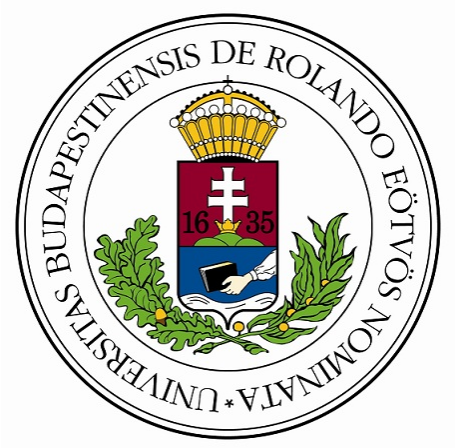
Eötvös Loránd Tudományegyetem
Természettudományi Kar

MCMC módszerek

Szakdolgozat

Sztahó Roland István
Matematika B.Sc., elemző szakirány

Témavezető: **Pröhle Tamás**, tanársegéd
Valószínűségelméleti és Statisztika Tanszék



Budapest
2012

Tartalomjegyzék

1. Bevezetés **3**

1.1. Integrálás Monte Carlo módszerrel 3

1.2. Markov láncok 4

2. MCMC szimuláció **8**

3. Metropolis-Hastings módszer **10**

3.1. A Metropolis-Hastings algoritmus 10

3.2. Megvalósítás 12

4. Gibbs Sampling **14**

4.1. A Gibbs mintavételező eljárás 14

4.2. Konvergencia 15

4.3. Két- és többváltozós eset 17

4.4. Megvalósítás 19

5. Slice sampling **20**

5.1. Egyváltozós eset 21

5.2. Többváltozós eset 26

5.3. Egyenletes eloszlás egy kétdimenziós tartományon 27

6. Perfect sampling **29**

6.1. Az alapötlet 29

6.2. A Coupling From The Past (CFTP) algoritmus 30

6.3. Fontainebleau lehulló levelei 31

6.4. Sandwiching 32

7. Összefoglalás **38**

8. Függelék **39**

8.1. R program a sandwiching módszer bemutatásához 39

9. Köszönetnyilvánítás **45**

1. fejezet

Bevezetés

MCMC (Markov chain Monte Carlo) módszerek alatt algoritmusok egy családját értjük, melyek a második világháború alatt, Los Alamosban láttak napvilágot nem sokkal az eredeti Monte Carlo módszerek születése után. Nicholas Metropolis több társszerzővel 1953-ban publikálta azt a módszert, melyet utána Metropolis algoritmusnak nevezünk. Ezt később W. Keith Hastings 1970-ben általánosította amelyet azóta is mint az egyik leismertebb eljárást, Metropolis-Hastings algoritmusnak nevezzük. A Geman testvérek 1984-ben publikált cikkükben bemutatták a Metropolis-Hastings algoritmus egy speciális esetét, amely Gibbs mintavételező eljárás néven híresült el. A számítógépek fejlődésével a módszereket egyre szélesebb körben alkalmazták, de a statisztikában csak 1990 után váltak igazán népszerűvé. Az algoritmus-családnak számos alkalmazási területe van, például: numerikus integrálás, Bayes-i statisztika, statisztikus fizika, bioinformatika.

1.1. Integrálás Monte Carlo módszerrel

Az eredeti Monte Carlo módszert fizikusok fejlesztették ki komplex integrálok kiszámítására véletlen számok segítségével. Tegyük fel, hogy szeretnénk kiszámítani a következő integrált:

$$\int_a^b f(x)dx.$$

Tegyük fel, hogy $f(x)$-et fel tudjuk írni a következő alakban: $f(x) = g(x) \cdot h(x)$, ahol $h(x)$ egy (a, b) intervallumon értelmezett sűrűségfüggvény.

Ekkor

$$\int_a^b f(x)dx = \int_a^b g(x) \cdot h(x)dx = E_{h(x)}[g(x)].$$

Tehát az integrált kifejezhetjük mint $h(x)$ várható értékét, ha a sűrűségfüggvény $g(x)$. Ezáltal ha van egy $x_1, \ldots, x_n$ mintánk a $g(x)$ sűrűségfüggvényből, kellően nagy n esetén

$$\int_a^b f(x)dx = E_{h(x)}[g(x)] \simeq \frac{1}{n} \sum_{i=1}^{n} f(x_i).$$

Ezt a közelítést nevezzük Monte Carlo integrálásnak.

1.2. Markov láncok

Mielőtt rátérnénk különböző MCMC módszerek tárgyalására, a Markov láncokkal kapcsolatos néhány alapfogalmat elengedhetetlen tisztázni. A Markov láncok szemléletes értelmezéséhez vegyünk egy véges sok csúcsból álló, élsúlyozott, irányított gráfot, melyben hurokélek is megengedettek. Legyen minden él súlya nemnegatív, valamint minden csúcs esetén a kimenő élek súlyainak összege legyen 1. Ha ezen a gráfon bolyongunk az élekre írt valószínűségek szerint, valamint egységnyi időközönként lépünk, akkor diszkrét paraméterű Markov láncot kapunk.

1.1. Definíció. *Legyen adott az $\{X_n\}_{n\in\mathbb{N}}$ folyamat, ahol $X_n : \Omega \to I$ valószínűségi változók az (Ω, F, P) valószínűségi mezőn, I pedig megszámlálható halmaz. A folyamatot diszkrét paraméterű homogén Markov láncnak nevezzük, ha a folyamat Markov tulajdonságú, azaz*

$$F_n = \sigma(X_0, X_1, \ldots, X_n)$$

jelöléssel minden $B \subseteq I$ és minden $m \geq n$ esetén

$$P(X_m \in B \mid F_n) = P(X_m \in B \mid X_n).$$

A Markov folyamat stacionárius (vagy homogén) átmenetvalószínűségű, azaz minden $i, j \in I$ esetén minden olyan n-re, melyre $P(X_n = i) > 0$,

$$P(X_{n+1} = j \mid X_n = i) = p_{ij},$$

n-től függetlenül.

Az I halmazt, azaz a valószínűségi változók lehetséges értékeit állapottérnek nevezzük.

Tehát a Markov lánc egy olyan $X_0, X_1, \ldots, X_n$ valószínűségi változó sorozat, mely Markov tulajdonságú, azaz a $(n+1)$-edik pillanatbeli állapot meghatározásához az előző állapotok nem szükségesek, csupán az éppen aktuális n-edik, ugyanis a korábbi állapotok ismerete nem változtatja meg egyik $i \to j$ átmenet valószínűségét sem.

A $P = (p_{ij})$ mátrixot átmenetmátrixnak nevezzük, elemei az átmenetvalószínűségek, azaz hogy az $(n+1)$-edik időpontban j-ben vagyunk, feltéve hogy az n-edikben i-ben voltunk. Az átmenetvalószínűségek összességét átmenetmagnak nevezzük. Az X_0 eloszlását kezdeti eloszlásnak hívjuk, és $p = (p_i)$-vel jelöljük. Ez a két objektum már meghatározza a folyamatot, hiszen a véges dimenziós eloszlásokat és a Markov-tulajdonságot felhasználva:

$$P(X_0 = i_0, X_1 = i_1, \ldots, X_n = i_n) =$$
$$= P(X_0 = i_0)P(x_1 = i_1 \mid X_0 = i_0) \ldots P(X_n = i_n \mid X_0 = i_0, \ldots, X_{n-1} = i_{n-1}) =$$
$$= p_{i_0} p_{i_0 i_1} \ldots p_{i_{n-1} i_n}.$$

Jelölje

$$\pi_j(t) = P(X_t = j)$$

annak a valószínűségét, hogy a lánc a t-edik időpontban j-ben van és jelölje $\pi(t)$ azt a sorvektort, melynek i-edik eleme $\pi_i(t)$. A láncot egy $\pi(0)$ kezdővektorral indítjuk, melynek gyakran minden eleme 0, egy kivételével, mely ebből adódóan 1. A t előrehaladtával ez az 1 valószínűség eloszlik $\pi(t)$ koordinátáin.

1.2. Definíció. *A P mátrix*

1. *sztochasztikus, ha $p_{ij} \geq 0$ és minden sorban az elemek összege* 1,
2. *duplán sztochasztikus, ha sztochasztikus és minden oszlopban az elemek összege* 1,
3. *szubsztochasztikus, ha $p_{ij} \geq 0$ és minden sorban az elemek összege legfeljebb* 1.

1.1. Tétel. *Tetszőleges I megszámlálható halmazon adott p eloszláshoz és $\mid I \mid \times \mid I \mid$ méretű P sztochasztikus mátrixhoz létezik I állapotterű Markov lánc, melynek kezdeti eloszlása p, átmenetmátrixa P.*

Annak a valószínűsége, hogy egy lánc a $(t+1)$-edik időpillanatban az s_i állapotban van, meghatározható a Chapman-Kolomogrov egyenlőség alapján:

$$\pi_j(t+1) = P(X_{t+1} = s_i) =$$

$$\sum_k P(X_{t+1} = s_i \mid X_t = s_k) \cdot P(X_t = s_k) = \sum_k p_{ki} \cdot \pi_k(t).$$

Ezt könnyen láthatjuk, ha iteratívan alkalmazzuk az egyenlőséget:

$$\pi(t) = \pi(t-1)P = (\pi(t-2)P)P = \ldots = \pi(0)P^t.$$

Ezek alapján definiáljuk az n-edrendű átmenetvalószínűséget:

1.1. Állítás. *Legyenek $p_{ij}^n = P(X_{n+m} = j \mid X_m = i)$ az n-edrendű átmenetvalószínűségek (feltesszük, hogy $P(X_m = i) > 0$). Ezekre teljesül a Chapman-Kolomogrov egyenlőség:*

$$p_{ij}^{n+m} = \sum_k p_{ik}^{(n)} p_{kj}^{(m)}.$$

Ez azt jelenti, hogy a p_{ij}^n mennyiségek éppen a P^n mátrix megfelelő elemei.

Szükségünk van még továbbá az állapotok osztályozására:

1.3. Definíció.

1. *Azt mondjuk, hogy az i állapotból elérhető j, $(i \to j)$, ha van olyan $n \geq 0$, hogy $p_{ij}^n \geq 0$.*

2. *Azt mondjuk, hogy i és j közlekednek (lehet hogy több lépés szükséges), ha i-ből elérhető j és fordítva. Ez ekvivalenciareláció, tehát osztályokra bontja az állapotteret (csak az átmenetmátrixtól függ). A Markov lánc irreducibilis ha csak egyetlen osztályból áll.*

Ezek a tulajdonságok osztálytulajdonságok, azaz vagy egy osztály minden eleme rendelkezik az adott tulajdonsággal, vagy egyik sem.

1.4. Definíció. *Az $\{n > 0 : p_{ii}^{(n)} > 0\}$ halmaz legnagyobb közös osztója az i periódusa, jele: $d(i)$. Ha a halmaz üres, akkor a periódust nem értelmezzük. Ha $d(i) = 1$, akkor az állapotot aperiodikusnak nevezzük.*

Szemléletesen ez azt jelenti, hogy semmilyen fix lépésszám esetén nem feltételezhetünk törvényszerű visszatérést egy adott állapotba.

