IPB University
Inspiring Innovation with Integrity

Pengantar Statistika Bayes - STA1312

Metode Komputasi Bayesian - Part 1

(MCMC, Algoritma Metropolis–Hastings, dan Gibbs Sampling)

Dr. Kusman Sadik, S.Si, M.Si
Program Studi Statistika dan Sains Data IPB
Tahun Akademik 2023/2024

Markov Chain Monte Carlo

Konsep Dasar Markov Chain Monte Carlo

- Pengembangan metode Markov Chain Monte Carlo (**MCMC**) telah menjadi hal yang sangat penting untuk statistika Bayes.
- Metode ini memungkinkan kita untuk **menarik sampel** dari sebaran posterior $g(\theta|y)$ yang eksak, berdasarkan bentuk proporsionalnya, yaitu $g(\theta).f(y|\theta)$ yang berupa perkalian prior dengan likelihood.
- Melalui MCMC, inferensi Bayesian didasarkan pada **sampel dari posterior** ini, bukan pada posterior yang eksak.
- Metode MCMC ini dapat pula digunakan untuk model yang rumit yang memiliki banyak parameter.
- Dua metode utama pada MCMC adalah: (i).Algoritma **Metropolis–Hastings**, dan (ii).Algoritma **Gibbs Sampling**.

Markov Chain

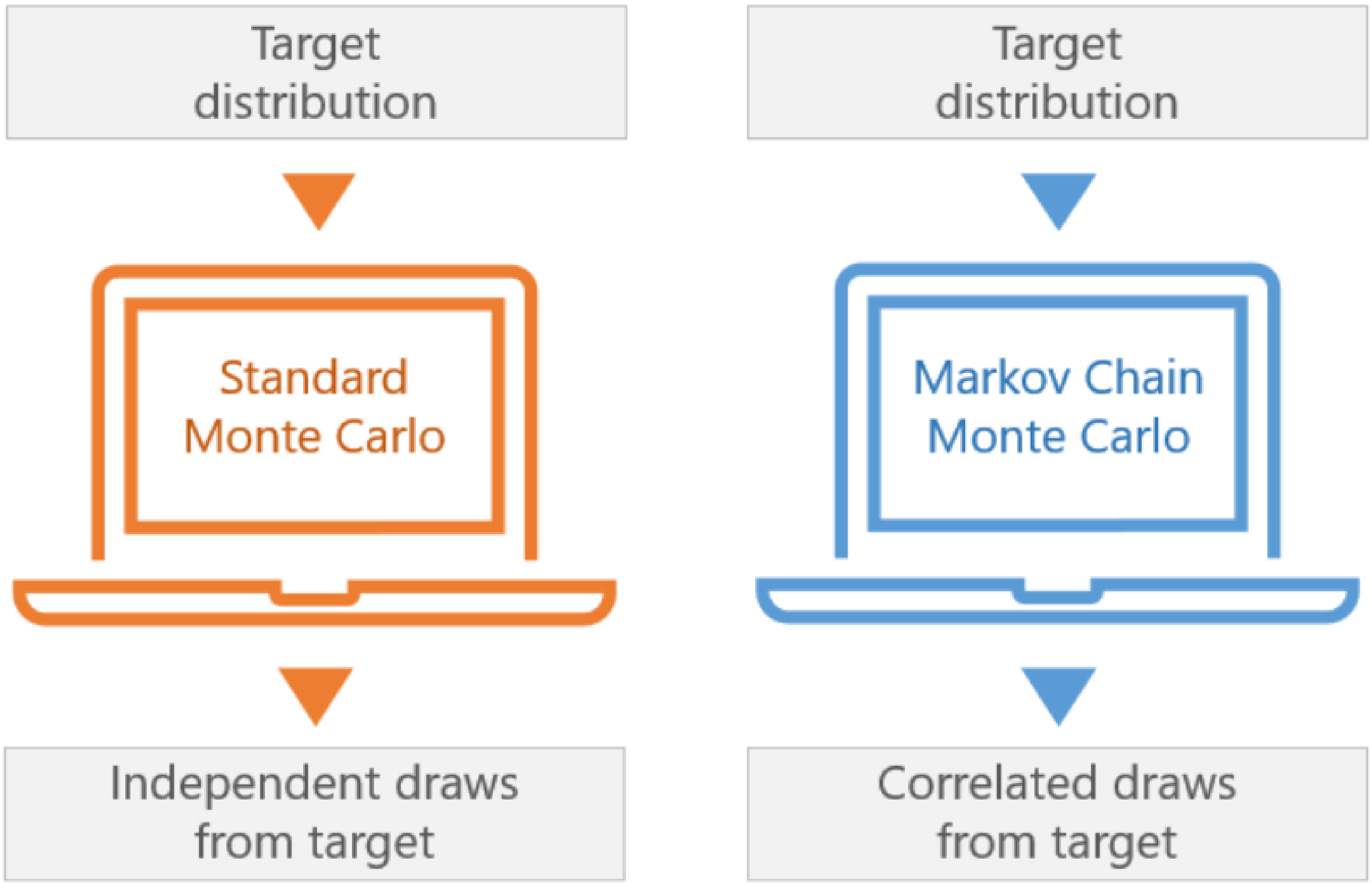
- Rantai Markov (Markov Chain) adalah model dari proses yang bergerak di sekitar serangkaian kemungkinan nilai yang disebut sebagai keadaan (***state***).
- Status keadaan masa depan (***future state***) akan dipilih secara acak menggunakan beberapa peluang transisi.
- Rantai Markov memiliki karakteristik "tanpa-memori“ (***memoryless***), artinya keadaan masa depan (*future state*) hanya bergantung pada keadaan saat ini (*current state*), tidak tergantung pada keadaan masa lalu (*past states*). Hal ini disebut sebagai **properti Markov**.
- Berdasarkan properti tersebut maka peluang transisi dari rantai Markov hanya akan bergantung pada keadaan saat ini, bukan pada keadaan sebelumnya.

