Domača naloga 5

Neža Kržan, Tom Rupnik Medjedovič

1 Cilj naloge

Želiva preučiti Metropolis-Hastingsov algoritem pri vzorčenju iz dvorasežne porazdelitve, katere poznava funkcijo gostote. Gre za nestandardno porazdelitev(kot so npr. Beta, Gamma, normalna,...), zato je vzorčenje z običajnimi metodami nemogoče. V ta namen bova torej uporabila Metropolis-Hastings, s pomočjo katerega bova vzorčila iz dane porazdelitve (z uporabo gostote). V najnem primeru je potrebno generirati koordinate točk, torej pare (x_i, y_i).

Ocenila bova verjetnost, da sta oba parametra manjša od $1(P(x < 1 \text{ in } y < 1))$ in analizirala porazdelitev te verjetnosti. Želiva preučiti ali tudi z manjšimi vzorci (velikosti 100) dovolj dobro opišemo dano porazdelitev, zato bova izračunala pokritost 95% intervala zaupanja za to verjetnost.

2 Generiranje vrednosti

Z uporabo algoritma Metropolis-Hastings bova generirala vrednosti iz porazdelitve, ki ima gostoto proporcionalno

$$f(x,y) = \begin{cases} x^2y^2e^{-x}e^{-y}e^{-xy}, \text{ kjer } x > 0 \text{ in } y > 0 \\ 0, \text{ sicer} \end{cases}$$

in večina vrednosti manjših od 5.

Algoritem je sestavljen iz naslednjih korakov:
Na začetku si izberemo neki začetni vrednosti x_0 in y_0, za kateri mora veljati, da je gostota večja od 0 (je možen izid). Nato na vsakem koraku s pomočjo gostote porazdelitve $g(X_p|X_i = x_i)$ predlagamo novo vrednost (x_p) pogojno na predhodnjo vrednost (x_i). Vendar pa še ne vemo ali predlagano vrednost (x_p) zares sprejmemo. Zato izračunamo verjetnost $\alpha = min\left(\frac{f(x_p)g(x_i|x_p)}{f(x_i)g(x_p|x_i)}, 1\right)$, ki nam pove verjetnost sprejema nove vrednosti, $(1-\alpha)$ pa verjetnost za ohranitev predhodnje na sledeč način

$$x_{i+1} = \begin{cases} x_p, \text{ z verjetnostjo } \alpha \\ x_i, \text{ z verjetnostjo } 1-\alpha \end{cases} .$$

