravdepodobnosti na mnohorozmerných polyédroch

Bakalárska práca

2018
Slavomír Hanzely

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Generovanie realizácií rovnomerného rozdelenia pravdepodobnosti na mnohorozmerných polyédroch

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Školiteľ:	doc. Mgr. Radoslav Harman, PhD.

Bratislava, 2018
Slavomír Hanzely

55658269

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Slavomír Hanzely
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Generovanie realizácií rovnomerného rozdelenia pravdepodobnosti na mnohorozmerných polyédroch
Random sampling from the uniform distribution on multidimensional polyhedra

Anotácia: V Monte-Carlo metódach výpočtu pravdepodobností a v znáhodnených optimalizačných metódach je často potrebné generovať realizácie z rovnomerného rozdelenia na mnohorozmerných polyédroch. Tieto polyédre môžu byť zadané buď systémom konečného počtu lineárnych nerovníc (takzvaná H-reprezentácia), alebo ako konvexný obal konečnej množiny bodov (takzvaná V-reprezentácia). V prípade oboch typov reprezentácií je rovnomerné generovanie vo vnútri všeobecného polyédra netriviálna úloha, kombinujúca techniky a poznatky z matematiky, štatistiky a informatiky.

Cieľ: Cieľom bakalárskej práce je: Po prvé vypracovať prehľad existujúcich prístupov generovania realizácií z rovnomerného rozdelenia na polyédroch (priame generovanie pre špeciálne polyédre, zamietacie algoritmy, MCMC algoritmy a iné); po druhé vypracovať a programovo implementovať vlastnú metódu založenú na elipsoide najmenšieho objemu obsahujúceho zadaný polyéder.

Vedúci: doc. Mgr. Radoslav Harman, PhD.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Daniel Ševčovič, DrSc.

Dátum zadania: 14.10.2018

Dátum schválenia: 24.10.2018

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

....
študent vedúci práce

Poďakovanie:

Abstrakt

Kľúčové slová:

Abstract

Keywords:

Obsah

Úvod

V rámci tejto práce sa budeme zaoberať metódami na rovnomerné generovanie bodov vo veľarozmernom polyédre (konvexnom mnohostene). Našou úlohou vytvoriť generátor, ktorý bude čo najrýchlejšie generovať body vnútri polyédru rovnomerne náhodne, tj. pravdepodobnosť, že dostaneme bod vnútri ľubovoľnej oblasti polyédra je iba lineárne závislá od objemu danej časti.

Ako základ náhody bude používať náš generátor bodu v polyédre používať rovnomerný generátor čísel $[0, 1]$. Pomocou generátoru na $[0, 1]$ možno triviálne generovať bod na $[0, k]$ (prenásobením konštantou k), tiež možno generovať bod na $[a, b]$ (vygenerovaním bodu na $[0, -a + b]$ a pripočítaním a), alebo bod na $[0, 1]^n$ (postupným vygenerovaním súradníc). **TODO doplnit, co budeme pouzivat**

Vo všeobecnosti možno polyéder reprezentovať viacerými spôsobmi, napríklad ako konvexný obal bodov alebo ako sústavu lineárnych nerovníc. Obidve spomenuté reprezentácie možno previesť na tú druhú, avšak prevod medzi nimi nie je lacný **TODO doplnit cenu**. Pre účely tejto práce budeme pracovať s polyédrom reprezentovaným sústavou lineárnych nerovníc, riešení systému $Ax \leq b$ ($x \in X$ ak $Ax \leq b$).

TODO prechod medzi metodami

TODO ujednotit notaciu v texte a algoritmoch

Kapitola 1

Metropolis-Hastings metódy

V tejto kapitole sa budeme zaoberať Metropolis-Hastings algoritmom na generovanie bodov z ľubovoľnej distribúcie. Najprv sa pozrieme na všeobecný Metropolis-Hastings algoritmus, následne sa pozrieme na jeho konkrétne realizácie.

1.1 Všeobecný Metropolis-Hastings algoritmus

Majme cieľovú hustotu Q z ktorej chceme generovať, v prípade generovania vnútri polyédru je rovnomerná na polyédri a nulová mimo neho.

Metropolis-Hastings algoritmus je vždy v stave $x^{(i)}$ reprezentovanom bodom v priestore, stav určuje hustotu $Q(x^{(i)})$ závislú na $x^{(i)}$. Algoritmus vygeneruje ďalší potenciálny stav y podľa hustoty $Q(x^{(i)})$. Ďalší stav algoritmu $x^{(i+1)}$ bude y s pravdepodobnosťou $\alpha(x^{(i)}, y)$, inak to bude $x^{(i)}$.