$$P(X_{t+1} \mid X_t, X_{t-1}, X_{t-2}, \dots, X_2, X_1) = P(X_{t+1} \mid X_t)$$

Markov Chain (*cont.*)

- Setiap keadaan (*state*) hanya terhubung langsung ke *state* sebelumnya dan tidak ke *state* lain yang lebih jauh ke belakang. Melalui proses ini maka hubungan antar *state* bersifat seperti **rantai**.
- Himpunan dari berbagai *state* disebut sebagai ruang *state* (***state-space***) yang dapat berupa diskrit maupun kontinu.
- Pada perkuliahan ini hanya menggunakan rantai Markov yang peluang transisinya tetap sama di setiap state yang disebut sebagai ***time-invariant***.
- Pada Markov Chain Monte Carlo (MCMC), kita perlu menemukan rantai Markov dengan sebaran jangka panjang (***long-run distribution***) yang sama dengan sebaran posterior $g(\theta|y)$.

Markov Chain (*cont.*)



Algoritma Metropolis–Hastings

Algoritma Metropolis–Hastings

- Alagoritma Metropolis–Hastings bertujuan untuk mengambil sampel dari beberapa **sebaran target** (*target density*) dengan memilih nilai dari sebaran kandidat (*candidate density*).
- Pemilihan untuk menerima suatu nilai kandidat, terkadang disebut **proposal**, hanya bergantung pada nilai yang diterima sebelumnya.
- Algoritma ini membutuhkan nilai awal (*initial value*) untuk memulai, peluang penerimaan (*acceptance*), dan peluang transisi.
- Apabila peluang transisi bersifat **simetris**, maka urutan nilai yang dihasilkan dari proses ini membentuk rantai Markov.
- Secara simetris artinya bahwa peluang bergerak dari state θ ke state θ’ **sama** dengan peluang bergerak dari state θ’ ke state θ.

Algoritma Metropolis–Hastings (*cont.*)

- Jika g(θ|y) adalah sebaran posterior (sebaran target) yang tidak diskalakan, dan q(θ, θ’) adalah sebaran kandidat, maka **peluang transisi** ditentukan sebagai berikut:

$$\alpha(\theta,\theta') = \min\left[1, \frac{g(\theta'|y)q(\theta',\theta)}{g(\theta|y)q(\theta,\theta')}\right]$$

Algoritma Metropolis–Hastings (*cont.*)

Langkah-langkah algoritma Metropolis–Hastings dapat diringkas sebagai berikut:

1. Start at an initial value $\theta^{(0)}$.
2. Do the following for $n = 1, \ldots, n$.
   (a) Draw θ' from $q(\theta^{(n-1)}, \theta')$.
   (b) Calculate the probability $\alpha(\theta^{(n-1)}, \theta')$.
   (c) Draw u from $U(0, 1)$.
   (d) If $u < \alpha(\theta^{(n-1)}, \theta')$, then let $\theta^{(n)} = \theta'$, else let $\theta^{(n)} = \theta^{(n-1)}$

Catatan: $\alpha(\theta, \theta') = \min \left[1, \dfrac{g(\theta'|y)q(\theta', \theta)}{g(\theta|y)q(\theta, \theta')} \right]$

Algoritma Metropolis–Hastings (*cont.*)

1. Start at an initial value $\theta^{(0)}$.
2. Do the following for $n = 1, \ldots, n$.
   (a) Draw θ' from $q(\theta^{(n-1)}, \theta')$.
   (b) Calculate the probability $\alpha(\theta^{(n-1)}, \theta')$.
   (c) Draw u from $U(0,1)$.
   (d) If $u < \alpha(\theta^{(n-1)}, \theta')$, then let $\theta^{(n)} = \theta'$, else let $\theta^{(n)} = \theta^{(n-1)}$

We should note that having the candidate density $q(\theta, \theta')$ close to the target $g(\theta|y)$ leads to more candidates being accepted. In fact, when the candidate density is exactly the same shape as the target

$$q(\theta, \theta') = k \times g(\theta'|y)$$

the acceptance probability is given by

$$\begin{aligned}\alpha(\theta, \theta') &= \min\left[1, \frac{g(\theta'|y)\, q(\theta', \theta)}{g(\theta|y)\, q(\theta, \theta')}\right] \\ &= \min\left[1, \frac{g(\theta'|y) g(\theta|y)}{g(\theta|y) g(\theta'|y)}\right] \\ &= 1\,.\end{aligned}$$

Algoritma Gibbs Sampling

Algoritma Gibbs Sampling

- Algoritma Gibbs-Sampling sangat relevan ketika terdapat **banyak parameter** dalam model.
- Algoritma **Metropolis-Hastings** dapat diperluas ke masalah dengan banyak parameter tersebut.
- Namun, seiring bertambahnya jumlah parameter, **performa** algoritma untuk *acceptance rate*-nya umumnya menurun.
- *Acceptance rate* dapat ditingkatkan di Metropolis–Hastings dengan hanya memperbarui satu blok parameter di setiap iterasi. Hal ini kemudian dikenal dengan algoritma ***blokwise***-Metropolis-Hastings.
- Algoritma **Gibbs-Sampling** merupakan **kasus khusus** dari algoritma *blokwise*-Metropolis-Hastings tersebut.

Algoritma Gibbs Sampling (*cont.*)

- Pada penerapannya, algoritma Gibbs-Sampling sangat berguna untuk model berhirarki (***hierarchical model***), karena ketergantungan antar parameter model terdefinisi dengan baik.
- Gibbs-Sampling merupakan algoritma simulasi rantai Markov dari sebaran posterior bersama (***joint posterior distribution***) dengan cara mensimulasikan tiap parameter secara individual (***univariate***) dari rangkaian sebaran bersyarat.
- Mensimulasikan satu nilai dari setiap parameter ini disebut sebagai satu siklus (***one cycle***) dari Gibbs-Sampling.
- Secara umum, performa algoritma simulasi Gibbs-Sampling akan **konvergen** pada sebaran target, yakni ***joint posterior distribution***.

