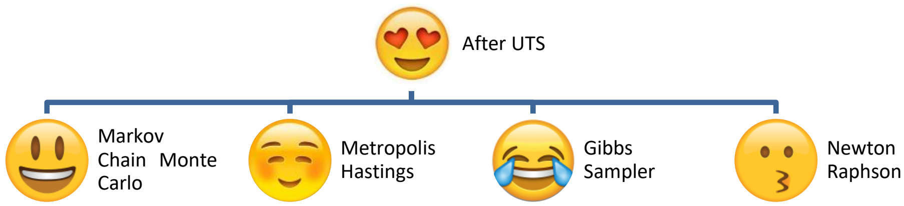
RANGKUMAN MATA KULIAH
TEKNNIK SIMULASI DATA



I. MARKOV CHAIN MONTE CARLO

a. PENGERTIAN

Markov Chain Monte Carlo (MCMC) adalah sebuah metode untuk membangkitkan peubah-peubah acak yang didasarkan pada rantai markov[1]. Dengan MCMC akan diperoleh sebuah barisan sampel acak yang berkorelasi, yakni nilai ke–*j* dari barisan $\{\boldsymbol{\theta}_j\}$ disampling dari sebuah distribusi peluang yang bergantung pada nilai sebelumnya $\{\boldsymbol{\theta}_{j-1}\}$.

b. KEGUNAAN MARKOV CHAIN MONTE CARLO

1. Untuk membentuk model yang sangat kompleks, berdemensi tinggi, atau sifat data yang berkorelasi tinggi (multicolinear)
2. Sangat khusus digunakan oleh pengguna Bayesian untuk pemodelan data
3. Membangun suatu Markov Chain yang distribusi stationeritasny aadalah berupa joint posterior dari semua parameter dan missing data yang ‘unknowns’ dari suatu model tertentu yang bersyarat pada data observasi yang diberikan.
4. Mengolah fakta yang diperoleh, dengan mengacu pada suatu aturan tertentu, untuk membentuk distribusi *joint posterior* dengan mengalikan antara "full conditional distributions" dari setiap unknown parameter dengan diberikan oleh semua parameter yang lainnya di dalam model.

c. PERCOBAAN MONTE CARLO

i. Pendahuluan

Untuk mencari nilai harapan dari $h(x)$ dimana rumusnya adalah

$$E_f\big(h(X)\big) = \int_X h(x)f(x)dx$$

Percobaan monte carlo mencoba untuk menggenerate sample $(X_1, X_2, X_3, \dots, X_n)$ yang memiliki distribusi $f(x)$, kemudian nilai integral tersebut didekati dengan

$$\overline{h_n} = \frac{1}{n}\sum_{j=1}^{n} h(x_j)$$

- Untuk **sampel besar** nilai dari $\overline{h_n} \sim \int_X h(x)f(x)dx$
- When $h^2(X)$ has finite expectation under f

  $$\frac{\overline{h_n} - E_f\big(h(X)\big)}{\sqrt{v_n}} \sim N(0,1)$$

  Follows from the Central Limit Theorem

  $$v_n = \frac{1}{n^2}\sum_{j=1}^{n}\big(h(x_j) - \overline{h_n}\big)^2$$

ii. Hal-Hal penting Pada Monte Carlo

- Terdapat evaluasi kesalahan pada metode monte carlo
- v_n diasumsikan sebagai perkiraan yang tepat untuk variance dari $\overline{h_n}$
- Jika v_n tidak konvergen, konvergen dengan lambat, maka asumsi CLT tidak dapat dipenuhi

iii. Bagaimana jika distribusi $f(x)$ tidak diketahui

Dapat melakukan transfomasi rumus seperti berikut :

$$E_f\big(h(X)\big) = \int_X h(x)f(x)dx = \int_X h(x)\frac{f(x)}{g(x)}g(x)dx = E_g\left(h(X)\frac{f(x)}{g(x)}\right)$$

Keterangan :

$f(x)$ adalah target density

$g(x)$ adalah candidate density

- $g(x)$ dapat dipilih dari distribusi yang mudah disimulasikan atau efisien dalam perkiraan integral.