1.5. Definíció. *Legyen P egy átmenetmátrix. A $(p_i)_{i\in I}$ eloszlás stacionárius, ha $p_i = \sum_{k\in I} p_k p_{ki}$ minden $i \in I$ esetén, azaz a p_i kezdeti eloszlású, P átmenetvalószínűségű X_n Markov láncra $P(X_n = i) = p_i$ minden $i \in I, n \geq 0$ esetén. Ez utóbbi esetben azt mondjuk, hogy a Markov lánc stacionárius.*

Ebből adódóan, ha π^* stacionárius eloszlás, akkor $\pi^* = \pi^* P$. Így tekinthetünk π^*-ra, mint a P mátrix baloldali $\lambda = 1$ sajátértékhez tartozó sajátvektorára. A stacionárius elolszás feltétele, hogy a Markov láncunk irreducibilis és aperiodikus legyen. Ha egy Markov lánc periodikus, akkor megvan annak a lehetősége, hogy két állapot között váltakozik és nem tart egy stacionárius eloszláshoz, illetve a váltakozás maga lesz a stacionárius eloszlás. Ha P-nek nincs (-1)-gyel egyenlő sajátértéke, akkor a lánc aperiodikus. Az egyértelműséghez szükség van továbbá a megfordíthatósági feltételre:

$$p_{ij}\pi_i^* = p_{ji}\pi_j^* \qquad \forall i, j.$$

1.6. Definíció. *A fenti egyenlőség teljesülése esetén a Markov láncot megfordíthatónak nevezzük.*

Könnyen láthatjuk, hogy ebből már következik $\pi = \pi P$, ugyanis πP j-edik eleme:

$$(\pi P)_j = \sum_i \pi_i p_{ij} = \sum_i \pi_j p_{ji} = \pi_j \sum_i p_{ji} = \pi_j \cdot 1 = \pi_j.$$

A diszkrét állapotterű Markov láncok általánosítása folytonos állapottérre azon a megfontoláson alapszik, hogy az átmenetmag, $P(x, y)$ kielégíti az

$$\int P(x, y)dy = 1$$

egyenlőséget, illetve a Chapman-Komologrov egyenlőséget értelmezzük folytonos esetben:

$$\pi_t(y) = \int \pi_{t-1}(x)P(x, y)dy.$$

Így (ha létezik) a stacionárius eloszlás, kielégíti az alábbi egyenlőséget:

$$\pi^*(y) = \int \pi^*(x)P(x, y)dy.$$

2. fejezet

MCMC szimuláció

Ahogy láttuk az előző fejezetben, a Markov láncok általánosságban a következő módon működnek: adott egy átmenetmag, mely tartalmazza a (feltételes) átmenetvalószínűségeket arra vonatkozóan, hogy egy pontból egy másikba lépünk, legyen ez $P(x, A)$, ahol $x \in \mathbb{R}^d$, A pedig σ-algebra. Az átmenetmag egy feltételes eloszlásfüggvény, mely az x állapotból egy másik állapotba lépés valószínűségét írja le A-n, ezért $P(x, \mathbb{R}^d) = 1$, illetve mivel megengedjük hogy az x állapotból x-be lépjünk, $P(x, \{x\})$ nem feltétlenül nulla. A célunk az volt, hogy meghatározzuk azokat a feltételeket melyek esetén létezik a $\pi^*(x)$ stacionárius eloszlás, melyhez az iteráció során tart az átmenetmag, amely kielégíti a következő egyenlőséget:

$$\pi(y) = \int_{\mathbb{R}^d} P(x, y)\pi(x)dx.$$

Az MCMC módszerek pont fordítva működnek: a kívánt $\pi^*(\cdot)$ eloszlás ismert, de az átmenetmag ismeretlen. Ahhoz, hogy $\pi^*(\cdot)$-ból mintákat generáljunk, olyan átmenetmagot kell gyártanunk, mely az iterációk számának növelésével konvergál a $\pi^*(\cdot)$ ismert eloszláshoz. A folyamatot egy tetszőlegesen választott vektorral kezdjük és kellően sok iteráció után a szimuláció során megfigyelt értékek eloszlása közelítőleg azonos lesz a $\pi^*(\cdot)$ ismert eloszlással. Ez nagy segítséget jelent a Monte Carlo integrálás során, ha az adott eloszlásfüggvény túlzottan összetett, azaz nehezen vehetünk mintákat. Az MCMC módszerek gyökere tulajdonképpen ennek a problémának az áthidalása.

A feladatunk tehát egy megfelelő $P(x, y)$ átmenetmag találása. Keressük ezt a következő alakban:

$$P(x, dy) = p(x, y) + r(x)\delta_x$$

ahol $p(x, x) = 0$, δ_x a Dirac-féle delta függvény, valamint $r(x) = 1 - \int p(x, y)dy$ annak a valószínűsége hogy egy lépés során a lánc állapota nem változik. Ez a valószínűség nem lehet 0, hiszen a kifejezésben szereplő integrál értéke nem feltétlenül 1. Ha a $p(x, y)$ függvényre teljesül a megfordíthatóság feltétel, akkor $P(x, \cdot)$ a $\pi(\cdot)$ kívánt eloszláshoz fog tartani, ha az iterációk számával tartunk a végtelenbe. Ennek belátásához értékeljük ki az első egyenlőség jobb oldalát, először helyettesítsünk be P-t:

$$\int P(x, A)\pi(x)dx = \int \left[\int_A p(x, y)dy \right] \pi(x)dx + \int r(x)\delta_x(A)\pi(x)dx =$$

cseréljük meg az integrálokat, valamint mivel δ_x A felett pontosan 1,

$$= \int_A \left[\int p(x, y)\pi(x)dy \right] dy + \int_A r(x)\pi(x)dx =$$

a megfordíthatósági feltétel miatt

$$= \int_A \left[\int p(y, x)\pi(y)dx \right] dy + \int_A r(x)\pi(x)dx =$$

$r(y)$ definíciója alapján

$$= \int_A (1 - r(y))\pi(y)dy + \int_A r(x)\pi(x)dx = \int_A \pi(y)dy.$$

A Metropolis-Hastings algoritmus célja, hogy találjon egy $p(x, y)$ függvényt, melyre teljesül a fenti tulajdonság.

3. fejezet

Metropolis-Hastings módszer

3.1. A Metropolis-Hastings algoritmus

A célunk az lesz, hogy mintákat generáljunk egy $\pi = \frac{f}{K}$ alakú sűrűségfüggvényből, ahol f egy sűrűségfüggvény, K pedig a normalizáló konstans.

Tegyük fel, hogy van egy sűrűségfüggvényünk, mellyel generálhatunk jelölteket. Legyen ez $q(x, y)$, ahol $\int q(x, y)dy = 1$. Ez szemléletesen azt jelenti, hogy ha épp az x állapotban vagyunk akkor egy y értéket generálunk $q(x, y)$-ből. Ha minden x és y esetén $q(x, y)$ kielégítené a megfordíthatósági feltételt, akkor nem is kellene tovább keresnünk, de sajnos könnyen lehet hogy például

$$\pi(x)q(x, y) > \pi(y)q(y, x).$$

Ez azt eredményezi, hogy a folyamat túl gyakran lép az x állapotból y-ba, fordítva pedig túl ritkán. Az ötlet ennek kiküszöbölésére, hogy az x állapotból y állapotba lépést egy $\alpha(x, y) < 1$ valószínűséggel határozzuk meg. Ez az α sok esetben azt eredményezi, hogy a folyamat bizonyos esetekben nem lép x állapotból y-ba, hanem marad x-ben. Így az x-ből y-ba ($y \neq x$) lépés valószínűsége:

$$p_{MH}(x, y) \equiv q(x, y)\alpha(x, y).$$

Az egyenlőtlenségből adódóan $\alpha(y, x)$-et minél nagyobbnak célszerű választani és mivel ez egy valószínűség, a felső korlátja 1.

Viszont $\alpha(x,y)$-t már nem választahatjuk szabadon, mert $p_{MH}(x,y)$-nak ki kell elégítenie a megfordíthatósági feltételt, így

$$\pi(x)q(x,y)\alpha(x,y) = \pi(y)q(y,x)\alpha(y,x).$$

Ha $\alpha(y,x) = 1$, akkor kifejezés után: $\alpha(x,y) = \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}$. Ha a relációs jel a másik irányba nézne, akkor természetesen $\alpha(x,y)$-t választanánk 1-nek és $\alpha(y,x)$-et fejeznénk ki. Ezen választás esetén kiegyensúlyoztuk a két oldalt, azaz fennáll az egyenlőség. Ezek alapján legyen

$$\alpha(x,y) = \begin{cases} \min\left(\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}, 1\right), & \text{ha } \pi(x)q(x,y) > 0 \\ 1, & \text{egyébként.} \end{cases}$$

Az átmenetmag definíciójának teljességéhez szükségünk van még arra a (nem nulla) valószínűségre, hogy a folyamat helyben marad, legyen ez:

$$r(x) = 1 - \int q(x,y)\alpha(x,y)dy.$$

Így az Metropolis-Hastings lánc átmenetmagja:

$$P_{MH}(x,y) = q(x,y)\alpha(x,y) + \left[1 - \int q(x,y)\alpha(x,y)dy\right]\delta_x.$$

Mivel $p_{MH}(x,y)$-t úgy választottuk, hogy teljesítse a megfordíthatósági feltételt, a módszer átmenetmagja tartani fog a $\pi(x)$ stacionárius eloszláshoz.

Megjegyzések:

(1) Ha egy jelölt elutasításra kerül, akkor az aktuális állapot lesz a soron következő is egyben.

(2) Szerencsére $\alpha(x,y)$ kiszámításához nem szükséges a normalizáló konstans ismerete, ugyanis megjelenik a számlálóban és a nevezőben is, így egyszerűsíthetünk vele.

(3) Ha szimmetrikus a jelölt generálás, azaz $q(x,y) = q(y,x)$, akkor az állapotváltás valószínűsége $\pi(y)/\pi(x)$, így ha $\pi(y) \geq \pi(x)$, akkor 1 valószínűséggel y állapotba kerülünk, egyébként pedig $\pi(y)/\pi(x)$ valószínűséggel. Ez szemléletesen azt jelenti, hogy ha a generált jelölt értéke nagyobb mint az éppen aktuális, akkor biztos hogy lecseréljük, ha kisebb, akkor pedig $\pi(y)/\pi(x)$ valószínűséggel.

(4) Az algoritmus alapvető pontja $q(x, y)$ megválasztása. Erre nincs általános eljárás, de sok praktikus megoldást ismerünk.

Az eljárás összegzése algoritmikus formában:

Válasszunk tetszőlegesen egy $x^{(0)} > 0$ kezdőértéket

FOR $j = 1, 2, \ldots, N$

Generáljunk y értékeket $q(x^j, \cdot)$-ből és u értékeket $U(0, 1)$-ből

IF $u \leq \alpha(x^{(j)}, y) \rightarrow$ SET $x^{(j+1)} = y$

ELSE SET $x^{(j+1)} = x^{(j)}$

RETURN $\{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\}$

Mint általában az MCMC algoritmusoknál, az értékek a vizsgált $\pi(x)$ eloszlásból vett mintának tekinthetőek, ha már a kezdő értéknek semmilyen hatása nincs a folyamatra. Természetesen ehhez szükség van az első fejezetben részletezett feltételekre, azaz a Markov lánc irreducibilis és aperiodikus mivoltára, ezek viszont nem határozzák meg a konvergencia sebességét.

3.2. Megvalósítás

Mint a megjegyzéseknél említettük, a megvalósítás szempontjából alapvető kérdés $q(x, y)$ megválasztása, melyre sajnos nincs általános eljárás. A továbbiakban pár gyakran alkalmazott lehetőséget részletezünk.