Algoritma Gibbs Sampling (*cont.*)

Suppose we decide to use the true conditional density as the candidate density at each step for every parameter given all of the others. In that case

$$q(\theta_j, \theta_j' | \boldsymbol{\theta}_{-j}) = g(\theta_j | \boldsymbol{\theta}_{-j}, \mathbf{y}) \,,$$

where $\boldsymbol{\theta}_{-j}$ is the set of all the parameters excluding the j^{th} parameters. Therefore, the acceptance probability for θ_j at the n^{th} step will be

$$\begin{aligned} \alpha\left(\theta_j^{(n-1)}, \theta_j' | \boldsymbol{\theta}_{-j}^{(n)}\right) &= \min\left[1, \frac{g(\theta_j' | \boldsymbol{\theta}_{-j}, \mathbf{y})\, q(\theta_j', \theta_j | \boldsymbol{\theta}_{-j})}{g(\theta_j | \boldsymbol{\theta}_{-j}, \mathbf{y})\, q(\theta_j, \theta_{-j} | \boldsymbol{\theta}_{-j})}\right] \\ &= 1\,. \end{aligned}$$

so the candidate will be accepted at each step. The case where we draw each candidate block from its true conditional density given all the other blocks at their most recently drawn values is known as Gibbs sampling.

Implementasi MCMC dalam Program R

Studi Kasus 1:

- Misalkan diketahui **sebaran posterior** bagi theta adalah g(θ|y) = $e^{-\theta}$, dengan θ > 0.
- Lakukan **sampling** bagi θ berdasarkan sebaran posterior tsb (N = 10000).
- Pada metode MCMC, sebaran posterior dinyatakan sebagai sebaran target.
- Secara **analitik** tentukan penduga Bayes bagi θ. Bandingkan dengan hasil yang diperoleh melalui MCMC. Apa kesimpulannya?

```
> #Metode MCMC dengan Algoritma Metropolis-Hastings
>
> par(mar = c(3, 3, 3, 3))
>
> target <- function(s){
+   if (s < 0) {
+     return(0)
+   } else {
+     return(exp(-s))
+   }
+ }
```

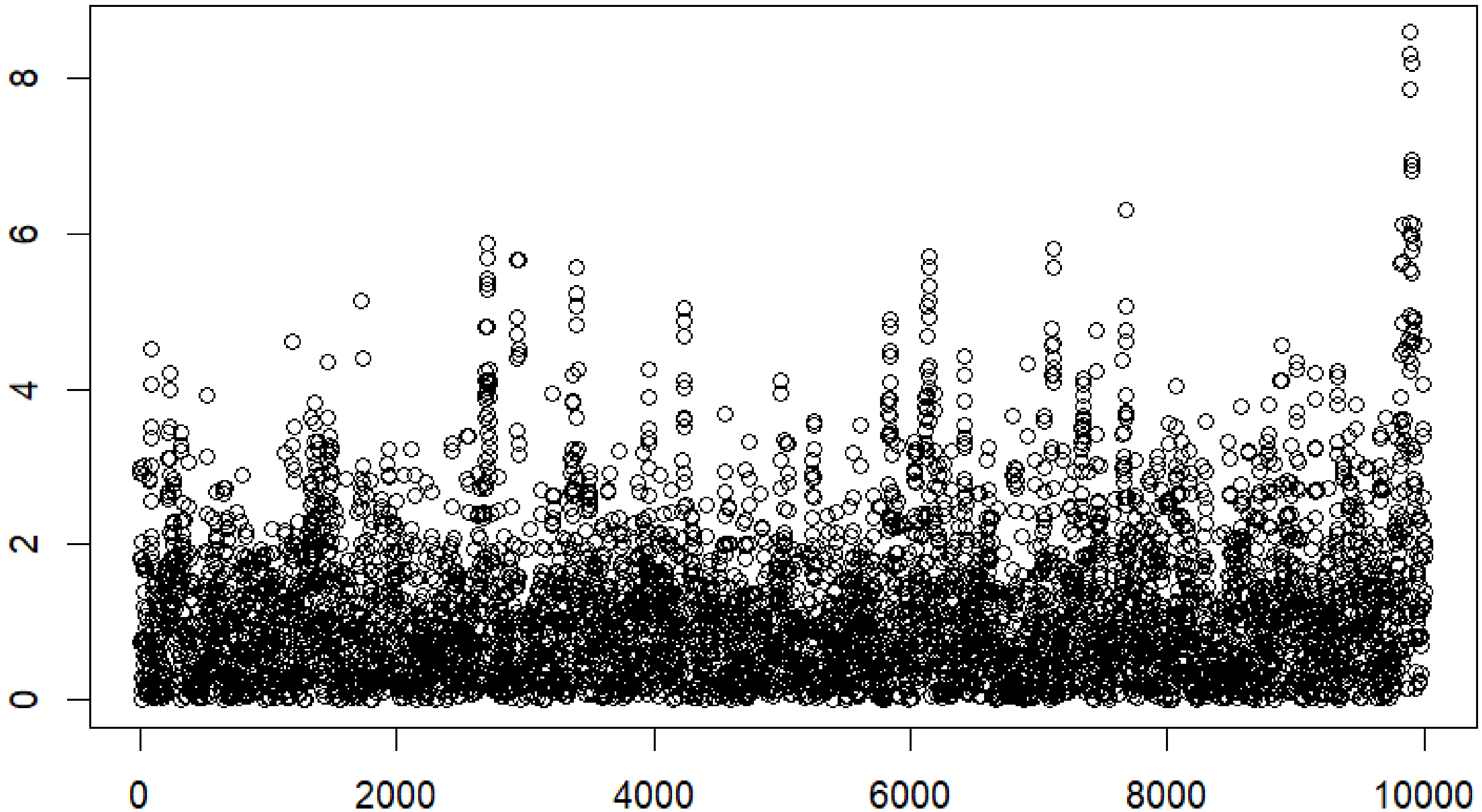
```
> # Algoritma Metropolis-Hastings untuk mendapatkan sampel yang
    sebarannya proporsional pada sebaran target (posterior)
>
> set.seed(1111)
> theta <- rep(0, 10000)
> theta[1] <- 3      # Sebagai nilai inisial bagi theta
> for (i in 2:10000) {
+   current.theta <- theta[i - 1]
> # Menggunakan Random-Walk
+   proposal.dtheta <- current.theta + rnorm(1, mean = 0, sd = 1)
+   A <- target(proposal.dtheta) / target(current.theta)
+   if (runif(1) < A) {
+     theta[i] <- proposal.dtheta
+   } else {
+     theta[i] <- current.theta
+   }
+ }
>
> # Nilai theta dari proses di atas merupakan realisasi dari Markov
    Chain
```