Za predlaganje novih vrednosti sva si izbrala

$$x_p = N(x_i, 1),$$
$$y_p = N(y_i, 1),$$

torej normalno porazdelitev ter za gostoto porazdelitve

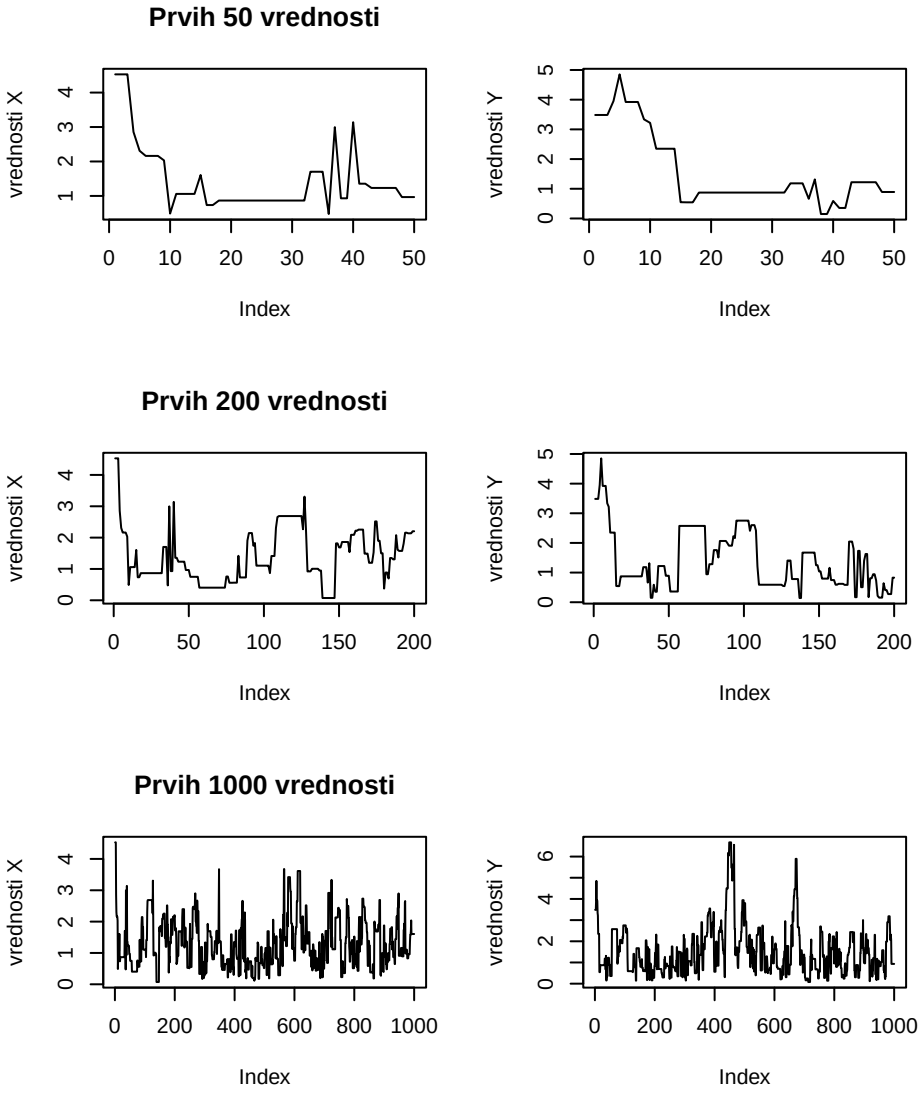
$$g((x_p, y_p)|(x_i, y_i)) = f_{N(x_i,1)}(x_p) \cdot f_{N(y_i,1)}(y_p),$$

kjer sta $f_{N(x_i,1)}$ in $f_{N(x_i,1)}$ gostoti $X_i = x_i$ in $Y_i = y_i$, s povrečjema x_i in y_i(z x_i in y_i, ki ju predlagamo) in s standardnim odklonom 1.

Velikost vzorca, ki ga generiramo, pa je enaka 10000(`n`).

Pri generiranju podatkov moramo določiti še začetno vrednost (izbrala sva $(x_0, y_0) = (4, 4)$, ker je večina vrednosti manjših od 5 in izbereva nekoliko manj) in vrednosti parametrov `burn in` in `step`.

Parameter `burn in` nam določi koliko začetnih vrednosti izpustimo iz vzorca (jih ne vključimo). Vrednost tega sva po prvem tesu algoritma določila, na podlagi spodnjih grafov, na 100. Oglejmo si gibanje vrednosti za prvih 50, 200 in 1000 vrednosti iz vzorca, kjer smo imeli `burn in = 1` in `step = 1`.