(1) A jelölt generálás egy lehetősége ha $q(x, y) = q_1(y - x)$, ahol $q_1(\cdot)$ többváltozós sűrűségfüggvény. Ezért az y jelöltet az $y = x + z$ folyamat szerint választjuk, ahol z egy q_1 szerinti valószínűségi változó. Így a jelölt egy jelenlegi és egy zaj értékének összege, ezt az esetet véletlen bolyongásnak is szokás nevezni. A véletlen mennyiség lehetséges megválasztása például a többváltozós normális vagy t-eloszlás.

Amikor q_1 szimmetrikus és $q_1(z) = q_1(-z)$, akkor az állapotváltás valószínűsége

$$\alpha(x, y) = \min\left(\frac{\pi(y)}{\pi(x)}, 1\right).$$

További fontos kérdés a jelöltgenerálás szórása, mely erősen befolyásolhatja az algoritmus hatékonyságát, ugyanis hatással van arra, hogy a folyamat mennyire gyakran fogadja el a generált jelöltet.

Tegyük fel, hogy a mintákat nagyrészt a módusz körül választjuk és a Markov lánc konvergál. Ha a szórás túl nagy, akkor egyes jelöltek túlságosan távol esnek az aktuális értéktől és így kis valószínűséggel cserélünk. A szórás csökkentésével kiküszöbölhetjük ezt a problémát, de túl kicsire sem választhatjuk azt, mert akkor az elfogadott állapotok sorozatának autokorrelációja túl nagy lesz, és így lassú a konvergencia. 2001-ben Roberts és Rosenthal megmutatta, hogy az optimális érték $\alpha = 0.23$ körül van, ha a közelíteni kívánt eloszlás azonos peremeloszlások szorzata.

(2) Választhatjuk $q(x, y)$-t $q_2(y)$-nak is, így az előző esettel ellentétben az aktuális állapottól teljesen függetlenül állítunk jelöltet, viszont ahhoz hasonlóan q_2 lehet többváltozós normális illetve t-eloszlású, de ebben az esetben meg kell adnunk a várható értéket és a szórást. Ekkor a Metropolis-Hastings hányados

$$\alpha(x, y) = \min\left(\frac{\pi(y)q_2(x)}{\pi(x)q_2(y)}, 1\right).$$

Ennek a választásnak egy érdekes esete, ha $q_2(y) = \pi(y)$, hiszen ekkor a hányados értéke pontosan 1, ezáltal a kívánt eloszlásból vesszük a mintákat.

(3) Kézenfekvő megoldás az ismert $\pi(\cdot)$ eloszlás felhasználása, amikor lehetőségünk van rá. Tegyük fel például, hogy $\pi(t) \sim v(t)h(t)$, ahol $h(t)$ eloszlásból mintákat vehetünk, $v(t)$ pedig egyenletesen korlátos, ekkor legyen $q(x, y) = h(y)$. Így csupán $v(t)$-re van szükségünk az átmenetvalószínűségek meghatározásánal. Ekkor

$$\alpha(x, y) = \min\left(\frac{v(y)}{v(x)}, 1\right).$$

4. fejezet

Gibbs Sampling

4.1. A Gibbs mintavételező eljárás

A Gibbs mintavételező eljárás a Metropolis-Hastings algoritmus egy speciális esete, így a feladat ugyanaz: hogyan konstruáljunk olyan Markov láncot, amely konvergál a kívánt eloszláshoz. Segítségével véletlen mintákat generálhatunk egy többváltozós valószínűségeloszlásból anélkül, hogy ismernénk a sűrűségfüggvényt. Ezeket a mintákat a későbbiekben használhatjuk az együttes eloszlás, egy változó peremeloszlásának vagy a változók egy részhalmazának együttes eloszlásának közelítéséhez.

Tegyük fel hogy adott egy $f(x) = (x, y_1, \ldots, y_p)$ együttes sűrűségfüggvény:

$$f(x) = \int \ldots \int f(x, y_1, y_2, \ldots, y_p) dy_1 dy_2 \ldots dy_p.$$

Számos esetben nagyon nehéz egyszerűen kiszámolni $f(x)$-et, ilyenkor használhatjuk a Gibbs mintavételezőt, hogy generáljuk egy $X_1, \ldots, X_m \sim f(x)$ sorozatot $f(x)$ ismerete nélkül. Ha elég nagy mintát generálunk, a várható érték, a szórás, vagy még akár a sűrűségfüggvény isg meghatározható egy adott hibahatáron belül.

Az eljárás bemutatásához először tekintsünk egy kétváltozós esetet. Legyenek a valószínűségi változó párok (X, Y). A Gibbs mintavételező ahelyett, hogy $f(x)$-ből generálna mintát, az (ismert) $f(x \mid y)$ és $f(y \mid x)$ feltételes eloszlásokat használja, mégpedig valószínűségi változók egy Gibbs sorozatát gyártja le:

$$Y_0', X_0', Y_1', X_1', Y_2', X_2', \ldots, Y_k', X_k'.$$

Az $Y_0^{'} = y_0^{'}$ kezdőérték ismeretében a további értékeket iteratívan határozzuk meg a következő generálási szabály szerint:

$$X_j^{'} \sim f(x \mid Y_j^{'} = y_j^{'})$$
$$Y_{j+1}^{'} \sim f(y \mid X_j^{'} = x_j^{'}).$$

Az így generált sorozatot Gibbs mintának nevezzük. Megfelőlen általános feltételek mellett $X_k^{'}$ eloszlása konvergálni fog $f(x)$-hez, ha $k \to \infty$. Így ha k elég nagy, akkor a Gibbs minta legutolsó eleme $X_k^{'} = x_k^{'}$, egy megfigyelt érték $f(x)$-ből.

4.2. Konvergencia

Egyáltalán nem nyilvánvaló, hogy egy $f(x)$ eloszlású valószínűségi változó előállítható a Gibbs minta segítségével, illetve hogy ez konvergál. Ez az iteráció Markov tulajdonságán múlik. Először nézzük a következő esetet: legyenek X és Y Bernoulli peremeloszlású valószínűségi változók a következő együttes eloszlással:

$P(X = 0, Y = 0) = p_1$, $\quad P(X = 1, Y = 0) = p_2$,
$P(X = 0, Y = 1) = p_3$, $\quad P(X = 1, Y = 1) = p_4$, ahol $p_i \geq 0$, és $\sum_i p_i = 1$.

Mátrixos alakban az együttes eloszlással:

$$\begin{pmatrix} f_{x,y}(0,0) & f_{x,y}(1,0) \\ f_{x,y}(0,1) & f_{x,y}(1,1) \end{pmatrix} = \begin{pmatrix} p_1 & p_2 \\ p_3 & p_4 \end{pmatrix}.$$

Ez alapján x peremeloszlása:

$$f_x = (f_x(0), f_x(1)) = [(p_1 + p_3), (p_2 + p_4)].$$

Ezek alapján a feltételes eloszlásokat könnyen számolhatjuk, mátrix alakban:

$$A_{y|x} = \begin{pmatrix} \frac{p_1}{p_1+p_3} & \frac{p_3}{p_1+p_3} \\ \frac{p_2}{p_2+p_4} & \frac{p_4}{p_2+p_4} \end{pmatrix},$$

valamint

$$A_{x|y} = \begin{pmatrix} \frac{p_1}{p_1+p_2} & \frac{p_2}{p_1+p_2} \\ \frac{p_3}{p_3+p_4} & \frac{p_4}{p_3+p_4} \end{pmatrix},$$

ahol $A_{y|x}$ az Y-nak $(X = x)$-re vonatkozó feltételes valószínűsége és fordítva.

Ha ezekre vonatkozóan előállítjuk a Gibbs mintát, akkor egy $0-1$ sorozatot fogunk kapni. Legyen $A_{x|y}$ az az átmenetmátrix, amely az x állapotokból az y állapotokba lépés és $A_{y|x}$ pedig az y-okból x-ekbe lépés átmenetvalószínűségeit tartalmazza.

Tekintsük csupán X peremeloszlásának közelítését, ekkor a Gibbs minta $X_i^{'}$ elemeire van szükségünk. Ahhoz, hogy $X_0^{'}$-ból $X_1^{'}$-et meghatározzuk, szükségünk van $Y_1^{'}$-re, így az iterációval kapott sorozatunk $X_0^{'} \to Y_1^{'} \to X_1^{'}$ és $X_0^{'} \to X_1^{'}$ Markov lánc lesz a következő átmenetvalószínűséggel:

$$P(X_1^{'} = x_1 \mid X_0^{'} = x_0) = \sum_y P(X_1^{'} = x_1 \mid Y_1^{'} = y) \cdot P(Y_1^{'} = y \mid X_0^{'} = x_0).$$

Általánosan, az $X^{'}$ sorozat átmenetmátrixa, $A_{x|x}$ az következő módon határozható meg:

$$A_{x|x} = A_{y|x} A_{x|y}.$$

Ennek segítségével bármely k-ra meghatározhatjuk $X_k^{'}$-t, mert az átmenetmátrix alapján $P(X_k^{'} = x_k \mid X_0^{'} = x_0)$ pontosan $(A_{x|x})^k$. Továbbá, ha $X_k^{'}$ peremeloszlását

$$f_k = [f_k(0), f_k(1)]$$

alakban írjuk fel, akkor akkor minden k-ra

$$f_k = f_0 (A_{x|x})^k = (f_0 (A_{x|x})^{k-1}) A_{x|x} = f_{k-1} A_{x|x}.$$

Ha $A_{x|x}$ minden eleme pozitív, akkor a fentiek alapján minden f_0 kezdeti valószínűség esetén, ha $k \to \infty$, f_k konvergál az f stacionárius elolszláshoz, melyre teljesül, hogy

$$f A_{x|x} = f.$$

Emiatt f az X változó peremeloszlása. Ez alapján kellően sokszor iterálva tetszőlegesen közel kerülhetünk a X peremeloszlásához, azaz $k \to \infty$ esetén $X_k^{'}$ közelítőleg f_x.

Az eddig tárgyalt 2×2-es eset könnyen átvihető bármilyen $n \times m$-es együttes eloszlásra. Hasonlóan definiálhatjuk az $n \times n$-es $A_{x|x}$ átmenetmátrixot, melynek stacionárius

eloszlása X peremeloszlása lesz, viszont folytonos X és Y esetén kisebb módosításokra van szükségünk. $X_1^{'}$ $X_0^{'}$-ra vonatkozó feltételes eloszlását a következő alakban írjuk fel:

$$f_{X_1'|X_0'}(x_1 \mid x_0) = \int f_{X_1'|Y_1'}(x_1 \mid y) f_{Y_1'|X_0'}(y \mid x_0) dy,$$

valamint a feltételes eloszlások: $X_1^{'} \mid X_0^{'}$, $X_2^{'} \mid X_3^{'}$, $X_1^{'} \mid X_0^{'}$, ... $A_{x|x}$-hez hasonlóan az átmenetmátrixnak ki kell elégítenie a következő egyenlőséget:

$$f_{X_k'|X_0'}(x \mid x_0) = \int f_{X_k'|X_{k-1}'}(x \mid t) f_{X_{k-1}'|X_0'}(t \mid x_0) dt.$$

Itt $f_{X_k'|X_{k-1}'}$ személeletesen az átmenet eloszlása. Hasonlóan a Bernoulli eloszlású változókhoz, ha $k \to \infty$, f_k szintén a stacionárius eloszláshoz tart, ami pedig X peremeloszlása lesz.