> # Nilai theta dari proses di atas merupakan realisasi dari Markov Chain (N = 10000)

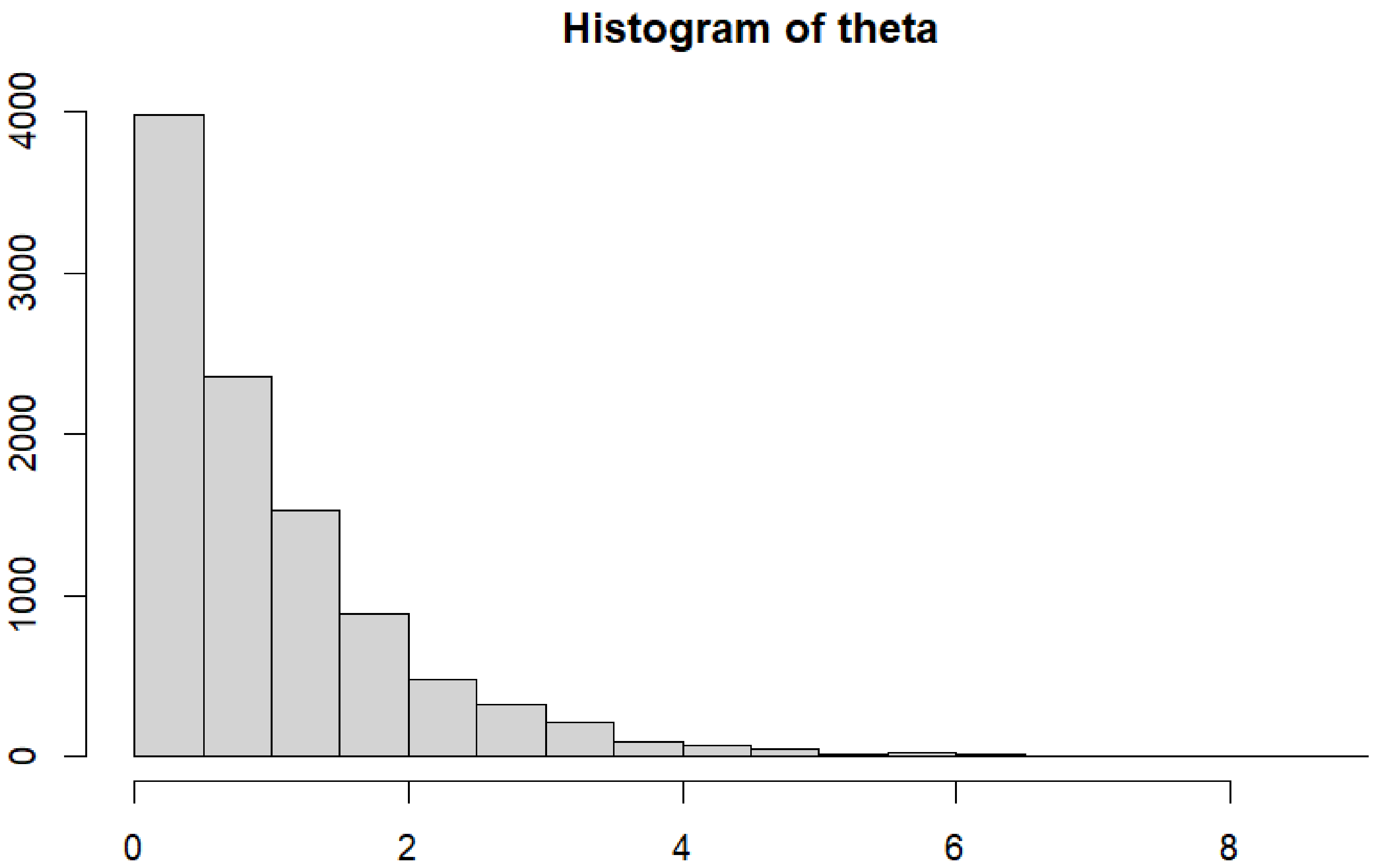
```
> theta

  [1] 3.0000000000 2.9134198887 1.8200030456 1.8200030456 0.7402045871 0.7402045871
  [7] 0.7402045871 2.0241454426 1.8253681405 0.2910865746 0.0001625694 0.0001625694
 [13] 0.0001625694 0.1124735338 0.1124735338 0.7900738330 0.7900738330 0.7900738330
 [19] 0.7900738330 0.7900738330 0.4626884062 0.4626884062 1.3838190122 1.2074620342
 [25] 1.2074620342 1.7179799001 0.2600910150 0.5680476567 0.7776767306 3.0059679173
 [31] 2.8881451089 1.7518898929 1.6694306328 1.6694306328 1.6694306328 1.0783515638
 [37] 0.1838411383 0.7689496223 0.6973162403 0.5216867201 0.5216867201 0.0841627928
 [43] 0.1700531043 0.5578498143 0.1531323266 0.1531323266 0.9450452004 0.9450452004
 [49] 0.9450452004 0.6200579158 0.6201201571 0.1489274163 0.8295695106 1.0301174526
 [55] 0.2632931578 0.0666508234 0.0666508234 0.5333585788 0.5333585788 1.1655024278
    .
    .
    .
 [967] 0.0100827629 0.0100827629 0.2344831061 0.2344831061 0.5066017372 1.0802644561
 [973] 1.0802644561 0.5790052709 0.3076113715 0.7081145189 0.7081145189 0.3796614005
 [979] 1.1317112828 1.1317112828 1.4653975502 1.4653975502 1.4653975502 1.4653975502
 [985] 1.8659347420 1.4170129586 1.2667857262 0.3088388461 0.3088388461 0.3088388461
 [991] 0.3088388461 0.3088388461 1.1143153294 0.9167967531 1.1041458576 1.1041458576
 [997] 1.1041458576 1.1979666189 1.1979666189 1.1979666189
 [ reached getOption("max.print") -- omitted 9000 entries ]
```