Algorithm 1 Všeobecný Metropolis-Hastings algoritmus [2]

```
inicializuj x^(0)
for i = 0, 1, ..., N do
    Vygeneruj bod y z Q(x^(i))
    Vygeneruj u z U(0, 1).
    if u ≤ α(x^(i), y) then
        Nastav x^(i+1) = y
    else
        Nastav x^(i+1) = x^(i)
Vráť x^(1), x^(2), ..., x^(N).
```

Môžeme si všimnúť, že v Metropolis-Hastings algoritme je bod $x^{(i)}$ závislý od predchádzajúceho bodu $x^{(i-1)}$. Podľa [2] je možné dokázať, že napriek závislosti po sebe idúcich bodov pre dostatočne veľké N budú body $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$ z hustoty Q.

TODO Ake velke N?

TODO Vseobecnejsie Monte Carlo Markov Chain metody - Metropolis algoritmus ako specialny pripad

TODO volba parametru α

TODO prakticke vyuzitie

TODO exaktny algoritmus na generovanie v polyedri

1.2 Hit-and-Run generátor

Ako jedna z možností na realizáciu Metropolis-Hastings algoritmu prichádza do úvahy Hit-and-Run generátor. Algoritmus je analogický s algoritmom Metropolis-Hasting, pričom hustota $Q(x^{(i)})$ je určená priamkou s náhodným smerom cez bod $x^{(i)}$.

Algorithm 2 Hit-and-Run generátor [1]

Inicializuj $x^{(0)}$
for $n = 0, \dots, N-1$ **do**
    Vygeneruj smer d_n z distribúcie D na povrchu sféry
    Nájdi množinu $S_n(d_n, x^{(n)}) = \{\lambda \in \mathbb{R}; x^{(n)} + \lambda d_n \in S\}$
    Zvoľ $y = x^{(n)} + \lambda_n d_n$
    Vygeneruj u z $U(0,1)$.
    if $u \leq \alpha_n(y|x^{(n)})$ **then**
        Nastav $x^{(i+1)} = y$
    else
        Nastav $x^{(i+1)} = x^{(i)}$
Vráť $x^{(1)}, x^{(2)}, \dots, x^{(N)}$.

TODO Vplyv volby distribucie D

TODO Dokaz konvergentnosti, rychlost [1]

TODO vyuzitie pri polyedroch

1.3 Gibbsov generátor

V tejto sekcii sa budeme zaoberať Gibbsovým generátorom, metódou generovania z triedy MCMC vhodnou na generovanie vo viacrozmernom priestore. Na Gibbsov generátor sa možno dívať ako na špeciálny prípad Metropolis-Hastings algoritmu.

Našou úlohou je generovať z K-rozmernej distribúcie Q, pričom z Q nevieme generovať priamo. Predpokladajme, že nevieme použiť priamo Metropolis-Hastings algoritmus, lebo $Q(x^{(i)}) = Q(x_1^{(i)}, x_2^{(i)}, \dots, x_K^{(i)})$ je príliš zložitá na generovanie. Taktiež predpokladajme, že ak $Q(x^{(i)})$ obmedzíme na jeden rozmer, tak v ňom vieme generovať rýchlo, tj. možno generovať rýchlo z $Q(x_j^{(i)}|x_1^{(i+1)}, x_2^{(i+1)} \dots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, x_{j+2}^{(i)} \dots, x_K^{(i)})$.

Gibbsov generátor bude fungovať nasledovne:

Algorithm 3 Gibbsov generátor [5]

inicializu $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_K^{(0)})$
for $i = 1, \dots, N$ **do**
  for $j = 0, 1, \dots, K$ **do**
    $x_j^{(i)} \sim Q(x_j | x_1^{(i+1)}, x_2^{(i+1)} \dots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, x_{j+2}^{(i)} \dots, x_K^{(i)})$
  $x^{(i+1)} = (x_1^{(i+1)}, x_2^{(i+1)} \dots, x_K^{(i+1)})$
Vráť $x^{(1)}, x^{(2)}, \dots, x^{(N)}$

Gibbsov generátor ako špeciálny prípad Metropolis-Hastings algoritmu má podobné vlastnosti ako Metropolis-Hastings algoritmus.