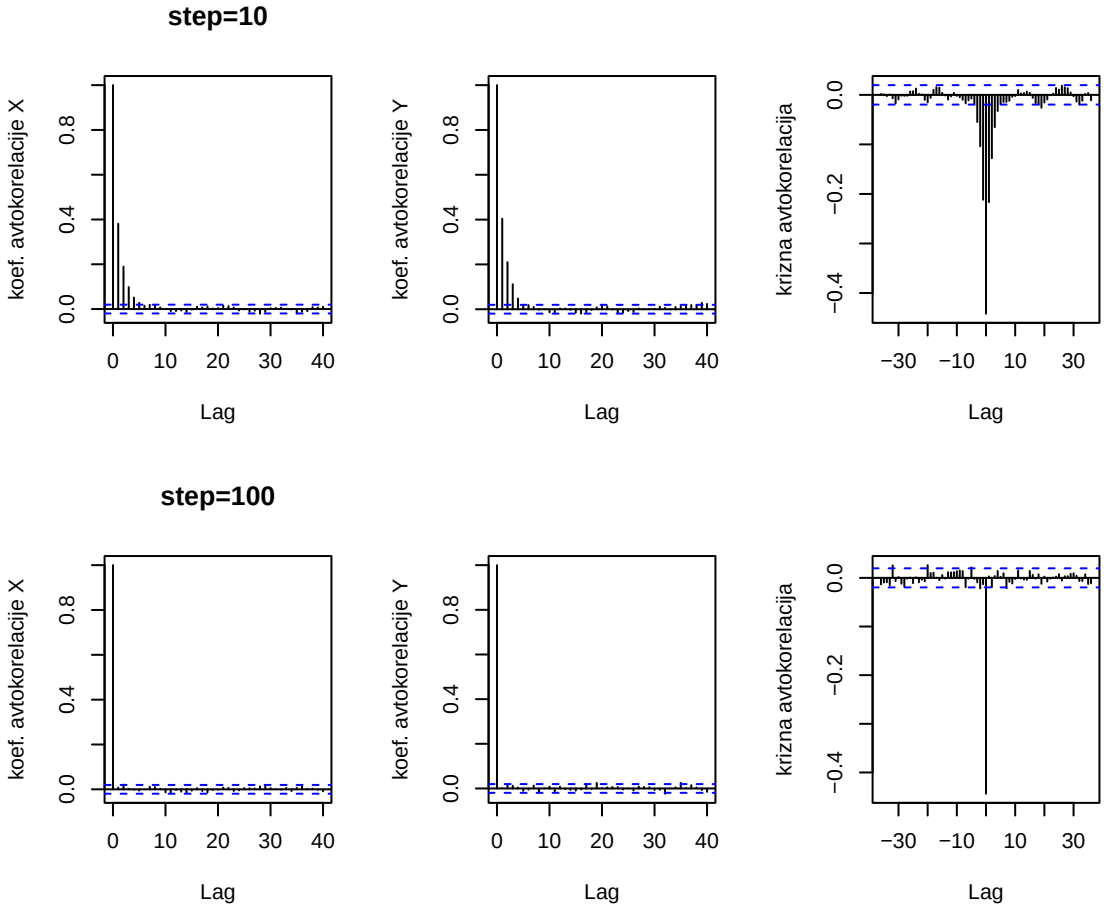


Slika 1: Graf gibanja prvih 150, 200 in 1000 vrednosti za X in Y.

Res lahko vidimo, da se vrednosti gibljejo znotraj pričakovanega območja (večina vrednosti je na intervalu $(0, 5)$). Pravzaprav nas v tem primeru zanima bolj ali so se vrednosti že ustalile oz. če v začetnih vrednostih ni drastičnega naraščanja ali padanja. Na prvih dveh grafih vidimo, da se vrednosti X in Y niso še ustalile(*prvih 50 vrednosti*), torej bi bilo potrebno nekaj prvih členov izpustiti - za to bova torej uporabila parameter `burn in = 100`.

Vidimo tudi, da vrednosti X in Y nekaj časa "ostaneta" v enakih vrednostih predno se "premakneta" naprej, torej morava izločiti avtokorelacijo med zaporednimi elementi v vzorcu. To nam določi parameter `step` - koliko zaporednih vrednosti ne vključimo v vzorec. S

Na spodnjih avtokorelogramih lahko vidimo, da če imamo v algoritmu `step = 10` in `burn in = 100`, je v vzorcu prisotno še kar nekaj avokorelacije, če pa nastavimo `step = 100` pa vidimo, da noben koeficient avtokorelacije ne sega iz 95% intervala zaupanja na avtokorelogramu(vodoravni modri črti). Tudi na 3. grafu pri `step = 10` vidimo negativno križno korelacijo - koef. so statistično pomembni, torej sta X in Y obratno povezana, kar pa nimamo več pri `step = 100`.