> # Deskripsi theta yang diperoleh

```
> plot(theta)
```



```
> hist(theta)
```



```
> # Inferensi Bayesian: penduga bagi theta
>
> mean(theta)
[1] 0.9785864

> var(theta)
[1] 0.9494195
>
> # Inferensi Bayesian: menentukan 95% credible interval
>
> nilai.kuantil <- quantile(theta, probs = c(0.025, 0.975))
> cat(paste("Pendekatan 95% credible interval : ["
+           , round(nilai.kuantil[1], 4), " ",
round(nilai.kuantil[2], 4), "]\n", sep = ""))
>
Pendekatan 95% credible interval : [0.0223 3.5293]
```

```
> # Inferensi Bayesian: penduga bagi theta
>
> mean(theta)
  0.9785864
```

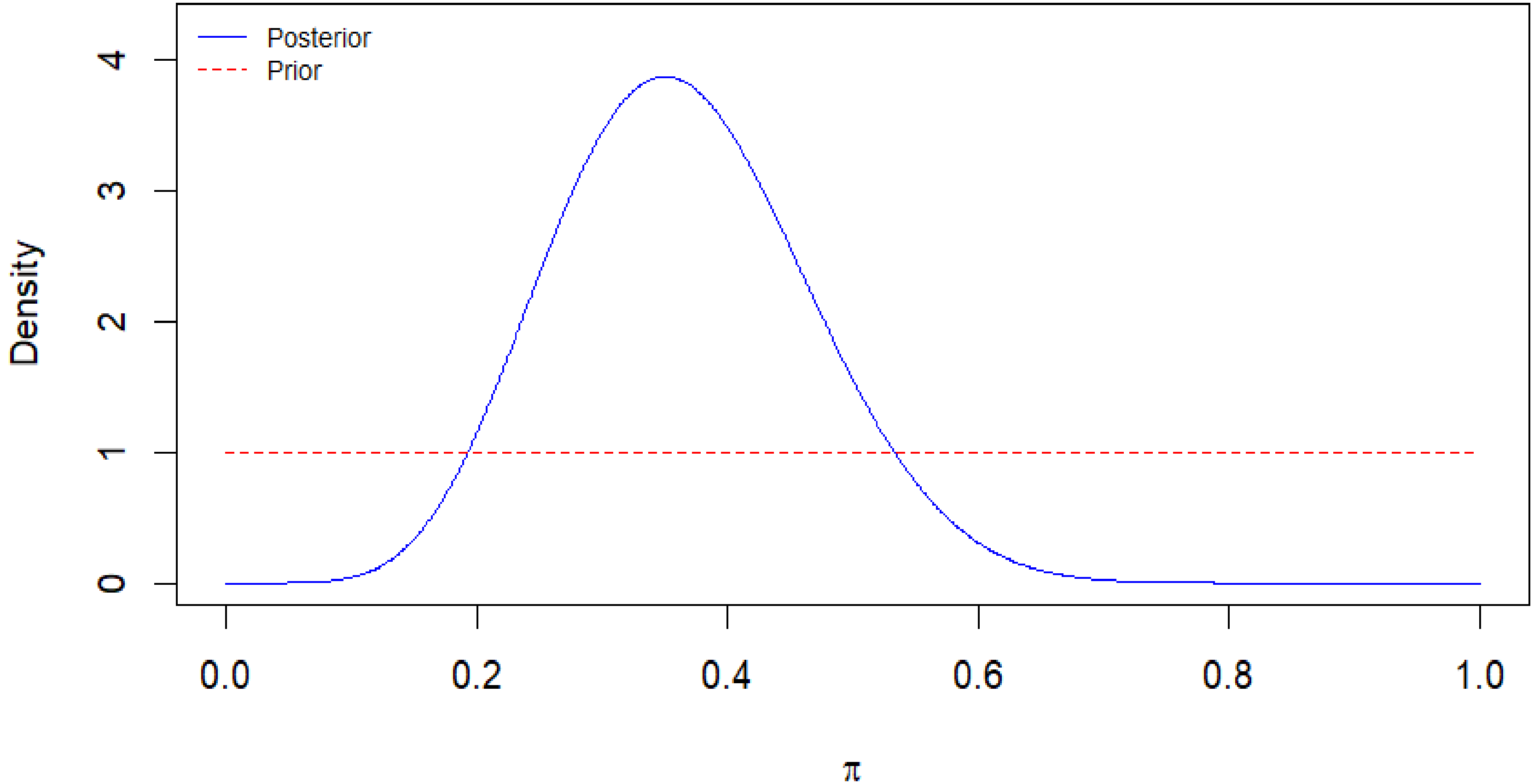
Secara analitik tentukan penduga Bayes bagi θ. Bandingkan dengan hasil yang diperoleh melalui MCMC. Apa kesimpulannya?