4.3. Két- és többváltozós eset

Először tegyük fel, hogy adottak X és Y valószínűségi változók, valamint $f_{X|Y}(x \mid y)$ és $f_{Y|X}(y \mid x)$ feltételes eloszlásfüggvények. Határozzuk meg az $f_X(x)$ peremeloszlást. Definíció szerint

$$f_X(x) = \int f_{XY}(x, y) dy,$$

ahol $f_{XY}(x, y)$ az ismeretlen együttes eloszlás. Mivel $f_{XY}(x, y) = f_{X|Y}(x \mid y) f_Y(y)$, behelyettesítve

$$f_X(x) = \int f_{X|Y}(x \mid y) f_Y(y) dy.$$

Ha $f_Y(y)$-t hasonlóan helyettesítjük

$$f_X(x) = \int f_{X|Y}(x \mid y) \int f_{Y|X}(y \mid t) f_X(t) dt dy =$$
$$\int \Big[f_{X|Y}(x \mid y) f_{Y|X}(y \mid t) dy \Big] f_X(t) dt = \int h(x, t) f_X(t) dt,$$

ahol $h(x, t) = \big[f_{X|Y}(x \mid y) f_{Y|X}(y \mid t) dy \big]$. Ennek az integrálegyenletnek $f_X(x)$ egyértelmű megoldása.

Ezáltal ha $k \to \infty$,

$$f_{X'_k|X'_0}(x \mid x_0) \to f_X(x),$$

valamint

$$f_{X'_k|X'_{k-1}}(x \mid t) \to h(x,t).$$

Sajnos annak ellenére, hogy X és Y együttes eloszlása meghatározza a feltételes- és peremeloszlásokat, a megfelelő feltételes eloszlások nem határozzák meg a megfelelő peremeloszlásokat és ezáltal az együttes eloszlást sem.

Ha több változónk van, a helyzet bonyolultabbá válik, hiszen nem minden feltételes- és peremeloszlás pár határozza meg az együttes eloszlást, ezáltal többféle módon is felírhatjuk a fenti integrálegyenletet, így a feltételes eloszlások különböző halmazait használhatjuk, hogy meghatározzuk a kívánt peremeloszlást.

Tegyük fel, hogy $f_X(x)$-et keressük, valamint X,Y és Z a valószínűségi változók. A fenti integrálegyenlethez hasonlóan, ha az (Y,Z) párt egy valószínűségi változóként értelmezzük:

$$f_X(x) = \int \left[\int f_{X|YZ}(x \mid y,z) f_{YZ|X}(y,z \mid t) dydz \right] f_X(t) dt.$$

$f_{X|YZ}$ és $f_{YZ|X}$ váltakozása szintén előállít egy sorozatot, mely az $f_X(x)$ eloszláshoz konvergál. Ezzel szemben a Gibbs mintavételező iteratívan venne mintákat $f_{X|YZ}$, $f_{Y|XZ}$ és $f_{Z|XY}$-ból, így a k-adik iteráció után:

$$X'_j \sim f(x \mid Y'_j = y'_j, Z'_j = z'_j)$$
$$Y'_{j+1} \sim f(y \mid X'_j = x'_j, Z'_j = z'_j)$$
$$Z'_{j+1} \sim f(z \mid X'_j = x'_j, Y'_{j+1} = y'_{j+1}).$$

Ez az iteráció a következő sorozatot gyártja le:

$$Y'_0, Z'_0, X'_0, Y'_1 Z'_1, X'_1, Y'_2, Z'_2, X'_2, \ldots$$

és ha k-t megfelelően nagynak választjuk, $X'_k = x'_k$ közelítőleg $f(x)$ egy megfigyelt értéke.

Mivel a Gibbs mintavételező felhasználja az összes egyváltozós eloszlást, ezek meghatározzák az együttes és az összes feltételes eloszlást, ezért az integrálegyenlet megoldható az iterációs séma segítségével.

4.4. Megvalósítás

A megvalósítás során az alapvető kérdés az, hogy milyen nagynak válasszuk a k-t, hogy az eltérés egy bizonyos határon belül maradjon. A legáltalánosabb módszer a Gibbs minta konvergenciájának vizsgálata bizonyos tényezőktől függően. Generálhatunk m darab független Gibbs mintát és választhatjuk úgy a k-t, hogy ezek a minták valamilyen statisztikai próba szerint azonos eloszlásúak legyenek, vagy vizsgálhatjuk a közelítő és a közelített eloszlások közötti különbségek változását. A megfelelő k megválasztása ma is számos kutatás tárgyát képezi.

5. fejezet

Slice sampling

A feladatunk hasonló az eddigiekhez, szeretnénk egy $p(x) = \frac{f(x)}{K}$ alakú eloszlásból mintát vételezni, ahol $x \in \mathbb{R}^n$. A slice sampling ötlete, hogy ezt megtehetjük úgy, hogy egyenletes eloszlású mintákat veszünk az $f(x)$ grafikonja alatti $(n+1)$ dimenziós tartományból. Az ötlet formalizálásához vezessünk be egy y segédváltozót és definiáljuk x és y együttes eloszlásfüggvényét úgy, hogy egyenletes legyen az $U = \{(x, y) : 0 < y < f(x)\}$ által meghatározott tartományon. Ezáltal (x, y) együttes eloszlása:

$$p(x,y) = \begin{cases} 1/Z & \text{ha } 0 < y < f(x) \\ 0 & \text{egyébként,} \end{cases}$$

ahol $Z = \int f(x)dx$. Az x peremeloszlása ekkor

$$p(x) = \int_0^{f(x)} 1/Z dy = f(x)/Z.$$

Így (x, y)-ra vonatkozó mintákat fogunk gyártani, majd egyszerűen figyelmen kívül hagyjuk y-t. Vegyük észre, hogy ezáltal a slice sampling egy speciális esete a Gibbs mintavételezőnek, ugyanis ha (X, Y) az együttes eloszlás, akkor az

$$Y \mid X = x \sim U(0, f(x)),$$

$$X \mid y \sim U(S), \text{ ahol } S = \{x : y < f(x)\}$$

egy Gibbs sorozat.

5.1. Egyváltozós eset

A legegyszerűbb eset az, ha a keresett eloszlásfüggvény egyváltozós. A továbbiakban többváltozós eloszlásfüggvényeket fogunk tekinteni, de az n-dimenziós mintát koordinátánkénti mintavételezéssel fogjuk legyártani. Minden x_i koordináta esetében meg kell határoznunk $f_i(x_i)$ értéket, mely a K normalizáló konstanstól eltekintve $p(x_i \mid \{x_j\}_{j \neq i})$.

Az egyszerűség kedvéért x jelöljön egy n-dimenziós vektort, $f(x)$ pedig jelöle az n-dimenziós eloszlásfüggvényt. A következő eljárás az aktuális x_0 vektort x_1-re cseréli:

1 Generáljunk egy $y \sim U(0, f(x_0))$ egyenletes eloszlású valós számot, ezzel meghatározunk egy vízszintes S szeletet: $S = \{x : y < f(x)\}$. x_0 mindig S-en belül lesz.

2 Keressünk egy $E = (L, R)$ intervallumt x_0 körül. Ha $x_0 \in S$, akkor x_0 egy környezete mindig tartalmazni fogja a szelet valamekkora részét.

3 Generáljunk egy x_1 egyenletes eloszlású véletlen pontot az E intervallumon.

Az első lépésben tehát generálunk egy y értéket és meghatározzuk a vízszintes szeletet. A második és harmadik lépést különböző módokon is megvalósíthatjuk, de ezeknek természetesen egy megfelelő Markov láncot kell eredményezniük.

Lehetőségek alkalmas E intervallum keresésére:

1 A lehető legjobb megoldás az lenne, ha $L = \inf(S)$ és $R = \sup(S)$ lenne, de ez általában nem kivitelezhető.

2 Ha a lehetséges x értékek korlátosak, akkor E-t ehhez az korláthoz hasonlóan állítjuk be.

3 Ha van egy w becslésünk az S méretére, véletlenszerűen kiválasztunk egy w hosszúságú intervallumot x_0 körül, majd addig növeljünk, amíg le nem fedi S-t, például a stepping out eljárással.

4 Hasonlóan, véletlenszerűen kiválasztunk egy w hosszúságú intervallumot x_0 körül, majd növeljük a duplázó eljárással.

Mindegyik eljárás esetén meg kell határoznunk egy A halmazt, mely az elfogadható jelölteket tartalmazza, a következő szerint:

$$A = \{x : x \in S \cap E \wedge P(E \mid x) = P(E \mid x_0)\}.$$

Ez természetes elvárás, ugyanis az elfogadható jelöltnek a szelet és a választott intervallum metszetében kell lennie, illetve az aktuális és a soron következő állapotot feltéve ugyanolyan valószínűséggel kell teljesülnie, hogy E-ben vagyunk.

Könnyen láthatjuk, hogy az 1. és 2. esetben $A = S$. Az 1. esetben E-t pont így állítottuk be, de ez csak akkor hatékony, ha valamilyen módszer segítségével elő tudjuk állítani $y = f(x)$ összes megoldását. A 2. lehetőséget könnyű megvalósítani, ha a lehetséges x jelöltek halmaza korlátos, viszont nem túl hatékony, ha a szelet általában sokkal kisebb ennél a korlátnál.

A 3. pontban említett stepping out eljárás algoritmusa:

Input: f függvény, x_0 pont, a szeletet meghatározó y érték, a szelet méretére vonatkozó w becslés és az m érték, a szelet maximális mw méretére vonatkozólag.

Output: (L, R) intervallum.

Generáljunk $U \sim U(0,1)$ és $V \sim U(0,1)$ egyenletes eloszlású véletlen számokat

$L \leftarrow x_0 - w \cdot U$; $R \leftarrow L + w$; $J \leftarrow [m \cdot U]$; $K \leftarrow (m-1) - J$

REPEAT WHILE ($J > 0$ AND $y < f(L)$):

$L \leftarrow L - w$; $J \leftarrow J - 1$

REPEAT WHILE ($K > 0$ AND $y < f(R)$):

$R \leftarrow R + w$; $K \leftarrow K - 1$

5.1. Megjegyzés. *Itt* $[m \cdot V]$ *az* $m \cdot V$ *egész részét jelöli.*

A stepping out eljárást abban az esetben alkalmazhatjuk, ha tudunk mondani egy w becslést az S szelet jellemző méretére. Ekkor mw a keresett intervallum hosszának egy felső korlátja, ahol $m \in \mathbb{Z}^+$. Speciális esetben $m = 1$, ekkor az intervallumot mindig w hosszúságúra állítjuk be, így nem szükséges meghatározni f értékét az intervallum végpontjaiban. Nagyon fontos, hogy az intervallum kezdeti elhelyezkedése, valamint az intervallum bal és jobb irányba történő növelésének elosztása véletlenszerű legyen, mert ez biztosítja, hogy a végeredményben kapott intervallum bármelyik kezdeti $x_0 \in S \cap E$ pontból előállítható legyen.

A 4. pontban említett doubling eljárás algoritmusa:

Input: f függvény, x_0 pont, a szeletet meghatározó y érték, a szelet méretére vonatkozó w becslés és a p érték, a szelet maximális $2^p w$ méretére vonatkozólag.

Output: (L, R) intervallum.

Generáljunk egy $U \sim U(0,1)$ egyenletes eloszlású véletlenszámot.

$L \leftarrow x_0 - w \cdot U$; $R \leftarrow L + w$; $K \leftarrow p$

REPEAT WHILE ($K > 0$ AND $\{y < f(L)$ OR $y < f(R)\}$:

Generáljunk egy $V \sim U(0,1)$ egyenletes eloszlású véletlenszámot.