TODO specifickost oproti všeobecnému Metropolis-Hastingsu: menej parametrov
TODO prakticke vyuzitie
TODO vyuzitie pri polyedroch

1.4 Slice sampling

Majme d rozmernú hustotu $\pi : \mathbb{R}^d \to [0, \infty)$, pričom $\pi(x) = \Pi_{i=0}^{K} f_i(x)$ pre nejaké $f_i : \mathbb{R}^d \to [0, \infty)$.

Slice sampler bude pracovať nasledovne. Začne s bodom $x^{(0)}$, na vygenerovanie bodu $x^{(n)}$ z $x^{(n-1)}$ najprv vygeneruje nezávislé náhodné premenné $y_{n,i}$ v závislosti od $f_i(x^{(n-1)}$. Následne pomocou $y_{n,i}$ vygeneruje bod $x^{(n)}$.

Algorithm 4 Slice sampling algoritmus [6]

inicializuj $x^{(0)}$
for $n = 1, \dots, N$ **do**
  vygeneruj nezávislé náhodné premenné $y_{n,1}, y_{n,2}, \dots, y_{n,K}$,
   kde $y_{n,i} \sim U(0, f_i(x^{(n-1)})$
  vygeneruj $x^{(n)}$ z distribúcie $f_0(.) \mathbf{1}_{L(y_n)}$,
   kde $L(y) = \{z \in \mathbb{R}^d; f_i(z) \geq y_{n,i}, i = 1, 2, \dots, K\}$
Vráť $x^{(1)}, x^{(2)}, \dots, x^{(N)}$.

Daný algoritmus je závislý jedine od faktorizácie distribúcie $\pi(x)$ na $\Pi_{i=0}^{K} f_i(x)$. **TODO rozvinut**

Daný algoritmus je tiež špeciálnym prípadom Metropolis-Hastings algoritmu, možno ho analyzovať rovnakým spôsobom.

TODO porovnanie so vseobecnym MH a Gibbsom

TODO vyuzitie pri polyedroch

Kapitola 2

Zamietacie metódy

Zamietacie metódy nám poskytujú jeden zo spôsobov rovnomerného generovania bodov na určitej množine. Myšlienka za nimi je nasledovná: Označme si X množinu, na ktorej chceme rovnomerne náhodne generovať prvky. Predpokladajme, že nevieme priamo rovnomerne generovať body na X, no vieme rovnomerne generovať na množine S, $X \subset S$.

Náš generátor G_z bude pracovať nasledovne:

Algorithm 5 Zamietacia metóda

1: **for** $n = 0, \ldots, N-1$ **do**
2: **repeat**Vygeneruj bod $x^{(n)} \in S$ rovnomerne náhodne
3: **until** $x^{(n)} \in X$
4: Vráť $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$

Generátor G_z vygeneruje bod $x \in S$, ak je ten bod aj z X, tak ho vráti ako výstup, inak vygeneruje nový bod $x \in S$.

Generátor G_z generuje na X rovnomerne náhodne. Očakávaná rýchlosť generovania závisí od toho, koľkokrát G_z vygeneruje bod mimo X. Z rovnomernosti G_z je tá pravdepodobnosť rovná $\frac{|S-X|}{|S|} = 1 - \frac{|X|}{|S|}$. Označme si p_k pravdepodobnosť, že G_z bude generovať bod z S k-krát. Platí $p_k = (1 - \frac{|X|}{|S|})^{k-1}\frac{|X|}{|S|}$. Očakávý počet generovaní G_z je $E(G_z) = \sum_0^\infty kp_k = \frac{|X|}{|S|}\sum_0^\infty k(1 - \frac{|X|}{|S|})^{k-1}$.

Táto metóda generovania je vhodná, ak je $\frac{|X|}{|S|}$ dosť veľké, tj. ak je obal S polyédru X dostatočne malý. Ak je $\frac{|X|}{|S|} \sim 0$, tak je táto metóda nepoužiteľná.

TODO citovat

2.1 Použitie na generovanie bodu vnútri polyédru

Zamyslime sa nad tým, ako by sme vedeli použiť túto metódu na generovanie bodu vnútri polyédru. Ako množinu možných S, $X \subset S$ môžeme použiť najmenší kváder so stranami rovnobežnými s osami. Vypočítať súradnece kvádra je ľahké, stačí nám to spraviť raz pred (začatím generovania) pomocou lineárneho programovanie.