Studi Kasus 2:

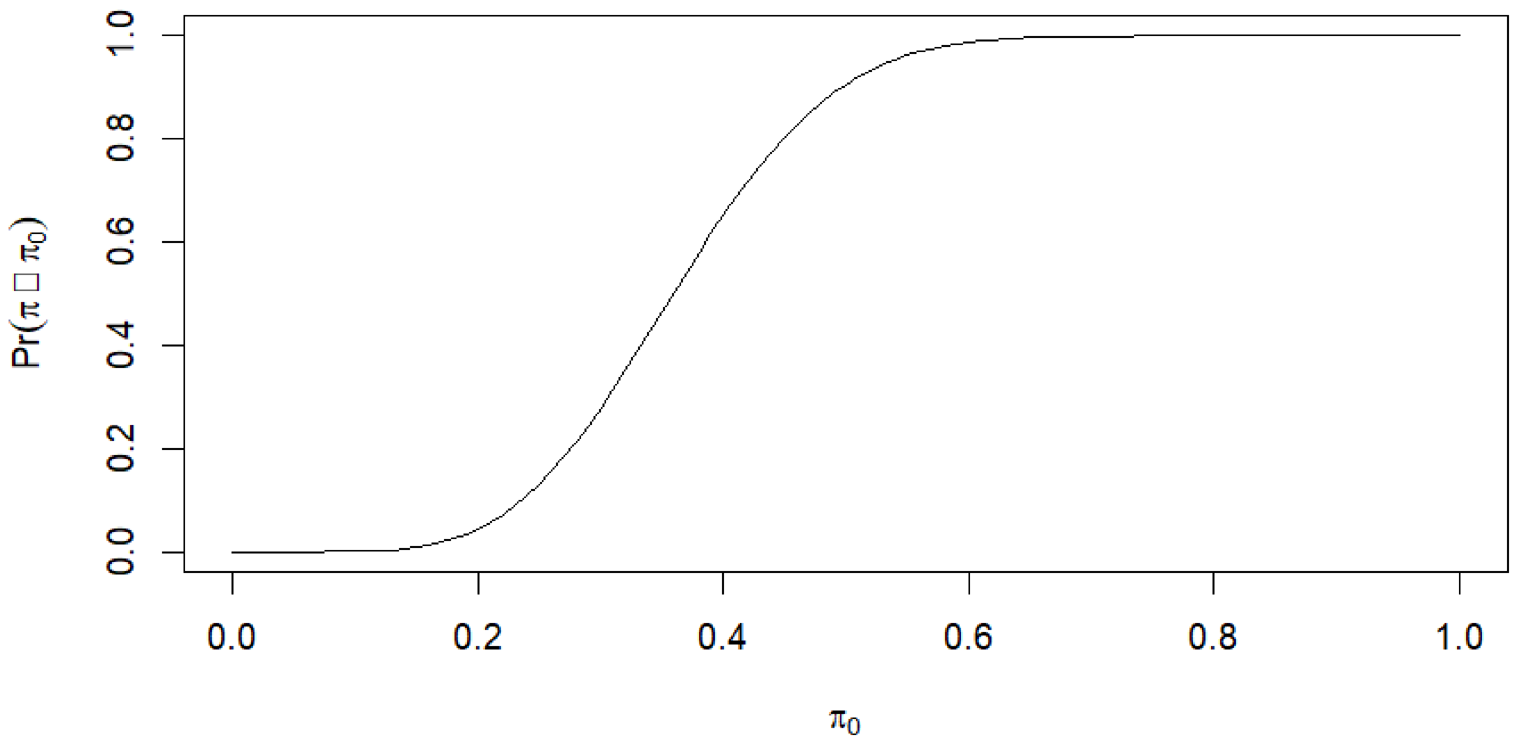
- Misalkan diketahui data menyebar Binomial(y=7, n=20), y adalah banyaknya sukses dari n percobaan.
- Sebaran prior bagi π adalah Uniform(0, 1), sedangkan π adalah peluang terjadinya sukses.
- Sebaran posterior akan dicari melalui MCMC menggunakan *function* **`binogcp()`** pada *package* "**`Bolstad`**" dalam R.

```
> ## melakukan simulasi untuk memperoleh sebaran
     posterior bagi pi melalui MCMC
>
> library(Bolstad)
>
> simulasi.post.pi <- binogcp(7, 20, density = "uniform",
                              params = c(0, 1))
```

```
> ## hasil simulasi berdasarkan MCMC
> simulasi.post.pi
```



```
> ## mencari sebaran kumulatif posterior (CDF)
     menggunakan Simpson's rule
> cdf.post = cdf(simulasi.post.pi)
> plot(cdf.post, type = "l", xlab = expression(pi[0]),
       ylab = expression(Pr(pi <= pi[0])))
```