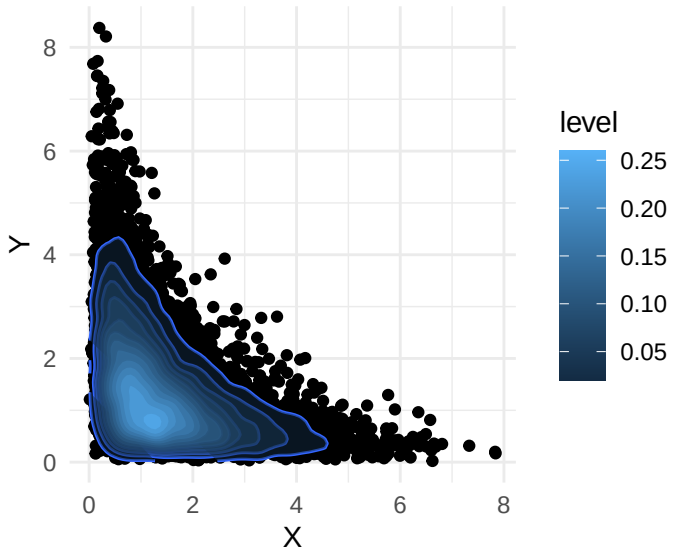


Slika 2: Graf avtokorelacije in križne avtokorelacije za X in Y pri step=10 in step=100.

Z izbrano vrednostjo parametra `step=100` sva torej odstranila avtokorelacijo med zaporednimi členi vzorca.

2.1 Prikaz vrednosti vzorca

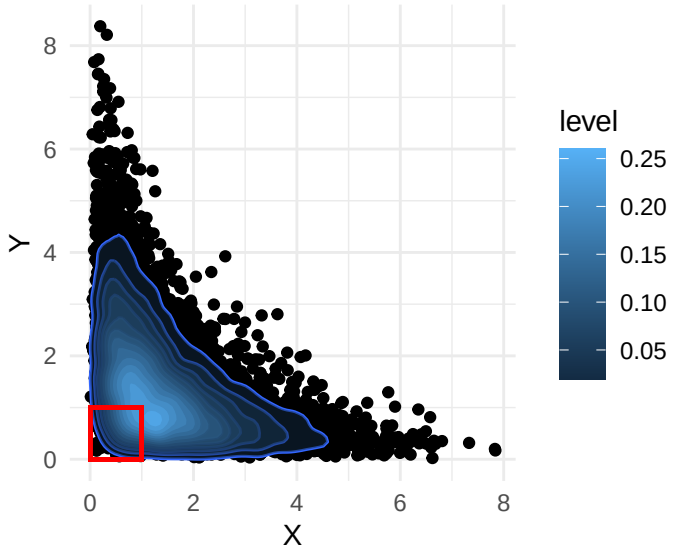
Na spodnjem grafu lahko vidimo, da je večina vrednosti x_i in y_i, $i = 1, \ldots, 10000$ zgoščenih na intervalu $[0, 3]$. To je tudi nekako pričakovano, saj gostota porazdelitve nekoliko spominja na gostoto eksponentne porazdelitve(kjer so točke bolj koncentrirane bližje začetku), in so temu primerno razporejene tudi točke.



Slika 3: Prikaz vrednosti iz porazdelitve za vzorec velikosti 10000.

3 Verjetnost $P(x < 1$ **in** $y < 1)$

Želimo oceniti verjetnost, da sta obe vrednosti (X in Y) manjši od 1. Na dovolj velikem vzorcu lahko to vrednost ocenimo kot delež točk, ki se nahajajo znotraj območja $(0, 1) \times (0, 1)$(primer območja lahko vidimo na spodnjem grafu). Ker je generiranje velikih vzorcev časovno zahtevni proces, sva zgenerirala vzorec velikosti 1000000. Ta vzorec bova uporabila kot "populacijo" iz katere bova izbrala naključne vrednosti in ustvarila manjše vzorce za potrebe simulacij.



Slika 4: Vrednosti za vzorec velikosti 10000 z označenim območjem (0,1)x(0,1).

Na tem (velikem) vzorcu, sva izračunala željeni delež (obe vrednosti sta manjši od 1) in dobila vrednost 0.091. Ker se nam ta vrednost zdi dokaj majhna glede na zgornji graf vrednosti, si oglejmo naslednjo tabelo. Vidimo, da je res velik delež vrednosti, ko je vsaj ena od vrednosti (x_i ali y_i) višja od 1.