IF $V < \frac{1}{2}$

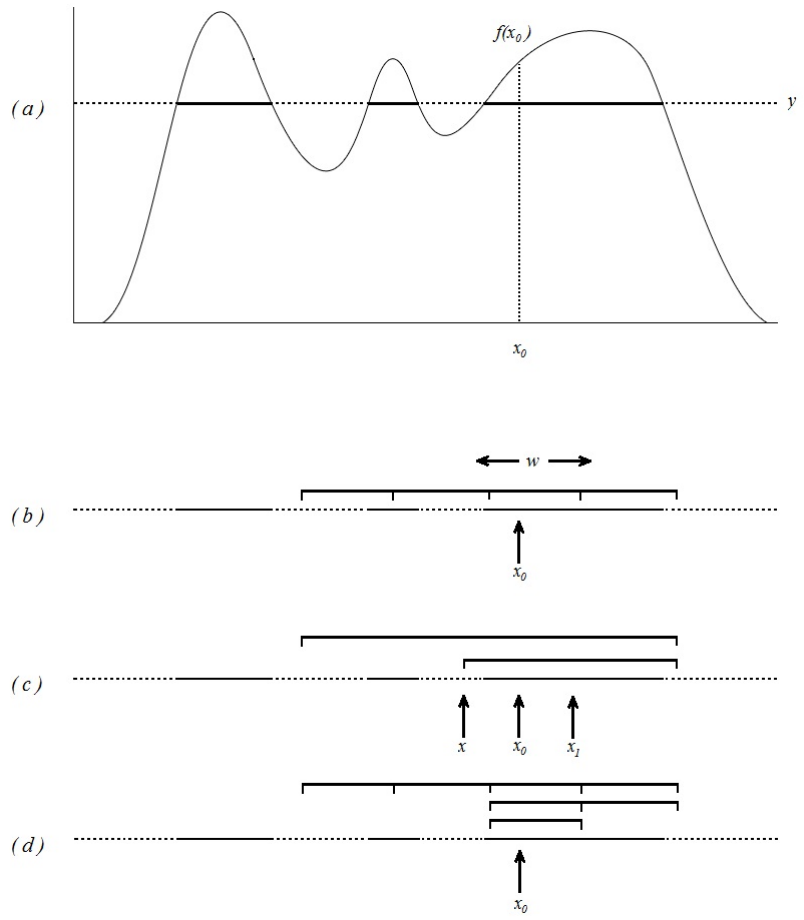
THEN $L \leftarrow L - (R - L)$

ELSE $R \leftarrow R + (R - L)$

$K \leftarrow K - 1$

Szemléletesen minden lépésben egy intervallumot gyártunk le, melynek hossza az előző intervallum hosszának kétszerese. Ezt addig ismételjük, amíg a soron következő intervallum mindkét végpontja a szeleten kívül esik, vagy az előre meghatározott lépésszámot el nem érjük. Fontos megjegyezni, hogy minden lépésben véletlenszerűen döntjük el, hogy melyik irányban növeljük az intervallumot (emiatt az egyik végpont változatlan marad), attól függetlenül, hogy az adott irány szerinti végpont a szeleten belül vagy azon kívül helyezkedik el. A véletlenszerűség a doubling eljáráshoz hasonlóan alapvető, mert itt is ez garantálja, hogy az intervallum más kezdeti $x_0 \in S \cap E$ pontból is előállítható legyen. Sajnos ebben az esetben viszont nem minden $x \in S \cap E$ pontra teljesül, hogy onnan indulva előállíthatjuk ugyanazt az intervallumot, ezért szükséges ellenőriznünk, hogy a következő állapot A-beli vagy sem.

A stepping out eljárással szemben ez a módszer gyorsan növeli a kezdeti intervallum méretét, ami nagyon hatékony, ha a szelet becsült w értéke kicsi, ugyanis hasonlóan kicsi kezdeti intervallummal indulunk és mivel exponenciálisan nő, hamar lefedjük a szeletet.



5.1.1. ábra: A stepping out, shrinkage és doubling eljárások.
Az (a) részben meghatározzuk y-t. A (b) részben egy w hosszúságú intervallumot véletlenszerűen elhelyezünk x_0 körül, majd w méretű intervallumok hozzáadásával addig növeljük, amíg az intervallum mindkét vége a szeleten kívül helyezkedik el. A (c) részben addig csökkentjük az intervallum méretét, amíg az új x_1 pont az szeleten belül nem helyezkedik el. A (d) részben kétszer duplázzuk az intervallum méretét, ekkor már mindkét vége a szeleten kívül esik.

Ha megtaláltuk a megfelelő intervallumot, akkor még generálnunk kell egy véletlen számot, mely a szeletre esik:

1 Egymás után generáljunk egyenletes eloszlású véletlen számokat az E intervallum egész addig, amíg nem kapunk egy A-belit.

2 Egymás után generáljunk egyenletes eloszlású véletlen számokat az E kiinduló intervallumon, majd a shrinkage módszerrel folyamatosan csökkentjük E-t egészen addig, amíg nem kapunk egy A-belit.

A 2. pontban említett shrinkage eljárás algoritmusa:

Input: f függvény, x_0 pont, a szeletet meghatározó y érték, (L, R) intervallum.
Output: x_1 pont.

$L' \leftarrow L$; $R' \leftarrow R$

REPEAT

Generáljunk egy $U \sim U(0,1)$ egyenletes eloszlású véletlen számot.

$x_1 \leftarrow L' + U \cdot (R' - L')$

IF $y < f(x_1)$ AND Accept(x_1)

THEN EXIT LOOP

IF $x_1 < x_0$

THEN $L' \leftarrow x_1$

ELSE $R' \leftarrow x_1$

5.2. Megjegyzés. *Itt az Accept(x_1) egy tesztet jelöl, arra vonatkozólag hogy $x_1 \in S \cap E$ vagy sem.*

Az algoritmus futása során minden egyes lépésben, amikor olyan pontot kapunk, mely nincs benne az $S \cap E$ halmazban, az aktuális intervallum összezsugorodik, mégpedig úgy, hogy az aktuális pont lesz az egyik végpont. Mivel x_0 mindig A-beli, a kapott intervallumban lesz A-beli pont, így a folyamat biztosan végetért. Az intervallum folyamatos csökkentése miatt a sorsolt pontok várható száma kicsi, ezért ez a módszer általában jól használható.

Az Accept(x_1) teszt:

Input: f függvény, x_0 pont, x_1 jelölt, a szeletet meghatározó y érték, a szelet méretére vonatkozó w becslés, w becsléssel és a doubling eljárással talált (L, R) intervallum.
Output: x_1 elfogadható következő pontnak vagy sem.

$L' \leftarrow L$; $R' \leftarrow R$

$D \leftarrow$ FALSE

REPEAT WHILE $R' - L' > 1,1 \cdot w$

$M \leftarrow (L' + R')/2$

IF $\{x_0 < M$ AND $x_1 \geq M\}$ OR $\{x_0 \geq M$ AND $x_1 < M\}$

THEN $D \leftarrow$ TRUE

IF $x_1 < M$

THEN $R' \leftarrow M$

ELSE $L' \leftarrow M$

IF D AND $y \geq f(L')$ AND $y \geq f(R')$

THEN nem fogadjuk el az új pontot

Az új pontot akkor fogadjuk el, ha a fenti eljárás során nem lett elutasítva

5.3. Megjegyzés. *Az* $1,1$ *szorzó a numerikus számítás során létrejövő kerekítési hiba kiküszöbölésére szolgál.*

Az eljárás ellenőrzi a doubling módszerrel legyártott intervallumokat, mégpedig vizsgálja hogy azok végei a szeleten kívül esnek-e vagy sem, ugyanis előbbi az algoritmus korábbi befejezését jelentené.

Ha a megfelelő intervallumot az $(1) - (2) - (3)$ módszerek valamelyikével kerestük, akkor $A = S \cap E$. A (4) eljárással lehet, hogy az A az $S \cap E$ egy részhalmaza.

5.2. Többváltozós eset

Ahelyett, a többváltozós eloszlás minden x_i koordinátájára alkalmaznánk az eddig ismertett módszerek egyikét, a megfelelő fogalmak n-dimenziós változatának használatával értelmezhetjük a módszereket magára az n-dimenziós eloszlásra. Az eddig használt $E = (L, R)$ intervallum helyett vegyünk egy n-dimenziós téglát:

$$H = \{x : L_i < x_i < R_i \ \forall \ i = 1, \ldots, n\}.$$

Az $x_1 = (x_{1,1}, x_{1,2}, \ldots, x_{1,n})$ következő állapot megtalálása az aktuális $x_0 = (x_{0,1}, x_{0,2}, \ldots, x_{0,n})$ mellett hasonló az egyváltozós esethez:

1 Generáljunk egy $y \sim U(0, f(x_0))$ egyenletes eloszlású valós számot, ezáltal meghatározunk egy S szeletet: $S = \{x : y < f(x)\}$.

2 Keressünk egy $H = (L_1, R_1) \times (L_2, R_2) \times \ldots \times (L_n, R_n)$ n-dimenziós téglát x_0 körül.

3 Generáljunk egy egyenletes eloszlású x_1-et az S szelet azon részén, ami a H téglán belül van.

Megfelelő H tégla kereséséhez az egyváltozós esethez hasonlóan ideális lenne az S-et tartalmazó legkisebb H megtalálása, de ez nehezen kivitelezhető. Ha minden változó korlátos, akkor H-t a korlátokhoz hasonlóan állíthatjuk be, de ez mint az egyváltozós esetben, itt sem hatékony, ha az S szelet sokkal kisebb. Ha minden tengely mentén van egy w_i becslésünk az S lehetséges méretére, akkor véletlenszerűen választhatunk egy $w = [w_i]$ téglát az x_0 pont körül, majd növelhetjük valamilyen eljárással. A shrinkage eljárás során felmerül az a nehézség, hogy mivel az n-dimenziós téglának 2^n csúcsa van, nagy n esetén túl sok tesztet kéne végeznünk. A stepping out módszer is túlzottan időigényes lehet az n lehetséges irány miatt. A doubling eljárás megvalósításánál szintén akkor állunk meg, amikor egy egyenletes eloszlás szerinti véletlenszám az S szeleten kívül esik.

5.3. Egyenletes eloszlás egy kétdimenziós tartományon

Mint az eddigi módszereknél, a slice sampling alkalmazhatósága azon múlik, hogy a generált Markov lánc stacionárius eloszlása megegyezik-e a szóban forgó eloszlásfüggvénnyel. Ezt nem láttuk be a fenti esetekben, viszont adunk egy szemléletes és heurisztikus bizonyítást a kétdimenziós esetre.

Legyen T egy tartomány a síkon, melynek határa nullmértékű. Generáljunk ezen a tartományon egyenletes eloszlású $v = (x, y)$ koordinátájú véletlen pontot a következő átmenetszabály szerint:

1 Az első lépésben válasszunk tetszőlegesen egy vízszintes egyenest, melynek van olyan szakasza, ami a tartományon belül van. Állítsuk be az y koordinátát $y =$

y_1-re. Válasszuk ki tetszőlegesen az egyik l_1 hosszúságú szakaszt, melynek bal végpontja legyen L_1, a jobb pedig R_1. Generáljunk egy $u_1 = U(0, l_1)$ egyenletes eloszlású véletlenszámot. Állítsuk be $x_1 = L_1 + u_1$-t.