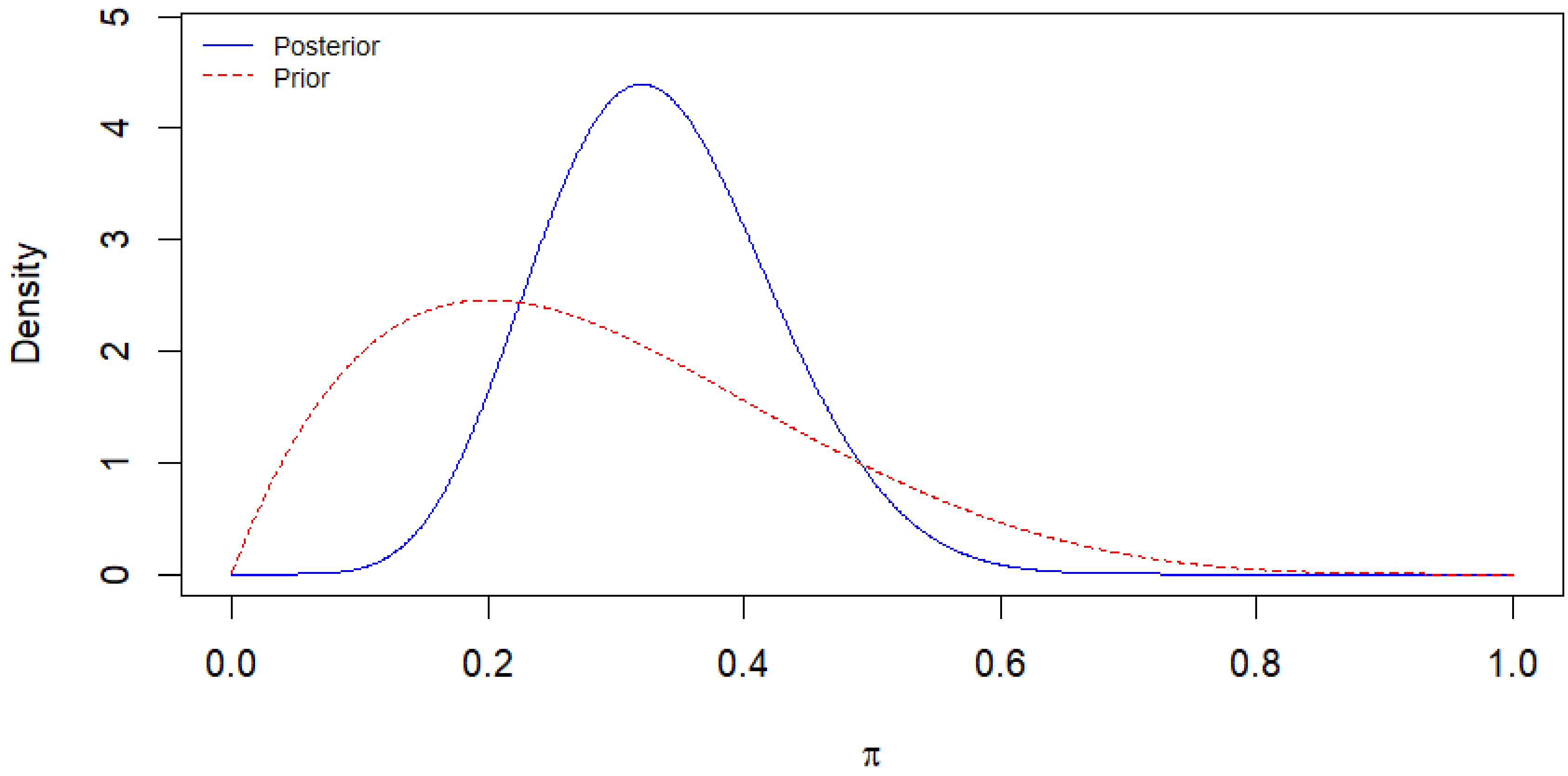
```
> ## mencari inferensi Bayesian dari posterior: nilai
     harapan dan ragam

> round(mean(simulasi.post.pi),5)
[1] 0.36364

> round(var(simulasi.post.pi),5)
[1] 0.01006

> ## menggunakan fungsi quantile untuk mencari 95%
     credible interval

> nilai.kuantil <- quantile(simulasi.post.pi, probs =
                   c(0.025, 0.975))

> cat(paste("Approximate 95% credible interval : ["
            , round(nilai.kuantil[1], 5), " ",
            round(nilai.kuantil[2], 5), "]\n", sep = ""))

Approximate 95% credible interval : [0.18057 0.56918]
```

Studi Kasus 3:

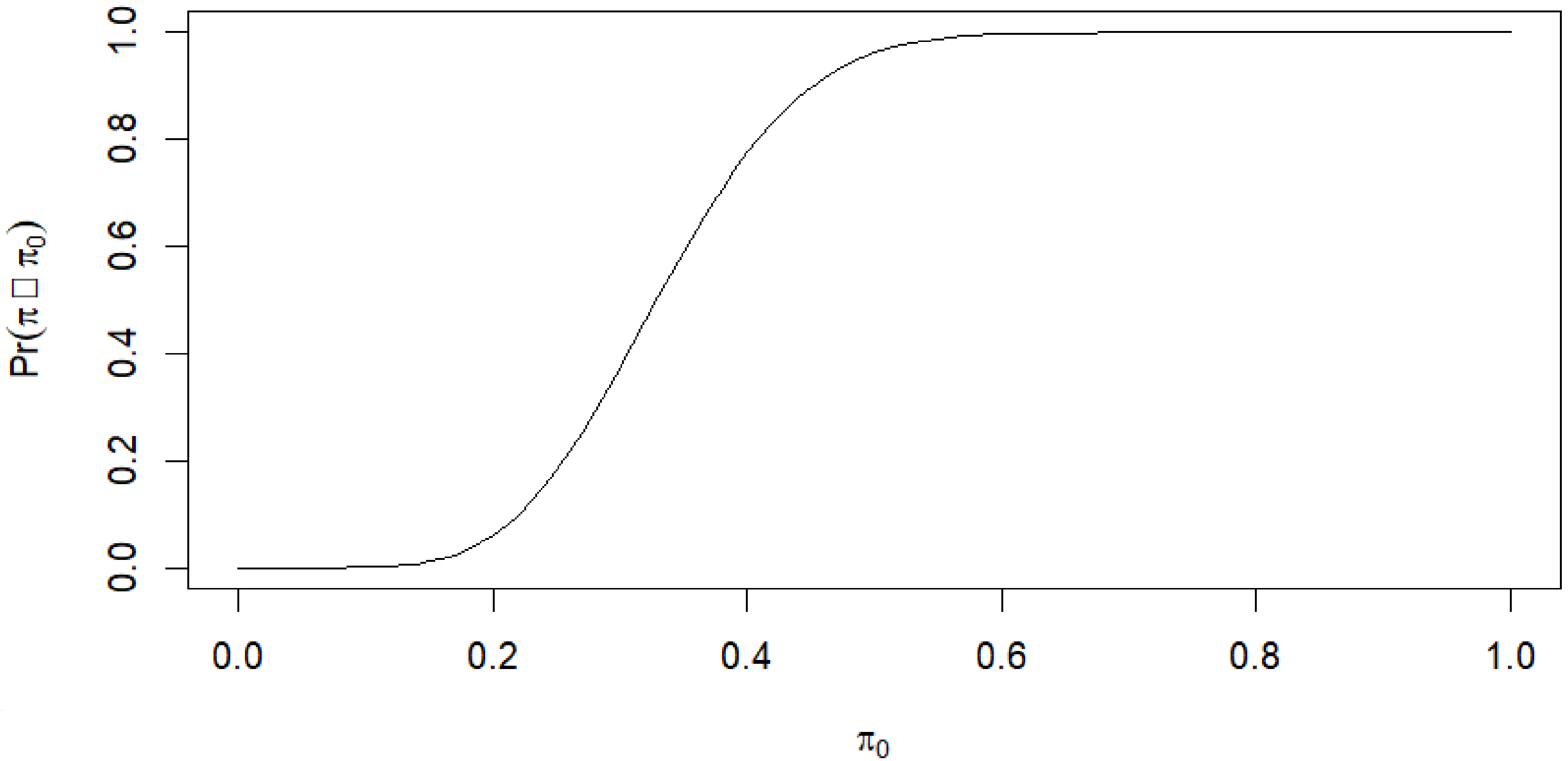
- Misalkan diketahui data menyebar Binomial(y=7, n=20), y adalah banyaknya sukses dari n percobaan.
- Sebaran prior bagi π adalah Beta(2, 5), sedangkan π adalah peluang terjadinya sukses.
- Sebaran posterior akan dicari melalui MCMC menggunakan *function* **`binogcp()`** pada *package* “**`Bolstad`**” dalam R.

```
> ## melakukan simulasi untuk memperoleh sebaran
     posterior bagi pi melalui MCMC
>
> library(Bolstad)
>
> simulasi.post.pi <- binogcp(7, 20, density = "beta",
                              params = c(2, 5))
```

```
> ## hasil simulasi berdasarkan MCMC
> simulasi.post.pi
```



```
> ## mencari sebaran kumulatif posterior (CDF)
     menggunakan Simpson's rule
> cdf.post = cdf(simulasi.post.pi)
> plot(cdf.post, type = "l", xlab = expression(pi[0]),
       ylab = expression(Pr(pi <= pi[0])))
```



```
> ## mencari inferensi Bayesian dari posterior: nilai
     harapan dan ragam

> round(mean(simulasi.post.pi),5)
[1] 0.33333

> round(var(simulasi.post.pi),5)
[1] 0.00794

> ## menggunakan fungsi quantile untuk mencari 95%
     credible interval

> nilai.kuantil <- quantile(simulasi.post.pi, probs =
                   c(0.025, 0.975))

> cat(paste("Approximate 95% credible interval : ["
            , round(nilai.kuantil[1], 5), " ",
            round(nilai.kuantil[2], 5), "]\n", sep = ""))

Approximate 95% credible interval : [0.17164 0.5174]
```