Žiaľ pre takúto množinu S môže byť podiel $\frac{|X|}{|S|}$ byť ľubovoľne malý. Ak bude X kváder s obsahom k pozdĺž diagonály kocky $[0,1]^n$, dotýkajúci sa každej steny kocky $[0,1]^n$, tak najmenšia množina S (kváder so stranami rovnobežnými s osami) je kocka $[0,1]^n$, ktorá má obsah 1. Platí $\frac{|X|}{|S|} = k$. Keďže vieme nájsť kváder taký, že k je ľubovoľne malé, tak očakávaná dĺžka generovania touto metódou (pre danú množinu S) je ľubovovoľne veľká.

V rámci tejto práce budeme skúmať použitie problému obalenia polyédru elipsoidom s najmenším obsahom (Minimum Volume Enclosing Elipsoid, ďalej MVEE). Ako množinu S zvolíme nájdený elipsoid. Generovať body v ňom rovnomerne náhodne vieme ľahko pomocou REX algoritmu, s ktorým sa oboznámime neskôr.

Taktiež sa pokúsime pomocou hlavných osí MVEE elipsoidu obaliť polyéder kvádrom s najmenším obsahom (jeho strany nemusia byť rovnobežné s osami sústavi). Ako množinu S môžeme zvoliť daný kváder.

Kapitola 3

Randomized exchange algorithm

V tejto kapitole si predstavíme Randomized exchange algoritmus [3] (ďalej REX) ako metódu na riešenie optimal design problému **TODO preložiť**. Taktiež, vďaka ekvivalencii optimal design problému a minimum volume enclosing elipsoidu (MVEE) [3], sa dá využiť aj na riešenie MVEE problému. Vzhľadom na dôležitosť algoritmu pre túto prácu **TODO ešte si to premyslieť** a kvôli lepšiemu pochopopeniu algoritmu sa najprv pozrime na metódy riešenia optimal design problému.

3.1 Metódy na riešenie optimal design point problému

Najprv si predstavíme metódu SAM ako všeobecnú iteratívnu metódu na riešenie optimal design problému a VEM algoritmus ako jej konkrétnu realizáciu **TODO čo je na VEM špecifické**. Následne sa pozrieme na REX algoritmus ako na špeciálny prípad SAM, ktorý kombinuje VEM metódu s pažravým prístupom.

Algorithm 6 SAM metóda [3]

Zvoľ regulárny m-point design $w^{(0)}$
while $w^{(k)}$ nespĺňa podmienky zastavenia **do**
    Zvoľ podmnožinu bodov $S_k \subset X$
    Nájdi aktívny podpriestor Ξ ako $\Xi_k \leftarrow \{w \in \Xi : w_x = w_x^k forx \notin S_k\}$
    Vypočítaj w^{k+1} ako riešenie $\max_{w \in \Xi_k} \Phi(M(w))$ spĺňajúce $\Phi(M(w^{k+1})) \geq \Phi(M(w^k))$
    Set $k \leftarrow k + 1$
Vráť w

TODO obkec okolo LBE

Algorithm 7 VEM algoritmus [3]

Zvoľ regulárny m-point design w
while $eff.act(w) < effandtime.act < time.max$ **do**
$k \leftarrow \arg\min\{d_u(w) : u \in supp(w)\}$
$l \leftarrow \arg\max\{d_v(w) : v \in X\}$
$\alpha^* \leftarrow \arg\max\{\Phi_D(M(w + \alpha e_l - \alpha e_k)) : \alpha \in [-w_l, w_k]\}$
$w_k \leftarrow w_k - \alpha^*$
$w_l \leftarrow w_l + \alpha^*$
Vráť w

3.2 Radomized Exchange Algoritmus

V tejto sekcii popíšeme randomized exchange algoritmus (REX) predstavený v [3], dá sa na neho pozerať ako na špeciálny prípad SAM algoritmu [3].

Hlavná myšlienka REX algoritmu je počnúc inicializovaným regulárnym bodom w a $g(w)$ **TODO co je g(w)** opakovane vyberať niekoľko bodov (ich počet sa bude líšiť v rámci iterácii) a náhodne vykonať optimálnu výmenu váh medzi vybranými bodmi. Voľba bodov závysí na $g(w)$.

REX algoritmus kombinuje kroky VEM algoritmu a pažravých algoritmov.