2 Az (x_1, y_1) ponton átmenő függőleges egyenes mentén legyen az (x_1, y_1)-t tartalmazó, m_1 hosszúságú szakasz alsó végpontja legyen D_1, a felső pedig U_1. Generáljunk egy $u_2 = U(0, m_1)$ egyenletes eloszlású véletlenszámot. Állítsuk be $y_2 = D_1 + u_2$-t.

3 Az (x_1, y_2) ponton átmenő vízszintes egyenes mentén legyen az (x_1, y_2)-t tartalmazó, m_2 hosszúságú szakasz bal végpontja legyen L_2, a jobb pedig R_2. Generáljunk egy $u_3 = U(0, m_2)$ egyenletes eloszlású véletlenszámot. Állítsuk be $x_2 = L_2 + u_3$-t.

4 A (2)-(3) lépéseket ismételjük egymás után.

Tehát minden páratlan lépésben vízszintesen választunk egy új pontot, minden párosban pedig függőlegesen. Így egy Markov láncot kapunk, melyre teljesülnek a megfelelő feltételek és stacionárius eloszlása egyenletes. Lássuk be ezt az állítást a Markov láncokra vonatkozó feltételek felhasználása nélkül.

Fedjük le a tartományt egy q lépéstávolságú rácshálóval és tekintsük a rácsháló által meghatározott négyzeteket, ezek közül is azokat, melyekre van olyan $(x, y) \in T$, hogy az (x, y) pont a négyzet belső pontja. Ezen négyzetek középpontjait kössük össze az oldallapjaikkal szomszédos négyzetek középpontjaival. Bolyongjunk (a paramétereket diszkrét esetben értelmezve) a négyzeteken a fenti átmenetszabály szerint. Világos, hogy ha csak két szomszédos négyzetünk van (akár vízszintesen, akár függőlegesen), mindkettőben $\frac{1}{2}$-$\frac{1}{2}$ valószínűséggel vagyunk, a teljes rácsháló pedig ilyen egymás melletti négyzetek halmaza. Azon pontok esetében, melyeket két négyzet csúcsa fed csupán, egészítsük ki a lefedést két tovább négyzettel, így az ilyen pontoknak lesz olyan sugarú környezete, melyben minden pont valamely négyzet belső pontja. Így ennek a láncnak a stacionárius eloszlása egyenletes lesz és ha a felosztást minden határon túlmenően finomítjuk ($q \to 0$), akkor az egész T tartományon egyenletes az eloszlás.

A fenti eset az általánosság csekély korlátossága mellett érvényes, de nyilvánvalóan kiterjeszthetjük általánosabb idomokra is.

6. fejezet

Perfect sampling

6.1. Az alapötlet

Az eddig tárgyalt módszereknél általános problémát jelentett, hogy milyen nagynak válasszuk t-t, hogy X_t közelítőleg egy minta legyen a π eloszlásból. A megvalósítás szempontjából további nehézséget jelent X_t és X_{t+1} korrelációjának meghatározása, mely a közelítés szórásához szükséges. A Perfect sampling mindkét problémát orvosolja azáltal, hogy pontosan a π eloszlásból gyárt független mintákat. Az ötlet az MCMC módszerekre általánosan jellemző probléma kiküszöbölésén nyugszik: ha megfelelően hosszú ($t \to \infty$) ideig futtatjuk a láncot, akkor megkapjuk a keresett eloszlást, de a számítási kapacitás és a tárhely korlátossága miatt meg kell állnunk egy bizonyos időpillanatban. A Perfect sampler véges sok lépésben legyártja pontosan azt a Markov láncot, melynek végtelen sok lépést kellene tennie azáltal, hogy lecseréljük a kezdeti időpillanatot $-\infty$-re, ∞-t pedig 0-ra. Így a futtatási idő $t = \infty$, a lánc stacionárius eloszlású a $t = 0$ időpillanatban és ekkor pontosan a keresett π eloszlásból vételezzük a mintát.

Először is egy olyan X_t-t kell keresnünk, mely független a kiindulási állapottól. Ezt több Markov lánc egyidejű futtatásával fogjuk elérni. Tegyük fel, hogy az I állapottérnek k darab állapota van és indítsunk a $t = 0$ időpillanatban k láncot minden különböző állapotból, ezek lesznek a párhuzamos láncaink. Ezeket kapcsoljuk össze egy ϕ átmenetszabállyal és U_t véletlen számokkal, ahol ϕ meghatározza X_{t+1}-et, mint X_t és U_{t+1} függvényét. Minden lánc esetében ugyanazt a ϕ függvényt és $\ldots, U_t, U_{t+1}, \ldots$ számokat használjuk, ahol általában $U_{t+1} \sim U(0,1)$ valamint $X_{t+1} = \phi(x_t, u_{t+1}) = F^{-1}_{X_{t+1}|x_t}(u_{t+1})$

az átmenetmag és az állapottér lineáris rendezése szerinti $X_{t+1} \mid x_t$ inverz eloszlásfüggvénye. Jelölje $X^{s,j} \equiv {X_t^{s,j}}_{t \geq s}$ azt a Markov láncot, mely a j állapotból indul az s időpillanatban.

6.1. Definíció. *Azt mondjuk, hogy a Markov láncok egyesültek a t időpillanatban, ha $X_t^{s,1} = X_t^{s,2} = \ldots = X_t^{s,k}$.*

6.1. Tétel. *Tegyük fel, hogy $X^{s,1}$, $X^{s,2}$,...,$X^{s,k}$ összekapcsolt Markov láncok, ahol*

1. *$X^{s,j}$ a j-edik állapotból az s időpillanatban indul, és*
2. *$X_{t+1}^{s,j} = \phi(X_t^{s,j}, U_{t+1})$, ahol az U_t-k páronként függetlenek.*

Ekkor

1. *Az egyesülés T időpillanata egy valószínűségi változó, ami csak U_1, U_2,... függvénye, valamint*
2. *az egyesülés állapota az X_T valószínűségi változó független a kezdeti állapottól.*

Az első állítás azonnal következik a ϕ konstrukciójából, a második pedig annak a következménye, hogy X_T nem függ j-től mivel csak U_1,...,U_T függvénye. A második állítás jelentése, hogy T az az időpillanat, amikor X_T már teljesen független a kiindulási állapottól, de sajnos azt nem mondhatjuk, hogy X_T egy minta a π stacionárius eloszlásból. Ha T^* egy fix időpillanat és X_{T^*} független $X_s^{s,j}$-től, akkor $X_{T^*} \sim \pi$, de mivel T egy valószínűségi változó, $X_T \nsim \pi$.

6.2. A Coupling From The Past (CFTP) algoritmus

Propp és Wilson nevéhez kötődik a CFTP algoritmus, mely fix futtatási időben kihasználja a láncok egyesülését, ezáltal olyan valószínűségi változót gyárt le, melynek pontosan a keresett π az eloszlásfüggvénye.

Tegyük fel, hogy egy láncot a $t = -\infty$ időpillanatban indítottunk valamilyen $X_{-\infty}$ állapotból. Ekkor a $t = 0$ időpillanatban (a végtelen sok lépés következtében) az X_0 egy minta a π stacionárius eloszlásból. Az implementálás az egyesülés megfontolásán alapszik: először keressünk egy olyan $-T$ -t, amire $X_0 \equiv X_0^{-T,j}$ nem függ $X_{-T}^{-T,j}$-től, tehát az egyesülés megtörtént a $-T$ és a 0 időpillanat között, majd minden lehetséges

állapotból futtatunk egy láncot a $t = -T$ időpillanattól $t = 0$-ig, ezáltal megkapjuk X_0-t. A feladatunk tehát egy megfelelő $-T$ és X_0 keresése, mely a következő módon történik:

1. Indítsuk az $X^{-1,1}$,$X^{-1,2}$,...,$X^{-1,k}$ láncokat a $t = -1$ időpillanatban (ahol az állapottér elemszáma k) és generáljunk egy U_0-t.

2. Az $X_0^{-1,j} = \phi(X_{-1}^{-1,j}, U_0)$ átmenetszabály segítségével tegyünk egy lépést. Ha a láncok egyesültek a $t = 0$ pillanatban, akkor $T = -1$ és X_0 egy minta az π eloszlásból.

3. Egyébként indítsuk az $X^{-2,1}$,$X^{-2,1}$,...,$X^{-2,k}$ láncokat a $t = -2$ pillanatban és generáljunk egy U_{-1}-et, majd az $X_{-1}^{-2,j} = \phi(X_{-2}^{-2,j}, U_{-1})$ és $X_0^{-2,j} = \phi(X_{-1}^{-2,j}, U_0)$ átmenetszabályok segítségével tegyünk két lépést. Ha a láncok egyesültek a $t = 0$ pillanatban, akkor $T = -2$ és X_0 egy minta a π eloszlásból.

4. Egyébként vegyük sorban a $t = -3,-4,\ldots$ pillanatokat és alkalmazzuk az eljárást iteratívan, amíg a láncok nem egyesülnek a $t = 0$-ban.

6.1. Megjegyzés. *Fontos, hogy a harmadik lépésben ugyanazt az U_0-at használjuk, mint az elsőben.*

A fenti algoritmus legyárt egy valószínűségi változót, melynek eloszlása pontosan megegyezik a Markov lánc eloszlásával. Az eljárással megkeresett T és X_0 viszont nem független változók.

6.3. Fontainebleau lehulló levelei

Egy látványos geometriai alkalmazása a CFTP módszernek a fákról lehulló levelek elhelyezkedésének modellezése, melyet Fontainebleau fái ihlettek. A modell egy véletlenszerűen elkészített mozaik kialakulásának folyamatát írja le, melyben egy üveglapra hulló levelek által alkotott képpel közelítjük a stacionárius eloszlást. A levelek lehullásának folyamatát jól modellezi egy Markov lánc, melynek elemei a különböző minták, melyeket a lehulló levelek alkotnak az üveglapon. David Wilson nevéhez kötődik a CFTP algoritmus egy módosítása, mely lépésenként építi fel a moziakot, visszakövetés nélkül, a végső képet pedig fokozatosan éri annak a ténynek a felhasználásával,

hogy minden újabb lehulló levél a későbbiekben csökkenti a kép módosulását az adott területen.

6.3.1.ábra: Fontainebleau lehulló levelei In: [12]

Tekintsünk a levelek alkotta képre az üveglap aljáról, ahelyett hogy felülről szemlélnénk. Így abban a pillanatban megszűnik változni a kép, amint az üveglap minden egyes pontja le van fedve egy levél által, ezért ekkor az aktuális állapot egy minta a keresett stacionárius eloszlásból.

6.4. Sandwiching

A CFTP algoritmus kiküszöböli a futási idő hosszának problémáját, de a gyakorlatban nehezen alkalmazható, ha az I állapottér k elemszáma túl nagy. A sandwiching ötlete a leginkább elterjedt és talán a leghatékonyabb eljárás a CFTP gyakorlati alkalmazásához nagy állapottér esetén. Ezt azonban csak olyan Markov láncok esetén haszhálhatjuk, melyek kielégítenek bizonyos monotonitási feltételeket, illetve melyek állapotterére definiálhatunk valamilyen rendezést.