Materi Praktikum

1

a. Misalkan diketahui sebaran posterior bagi θ adalah $g(\theta|y) = 6\theta(1-\theta)$ dengan $0 < \theta < 1$.

b. Gunakan metode MCMC pada Program *R* untuk melakukan sampling bagi θ berdasarkan sebaran posterior tersebut.

c. Berdasarkan poin (b) di atas tentukan penduga Bayes bagi θ.

d. Tentukan pula *credible interval* 90% bagi θ.

e. Secara analitik tentukan penduga Bayes bagi θ. Bandingkan hasilnya dengan poin (c) di atas. Apa kesimpulan Anda.

2

a. Misalkan diketahui *Y* menyebar Poisson(λ).

b. Diketahui 5 nilai pengamatan bagi *Y* yaitu 12, 17, 10, 8, dan 23.

c. Misalkan sebaran prior yang digunakan bagi λ adalah Gamma(3, 6).

d. Melalui metode MCMC pada Program *R* lakukan sampling bagi λ. Gunakan *function* poisgcp() pada *package* "Bolstad".

e. Berdasarkan poin (d) di atas tentukan penduga Bayes bagi λ serta *credible interval* 90% bagi λ.

3

a. Misalkan diketahui *Y* menyebar Poisson(λ).

b. Diketahui 5 nilai pengamatan bagi *Y* yaitu 12, 17, 10, 8, dan 23.

c. Misalkan sebaran prior yang digunakan bagi λ adalah *Laplace's prior*, yaitu:

$$\pi(\lambda) = \frac{1}{4}\exp\left(-\frac{|\lambda-3|}{2}\right), \quad -\infty < \lambda < \infty$$

d. Melalui metode MCMC pada Program *R* lakukan sampling bagi λ. Gunakan *function* poisgcp() pada *package* "Bolstad". Sebaran prior pada poin (c) di atas harus didefinisikan terlebih dahulu di *option* density = "user".

e. Berdasarkan poin (d) di atas tentukan penduga Bayes bagi λ serta *credible interval* 90% bagi λ.

4

a. Bangkitkan 100 data peubah acak *X*, yang mana diketahui *X* menyebar Normal(μ=5, σ^2=7).

b. Misalkan sebaran prior yang digunakan bagi μ adalah *Laplace's prior*, yaitu:

$$\pi(\mu) = \frac{1}{6}\exp\left(-\frac{|\mu-2|}{3}\right), \ -\infty < \mu < \infty$$

c. Melalui metode MCMC pada Program *R* lakukan sampling bagi μ. Gunakan *function* normgcp() pada *package* "Bolstad". Sebaran prior pada poin (b) di atas harus didefinisikan terlebih dahulu di *option* density = "user".

d. Berdasarkan poin (c) di atas tentukan penduga Bayes bagi μ serta *credible interval* 90% bagi μ. Bandingkan hasilnya dengan nilai μ yang sebenarnya yaitu μ = 5. Apa kesimpulan Anda?

Pustaka

1. Reich BJ dan Ghosh SK. (2019). *Bayesian Statistical Methods*. Taylor and Francis Group.

2. Bolstad WM dan Curran JM. (2017). *Introduction to Bayesian Statistics 3th Edition*. John Wiley and Sons, Inc.

3. Lee, Peter M. (2012). *Bayesian Statistics, An Introduction 4th Edition*. John Wiley and Sons, Inc.

4. Albert, Jim. (2009). *Bayesian Computation with R*. Springer Science and Business Media.

5. Ghosh JK, Delampady M, dan Samanta T. (2006). *An Introduction to Bayesian Analysis, Theory and Methods*. Springer Science and Business Media.

Terima Kasih