- **Krok LBE.** Pri danom bode w, vypočítaj $g(w)$ a urob LBE krok daný nasledovne:
$$\alpha^* \leftarrow \arg\max\{\Phi_D(M(w + \alpha e_l - \alpha e_k)) : \alpha \in [-w_l, w_k]\},$$
kde $k \in \arg\min\{d_u(w) : u \in supp(w)\}, l \in \arg\max\{d_v(w) : v \in X\}$. Optimálny krok $\alpha^*_{k,l}(w)$ nazvime *nulujúci*, ak je rovný buď $-w_l$ alebo w_k. To zodpovedá prípadu, keď sme sa optimálnym krokom pohli do niektorého z bodu w_l alebo w_k **TODO čomu to zodpovedá**

- **Výber aktívneho podpriestoru.** Podpriestor $S \subset X$, v ktorom sa pohneme bude zvolený ako zjednotenie dvoch množín. Jednou vybranou pažravým procesom (S_{greedy}) a druhou ako nosnou množinou bodu w ($S_{support}$).
  - **Pažravá množina.** Nech $L = \min(\gamma m, n)$ je počet bodov, ktoré vyberieme. Potom zvoľ S_{greedy} ako
  $$S_{greedy} = \{l_1^*, \ldots, l_L^*\} \subset X,$$
  kde l_i^* je najväčšia zložka vektoru $g(w)$.
  - **Nosná množina.** Nastav
  $$S_{support}(w) = supp(w).$$

Označme K veľkosť nosnej množiny $K = |supp(w)|$.

- **Aktívny podpriestor.** Aktívny podpriestor S je definovaný ako

$$S = S_{greedy} \cup S_{support}.$$

Váhy bodov mimo aktívneho podpriestoru nebudú upravované v tejto iterácii.

- **Krok v aktívnom podpriestore.** Teraz vykonáme krok v ktorom aktualizujeme hodnoty w_v pre $v \in S$. Body w_v pre $v \notin S$ ostanú nezmenené.

  - **Tvorba párov.** Nech $(k_1, \dots, k_K)$ je uniformne náhodná premutácia $S_{support}$ a nech $(l_1, \dots, l_L)$ je uniformne náhodná pormutácia S_{greedy}. Potom postupnosť aktívnych bodov je

$$(k_1, l_1), (k_2, l_1), \dots, (k_1, l_L), (k_2, l_L), \dots, (k_K, l_L)$$

  - **Aktualizácia.** Vykonaj postupne všetky Φ-optimálne kroky medzi bodmi z postupnosti **TODO odkaz sa na postupnost nad** bodov z $K \times L$ s prisluchajúcimi aktualizáciami w a $M(w)$.

Algorithm 8 REX algoritmus [3]

Zvoľ regulárny m-point design w
while $w^{(k)}$ nespĺňa podmienky zastavenia **do**
    Urob LBE krok vo w
    Nech k je vektor zodpovedajúci náhodnej permutácii prvkov $supp(w)$
    Nech l je vektor zodpovedajúci náhodnej permutácii $L = \min(\gamma m, n)$ indexov prvkov $g(w)$
    for $l = 1 \dots L$ **do**
        for $l = 1 \dots K$ **do**
            $\alpha^* \leftarrow \arg\max\{\Phi_D(M(w + \alpha e_l - \alpha e_k)) : \alpha \in [-w_l, w_k]\}$
            if LBE krok bol nulujúci alebo $\alpha^* = -w_l$ alebo $\alpha^* = w_k$ **then**
                $w_k \leftarrow w_k - \alpha^*$
                $w_l \leftarrow w_l + \alpha^*$
Vráť w

Literatúra

[1] Ming-Hui Chen and Bruce W. Schmeiser. General Hit-and-Run Monte Carlo sampling for evaluating multidimensional integrals. *Operations Research Letters*, 19(4):161–169, October 1996.

[2] Siddhartha Chib and Edward Greenberg. Understanding the Metropolis-Hastings Algorithm. *The American Statistician*, 49(4):327–335, November 1995.

[3] Radoslav Harman, Lenka Filová, and Peter Richtárik. A Randomized Exchange Algorithm for Computing Optimal Approximate Designs of Experiments. *arXiv:1801.05661 [stat]*, January 2018. arXiv: 1801.05661.

[4] Radoslav Harman and Vladimír Lacko. On decompositional algorithms for uniform sampling from n-spheres and n-balls. *Journal of Multivariate Analysis*, 101(10):2297–2304, November 2010.

[5] D. J. C. Mackay. Introduction to Monte Carlo Methods. In Michael I. Jordan, editor, *Learning in Graphical Models*, NATO ASI Series, pages 175–204. Springer Netherlands, Dordrecht, 1998.

[6] Gareth O. Roberts and Jeffrey S. Rosenthal. Convergence of Slice Sampler Markov Chains. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):643–660, January 1999.