Az ötlet bemutatásához tekintsük a következő példát:

Legyen k rögzített, az állapottér $I = 1, \ldots, k$, az átmenetmátrix pedig:

$$\left[p_{ij}\right]_{i,j\in I} = \begin{cases} \frac{1}{2}, & \text{ha} \quad i,j = 1 \\ \frac{1}{2}, & \text{ha} \quad |i-j| = 1 \\ 0, & \text{egyébként.} \end{cases}$$

Szemléletesen, ha az állapottér elemeit 1,2,...,k sorban vesszük, akkor a lánc minden lépésben eggyel nagyobb vagy alacsonyabb indexű állapotba lép $\frac{1}{2}$-$\frac{1}{2}$ valószínűséggel, illetve ha a legnagyobb vagy a legkisebb indexű állapotban vagyunk, akkor $\frac{1}{2}$ valószínűséggel helyben marad és $\frac{1}{2}$ valószínűséggel az eggyel kisebb illetve nagyobb indexű állapotba lép. Könnyen láthatjuk, hogy a lánc stacionárius eloszlása:

$$\pi_i = \frac{1}{k} \quad i = 1, \ldots, k.$$

Ezek alapján legyen az átmenetszabály a következő:

$$\phi(1,u) = \begin{cases} 1 & \text{ha} \quad u \in \left[0, \frac{1}{2}\right) \\ 2 & \text{ha} \quad u \in \left[\frac{1}{2}, 1\right], \end{cases}$$

$$\phi(k,u) = \begin{cases} k-1 & \text{ha} \quad u \in \left[0, \frac{1}{2}\right) \\ k & \text{ha} \quad u \in \left[\frac{1}{2}, 1\right], \end{cases}$$

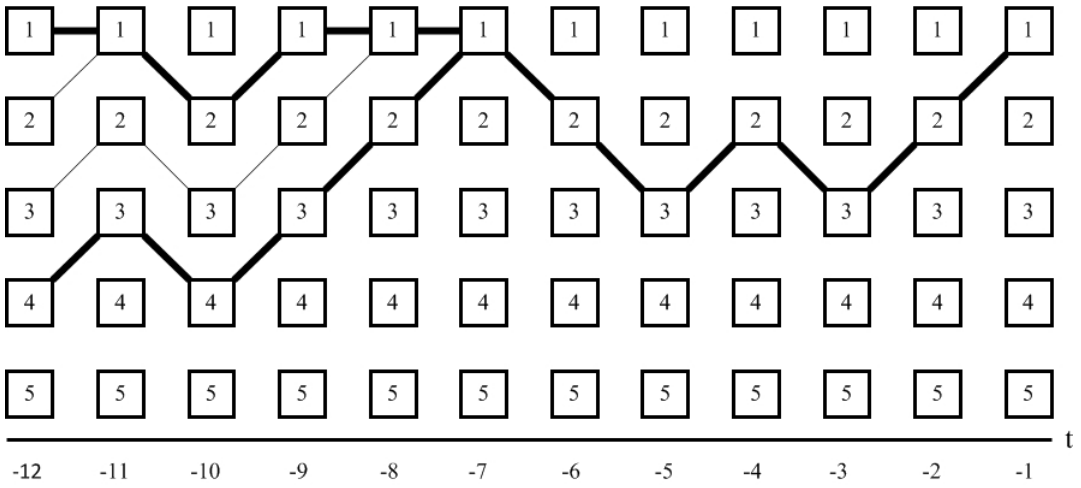
valamint $i = 2,\ldots,(k-1)$ esetén:

$$\phi(i,u) = \begin{cases} i-1 & \text{ha} \quad u \in \left[0, \frac{1}{2}\right) \\ i+1 & \text{ha} \quad u \in \left[\frac{1}{2}, 1\right]. \end{cases}$$

Ha alkalmazzuk a Propp-Wilson algoritmust ezzel az átmenetszabállyal, $k = 6$ lehetséges állapottal, egy jellemző output lehet a 6.4.1. ábrán szereplő illusztráció. Ezen az ábrán szembetűnő, hogy nincsenek „egymást keresztező” átmenetek, azaz ha a $t = i$-edik időpillanatban az egyik állapotból „felfelé” illetve „lefelé” lépünk, akkor ez így lesz az összes többi állapot esetében is, ha azok nem a maximális vagy minimális indexűek. Utóbbi esetben a lánc helyben marad. Ennek oka, hogy a fent definiált átmenetszabály megőrzi az állapottér rendezését: minden $u \in [0,1]$ és minden $i, j \in \{1, \ldots, k\}$ esetén, ha $i \leq j$, akkor

$$\phi(i,u) \leq \phi(j,u).$$

Ez egyszerű következménye annak, hogy minden lánc esetében ugyanazt az u-t használjuk. Ha $u \in [0, \frac{1}{2})$ és $i \leq j$, akkor $i \geq \phi(i,u)$ valamint $j \geq \phi(j,u)$, ezáltal $\phi(i,u) \leq \phi(j,u)$. Az $u \in [\frac{1}{2}, 1]$ eset hasonlóan. Egyenlőség abban az esetben állhat fent, ha az egyik állapot maximális vagy minimális indexű.



6.4.1. ábra: A Propp-Wilson algoritmus egy kimenete $k = 5$ esetben, a fenti átmenetszabállyal. Az egyesüléshez a $t = -12$ időpillanatban kellett indítanunk a láncot, a lineáris rendezés szerinti legnagyobb indexű kiindulási állapot $i = 4$, a legkisebb pedig $j = 1$, az innen induló láncokat jelöli a vastag vonal.

Ebből adódóan azon láncok, melyek valamely $i \in \{2, \ldots, (k-1)\}$ állapotból indulnak, mindig az 1 és k-adik állapotból indulók között maradnak, innen ered az eljárás neve is. Emiatt ha az 1 és k-adik indexű állapotból induló láncok egyesülnek, minden köztes indexűnek is egyesülnie kell velük, ezáltal k darab lánc egyesülésének ellenőrzéséhez elég csak kettőt nyomon követni. A 6.4.1. ábrán illusztrált esetben persze nem jelentene gondot minden lánc nyomon követése, de nagy k esetén ez egy jelentős egyszerűsítés a számítási kapacitás korlátossága miatt. Az ábrán továbbá szembetűnő, hogy az egyesülés a legkisebb indexű állapotban történt meg. Az átmenetszabálynak köszönhetően az egyesülés mindig a maximális vagy minimális indexű állapotban történik. A függelékben található R program segítségével bemutatunk pár szimulációs eredményt erre a feladatra:

Ha $k = 5$ állapotunk, akkor 10^6-on szimuláció elvégzése után a kisorsolásának számát mutatja a következő táblázat:

k: állapot	1	2	3	4	5
darabszám	200233	200395	199836	199717	199819

A számokból jól látszik, hogy az eloszlás közelítőleg egyenletes.

A következő táblázatban különböző k állapotszámokra 50-50 szimuláció eredményeinek átlagait láthatjuk:

k	3	4	5	6	10	20
Bolyongás hosszának átlaga	2.94	5.34	10.5	14.06	40.2	216.98
Egyesülés idejének átlaga	2	3.64	6.04	8.64	27.36	124.2
Egyesülés utáni bolyongási hossz átlaga	0.94	1.7	4.46	5.42	12.84	92.78

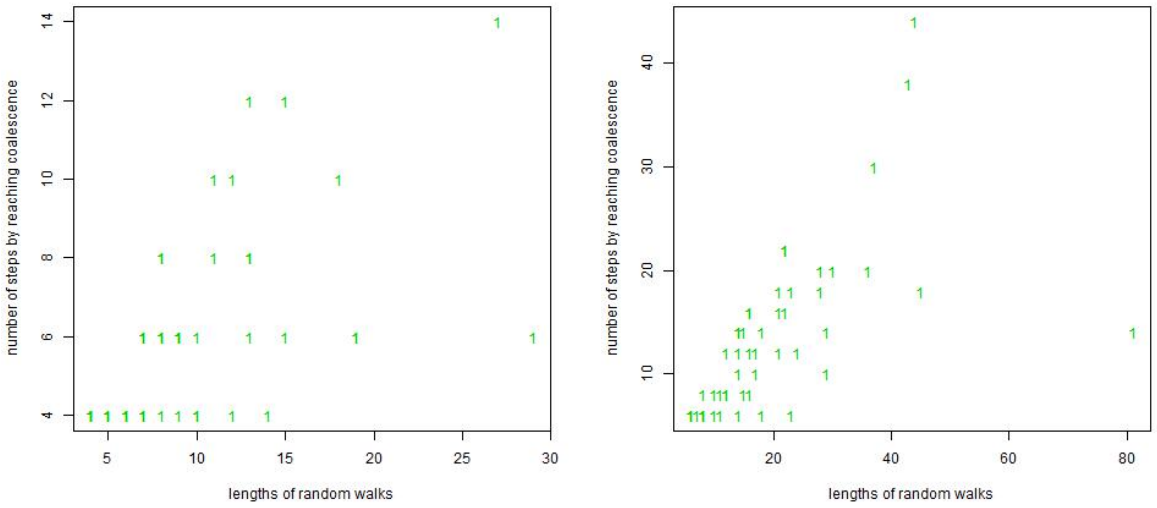
A táblázat eredményei alapján a bolyongás hosszának átlaga nem lineáris függvénye az állapotszámnak. Nyilvánvaló, hogy k állapot esetén legalább $(k-1)$ lépést kell tennünk, hogy a felső és az alsó láncok találkozzanak. A két szám kapcsolatából viszont arra következtethetünk, hogy az algoritmus futási idejének csökkentése céljából érdemes nagy k esetén nagyobb kezdő T értéket választanunk, így elhagyva azokat a lépésszámokat, melyek esetén elhanyagolható a találkozás valószínűsége, csökkentjük a végigszámolt esetek számát.

A 6.1.-es ábra az egyesülési időket mutatja a bolyongások teljes hosszának függvényében. Itt a $k = 4, 7, 10, 20$ állapotszámú eseteket láthatjuk 50-50 szimuláció után. Jól látszik, hogy a bolyongások hossza gyorsabban növekszik, mint az egyesülési idő, ha k-t növeljük.

Az eddigiek alapján a módszer általános leírásához legyen I az állapottér, melyen adott egy $\leq$ részben rendezettség, valamint a ϕ átmenetszabályra teljesüljön a monotonitási feltétel, azaz $x \leq y$ esetén $\phi(x, U_0) \leq \phi(y, U_0)$, minden $x, y \in I$ esetén. Tegyük fel továbbá, hogy az I állapottér x elemeire: $0 \leq x \leq 1$. Legyen