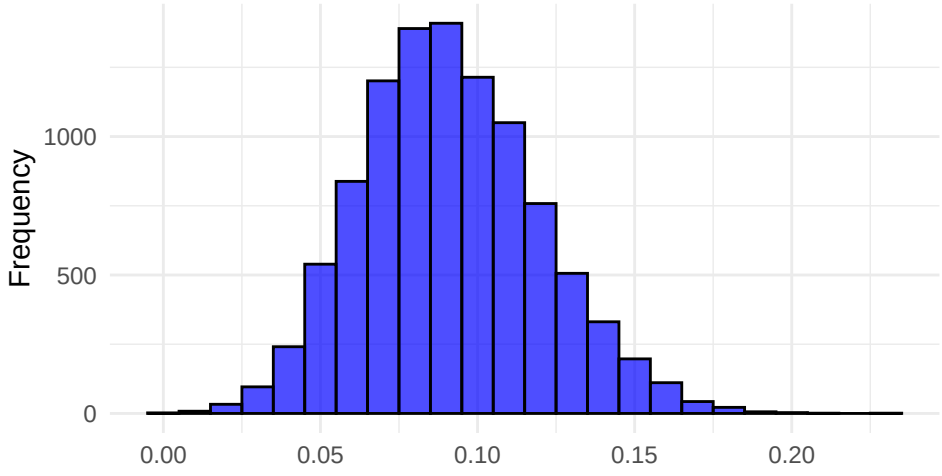
Tabela 1: Tabela razporeditev vrednosti glede na X in Y v populaciji.

	$Y < 1$	$1 \leq Y < 2$	$2 \leq Y < 3$	$3 \leq Y$
$X < 1$	9.15%	16.22%	9.44%	6.28%
$1 \leq X < 2$	16.22%	15.55%	4.37%	1.06%
$2 \leq X < 3$	9.49%	4.36%	0.48%	0.04%
$3 \leq X$	6.26%	1.04%	0.04%	0%

3.1 Vzorci velikosti 100

Poglejmo si kakšna je porazdelitev zgornjega deleža v primeru, da imamo vzorce velikosti le 100. Generiramo veliko število vzorcev (npr. 10000) velikosti 100, na vsakem izračunamo verjetnost da sta tako x_i kot y_i manjša od 1 in narišemo histogram.

Vidimo lahko, da se vrednosti porazdeljujejo zelo podobno normalni porazdelitvi, ampak asimetrično v levo, s parametroma μ =0.0913 in σ =0.0284.



Slika 5: Histogram porazdelitve verjetnosti P(X<1 in Y<1).

3.1.1 Pokritost

Izračunajmo še pokritost 95% intervala zaupanja za to vrednost. Interval zaupanja za povprečje porazdelitve oziroma verjetnosti, da sta tako X kot Y manjša od 1, bomo izračunali po formuli za intervale zaupanja iz funkcije `prop.test()`, in sicer ta uporablja asimptotski Waldov test za izračun intervalov zaupanja za deleže. Interval zaupanja temelji na normalni aproksimaciji binomske porazdelitve, torej je metoda zanesljiva pri velikih vzorcih.

Vendar pa nastopi problem, saj ne poznamo "prave" vrednosti verjetnosti, saj ne poznamo parametrov porazdelitve oziroma vrednosti populacije. To lahko rešimo tako, da za "pravo" vrednost vzamemo verjetnosti delež izračunan na velikem vzorcu (v našem primeru ima 1000000 enot).

Prava vrednost deleža je enaka 0.09148.

Izvedla sva simulacijo s 10000 ponovitvami, kjer sva iz populacije vsakič izbrala nov vzorec velikosti 1000, na njem s pomočjo funkcije prop.test() izračunala interval zaupanja za verjetnost in pogledala ali vrednost deleža na populaciji pripada intervalu zaupanja. Po izvedeni simulaciji je vrednost pokritja 95% intervala zaupanja enaka 0.9511.

4 Zaključek

Z algoritmom Metropolis-Hastings lahko dobro generiramo vrednosti iz pogojne porazdelitve. Za izračun potrebujemo le gostoto želene porazdelitve. Pri tem moramo paziti le na pravilno izbiro začenih vrednosti (v najnem primeru (x_0, y_0)) in parametrov `burn in` ter `step`. Tudi v primeru manjših vzorecev (velikosti 100) dovolj dobro opišemo dano porazdelitev. To sva dodatno preverila s simulacijami, kjer smo ocenjevali verjetnost, da sta obe vrednosti manjši od 1. Pokritost 95% intervala je bila zelo blizu željeni vrednosti (izračunana vrednost je enaka 0.9511), torej smo z izidom zadovoljni.