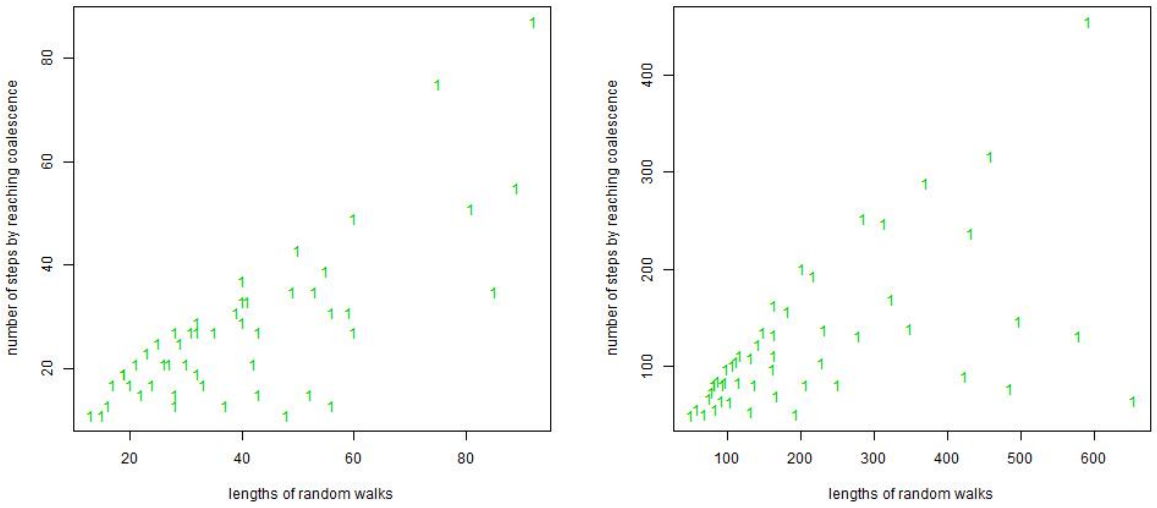
$$\Phi_{t_1}^{t_2}(x, u) = \phi_{t_2-1}(\phi_{t_2-2}(\ldots(\phi_{t_1}(x, u_{t_1}), u_{t_1+1}), \ldots), u_{t_2-2}), u_{t_2-1})$$

ahol $u = (\ldots, u_{-1}, u_0)$. Ha u_{-T},u_{-T+1},...,u_{-2},u_{-1}-re teljesül, hogy $\Phi_{-T}^0(0, u) = \Phi_{-T}^0(1, u)$, akkor a monotonitási feltétel biztosítja, hogy $\Phi_{-T}^0(x, u) = \Phi_{-T}^0(y, u)$, minden $x, y \in I$ esetén. Ezáltal a legkisebb T, melyre $\Phi_{-T}^0(\cdot, u)$ konstans, egyenlő a legkisebb T-vel, melyre $\Phi_{-T}^0(0, u) = \Phi_{-T}^0(1, u)$. Jelöljük ezt a T-t T^*-gal. Próbáljuk egymás után a T = 1,2,4,8,... értékeket, amíg nem találunk egy k-t, melyre $T = 2^k$ esetén $\Phi_{-T}^0(0, u) = \Phi_{-T}^0(1, u)$. Ezáltal a szimuláció lépésszáma $2(1 + 2 + 4 + \ldots + 2^k) < 2^{k+2}$, ahol az egyenlőtlenség elején szereplő 2-es szorzó a két lánc párhuzamos futtatása miatt jelenik meg (az egyiket a 0-ból, a másikat az 1-ből indítjuk). Ez közelítőleg optimális, ugyanis T^*-nak legalább 2^{k-1}-nek kell lennie, különben nem kellett volna $T =$



(a) k=4 állapot

(b) k=7



(c) k=10 állapot

(d) k=20 állapot

6.1. ábra. Az egyesülés ideje a bolyongások hosszának függvényében

2^k lépést tennünk, ezáltal a szükséges szimulációs lépések száma, hogy megmutassuk $\Phi^0_{-T^*}(0,u) = \Phi^0_{-T^*}(1,u)$ nagyobb mint $2 \cdot 2^{k-1} = 2^k$.

Az eljárás összegzése algoritmikus formában:

T $\leftarrow 1$

REPEAT

$upper \leftarrow 1$

$lower \leftarrow 0$

FOR $t = -T$ to -1

$upper \leftarrow \phi_t(upper, u_t)$

$lower \leftarrow \phi_t(lower, u_t)$

$T \leftarrow 2T$

UNTIL $upper = lower$

RETURN *upper*

6.2. Megjegyzés. *Mint a korábbi, általános CFTP esetében, itt is fontos, hogy ugyanazokat az u-kat használjuk, és ne generáljunk a REPEAT során újakat.*

Mint láthattuk, a sandwiching módszer a gyakorlatban jól alkalmazhatóvá teszi a CFTP algoritmust, de sajnos olyan Markov láncok eseten nem alkalmazható, melyek átmenetszabályára nem teljeseül a monotonitási feltétel, vagy nem adható meg egy részben rendezettség maximális és minimális állapotokkal az állapottéren.

7. fejezet

Összefoglalás

A dolgozat elkészítésével az volt a célom, hogy megismerjem az addig számomra kevéssé ismert Markov láncokat és az azokat felhasználó MCMC módszereket. Az első fejezetben bemutatásra kerültek a Markov láncok, mely jórészt az [1]-en alapult. Ezután az algoritmus család időbeli fejlődését követve bemutattam a talán leginkább elterjedet eljárásokat, kezdve a rengeteg helyen alkalmazott Metropolis-Hastings algoritmussal és a Gibbs mintavételezővel. Ezen fejezetek alapját az [5],[6],[7] szolgáltak. Némi kitekintésektől eltekintve, a dolgozat kereteinek fényében szinte mindenhol diszkrét eseteket tárgyaltam. Ezt követte a Slice sampling általános eljárásának bemutatása illetve a megvalósítás különböző lehetőségeinek tárgyalása. A fejezet alapját a [16] képezi. Végül pedig bemutatásra került a Perfect sampling ötlete, egy általánosan használható eljárása és egy, a gyakorlati alkalmazást is lehető tévő kiegészítése, a sandwiching módszer. Ezt a módszert egy megértést elősegítő példán keresztül mutattam be, illetve a Perfect Samplingre jellemző tulajdonságok vizsgálatára egy R programot is készítettem. Ez a program kiterjeszthető további eloszlások mintavételezésére is. A fejezet alapját a [9],[10],[12] és [14] adta. A Perfect samplinggel kapcsolatban még manapság is számos nyitott probléma van ezért sok kutatás tárgyát képezi.

8. fejezet

Függelék

8.1. R program a sandwiching módszer bemutatásához

Az alábbi programkód a dolgozatban szereplő eredmények kiszámítását végzik el. Ha azok csak közül 1-1-re vagyunk kíváncsiak, a megfelelő részeket kommentként átírhatjuk. A sandw függvény for ciklusba ágyazása miatt, ha csupán az egészet bemásoljuk az R-be, nem jelenik meg a DF és a DF2 adatsor, mely az állapotok változásának folyamatát írja ki a konzolra.

```
setwd('C:/R')

### Creating a uniform distribution on {0,1}
distr <- function() {
randomNumber <- runif(1, min=0, max=1)

if (randomNumber >= 0.5)
randomNumber = 1
else if (randomNumber < 0.5)
randomNumber = -1
}

nb <- 50 # set the number of simulations
```

```
c <- c(1:nb)
d <- c(1:nb)
e <- c(1:nb)
for (i in 1:nb)

{

upd <- distr() # set upd's beginning value
k <- 20 # set k as the number of possible states

### Building Sandwiching function
sandw <- function (upd) {
x <- seq(1:k) # index vector
mtx <- matrix(0,k,1) # empty state matrix
DF = data.frame(x)

mav <- k # state with maximal index
miv <- 1 # state with minimal index

y <- seq(1:k)
mtx2 <- matrix(0,k,1)
DF2 = data.frame(y)

ma <- k
mi <- 1

for (i in 1:length(upd))
{
mtx[mav,1] <- 0 # set previous element with index mav to 0
mtx[miv,1] <- 0 # set previous element with index miv to 0

mav <- min(max(mav+upd[i],1),k) # update mav
miv <- min(max(miv+upd[i],1),k) # update miv
```

```
mtx[mav,1] <- 1 # update the vector element
mtx[miv,1] <- 1 # update the vector element

DF <- data.frame(mtx,DF)

if (ma!=mi) # do until reach the state of coalescence
{
mtx2[ma,1] <- 0
mtx2[mi,1] <- 0

ma <- min(max(ma+upd[i],1),k)
mi <- min(max(mi+upd[i],1),k)

mtx2[ma,1] <- 1
mtx2[mi,1] <- 1

DF2 <- data.frame(mtx2,DF2)
}
}
if (mav!=miv) return(NA)
else list(mav,DF,DF2,ncol(DF)-1,ncol(DF2)-1)
# print random walks and numbers of them
}

while (is.na(sandw(upd))) upd <- c(distr(), upd)
# do until reaching the state of coalescence

sandw(upd)  # listing
# create a vector with maximum values of the simulations
c[i] <- sandw(upd)[1]
# create a vector with total lengths of random walks
d[i] <- sandw(upd)[4]
# create a vector with number of steps by reaching coalescence
e[i] <- sandw(upd)[5]
```

```
}

d <- unlist(d)
e <- unlist(e)

table(unlist(c)) # summary of draws

tb <- data.frame(d,e,d-e)

mean1 <- sum(tb[1])/nb # mean: total lengths of random walks
mean2 <- sum(tb[2])/nb # mean: number of steps by reaching coalescence
mean3 <- sum(tb[3])/nb # mean: number of steps after reaching coalescence

mean1; mean2; mean3

jpeg(filename="plot.jpg")
matplot(d,e,type="p",lwd=2,col=3,xlab="lengths of random walks",ylab=
"number of steps by reaching coalescence")
dev.off()
```

Irodalomjegyzék

[1] http://www.stat.ufl.edu/ casella/Papers/MCMCHistory.pdf

[2] http://www.cs.elte.hu/ villo/ml/ML.pdf

[3] I. M. Szobol: A Monte-Carlo mószerek alapjai *Műszaki Könyvkiadó*, Budapest, 1981

[4] Gilks W.R., Richardson S., Spiegelhalter D.J.: Markov Chain Monte Carlo in Practice *Chapman & Hall/CRC*, 1996.

[5] http://web.mit.edu/ wingated/www/introductions/mcmc-gibbs-intro.pdf

[6] Siddhartha Chib, Edward Greenberg: Understanding the Metropolis-Hastings Algorithm. In: *The American Statistician*, Vol. 49, No. 4. (Nov., 1995), pp. 327-335

[7] George Casella, Edward I. George: Explaining the Gibbs Sampler. In: *The American Statistician*, Vol. 46, No. 3 (Aug., 1992), pp. 167-174.

[8] http://www.stat.lsa.umich.edu/ kshedden/Courses/Stat606/Notes/mcmc.pdf

[9] George Casella, Michael Lavine, Christian P. Robert: Explaining the Perfect Sampler. In: *The American Statistician*, Vol. 55, No. 4. (Nov., 2001), pp. 299-305.

[10] Olle Haggstrom: Finite Markov Chains and Algorithmic Applications *Cambridge University Press*, 2002.

[11] David B. Wilson: Exact Sampling with Markov Chains *Massachusetts Institute of Technology*, 1996.

[12] Wilfrid S. Kendall: Notes on Perfect Simulation *University of Warwick*, 2004.

[13] Solymosi Norbert: Bevezetés az R-nyelv és környezet használatába, 2005.

[14] James J. Propp, David B. Wilson: Exact Sampling with Coupled Markov Chains and Applications to Statistical Mechanics. *Massachusetts Institute of Technology*, 1996.

[15] László Lovász, Peter Winkler: Exact Mixing in an Unknown Markov Chain. *The Electronic Journal of Combinatorics*, Vol. 2, R15 (1995)

[16] Radford M. Neal: Slice sampling. In: *The Annals of Statistics*, Vol. 31, No. 3 (Jun., 2003), pp. 705-741.

9. fejezet

Köszönetnyilvánítás

Ezúton szeretném megköszönni témavezetőmnek, Pröhle Tamásnak a dolgozat elkészítésében nyújtott segítségét, folyamatos útmutatását és hogy bevezetett az R program használatába. Köszönöm évfolyamtársamnak, Csillagvári Dánielnek az ábrák elkészítésében nyújott segítségét.

Nyilatkozat

A szakdolgozat szerzőjeként fegyelmi felelősségem tudatában kijelentem, hogy a dolgozatom önálló munkám eredménye, saját szellemi termékem, abban a hivatkozások és idézések standard szabályait következetesen alkalmaztam, mások által írt részeket a megfelelő idézés nélkül nem használtam fel.