- Selain itu, sampel yang sama (dihasilkan dari $g(x)$) dapat digunakan berulang kali. Tidak hanya untuk fungsi yang berbeda, tapi juga untuk kepadatan yang berbeda dari $f(x)$.

d. PERCOBAAN MARKOV CHAIN

i. Pendahuluan

Markov Chain $\{X^{(t)}\}$ adalah sebuah statistic urut dependent random variable dari $X^{(0)}, X^{(1)}, X^{(2)}, \dots, X^{(t)}$, dimana $X^{(t)}$ **hanya dependen** pada $X^{(t-1)}$

The conditional distribution of X(t)|X(t−1) is a transition kernel K,

X(t+1) | X(0), X(1), X(2), . . . , X(t) ~ K(X(t), X(t+1)).

Contohnya adalah proses random walk

X(t+1) = X(t) + et , et ~ N (0, 1),

The Markov kernel K(X(t), X(t+1)) corresponds to a N (X(t), 1) density.

Markov Chain Monte Carlo (MCMC) biasanya memiliki kekuatan yang sangat kuat dalam stabilitas properti.

MCMC dapat dirumuskan sbb:

Jika $X^{(t)} \sim f$ dan $X^{(t+1)} \sim f$ kita mendapatkan persamaan

$$\int_X K(x,y) f(x) dx = f(y)$$

- MCMC Markov chains are also irreducible, or else they are useless
  - The kernel K allows for free moves all over the state-space
  - For any $X^{(0)}$, the sequence $\{X^{(t)}\}$ has a positive probability of eventually reaching any region of the state-space
- MCMC Markov chains are also recurrent, or else they are useless
  - They will return to any arbitrary nonnegligible set an infinite number of times

Catatan tambahan :

- Suatu kumpulan data missal X1,X2,X3,… dikatakan Markov Chain jika Xi|Xi-k,..,Xi-1 hanya bergantung pada Xi-1. Kemudian, Markov Chain dikatakan stationary transition probabilities jika semua data Xi tidak bergantuk pada data ke Xn jadi kalau di grafik nilainya konvergen ke titik tertentu.Serta juga Markov Chain dikatakan stationer jika dan hanya jika marginal distribution dari Xn tidak tergantung pada n. Initial distribution dikatakan stationer untuk semua transition probability jika Markov chain di spesifikasikan dengan initial distribution itu stationer.

- MCMC digunakan untuk mencapai titik optimal atau titik equilibrium.
- Prior + data menghasilkan posterior. Misal : gerakan robot ditentukan gerakan awalnya. JIka diberi perlakuan tertentu akan membentuk suatu pola. Jika diberi perlakuan yang berbeda lagi akan membentuk pola yang berbeda pula. Gerakan robot yang ditentukan gerakan awalnya disebut sebagai distribusi prior, sedangkan perlakuan yang diberikan disebut data atau perintah.Lalu, pola yang diperoleh disebut dist. Posterior.
- Data augmentation digunakan pada gibb sampler.
- Perumpamaan marginal distribution seperti missal X|teta dimana teta merupakan parameter tertentu. Intinya suatu conditional distribution yang paling kecil yang tidak bisa dibreakdown lagi atau tidak bisa dipecah lagi. Biasanya disebut juga sebagai initial distribution.
- MCMC merupakan proses yang stokastic bukan deterministic
- Biasanya state space pada MCMC itu tidak terbatas sehingga susah untuk menentukan initial distribution.
- Algoritma Metropolis-Hastings-Green juga digunakan untuk mencapai titik optimal atau equilibrium sama seperti MCMC.
- Suatu Markov Chain dikatakan reversible jika pasangan distribusi dari (Xi,Xi+1) dapat ditukar. Jika sutu data Markov Chain reversibility maka dia stationer, tetapi jika stationer belum tentu reversibility. Markov Chain yang reversible , jika datanya dibalik maupun tidak memiliki distribusi yang sama. Misal : (xi+1,…xi+k) berdistribusi sama jika data itu dibalik (xi+k,…,xi+1).

II. METROPOLIS HASTINGS

Algoritma Metropolis-Hastings (M-H) merupakan salah satu algoritma untuk mencari Markov yang tak tereduksi dan tak berkala dan diberi nama setelah Metropolis *et al.* (1953) dan Hast-ings (1970). Secara garis besar algoritma ini dapat dijelaskan sebagai berikut. Misalkan *f* adalah densitas target, *q.y*ð*x*/ adalah densitas proposal atau instrumental yang mudah disim-ulasikan, secara eksplisit tersedia, atau simetrik. Syarat umum yang diperlukan adalah rasio *f* *.y*/_*q.y*ð*x*/ harus diketahui sampai suatu konstanta tertentu yang bebas dari *x*.

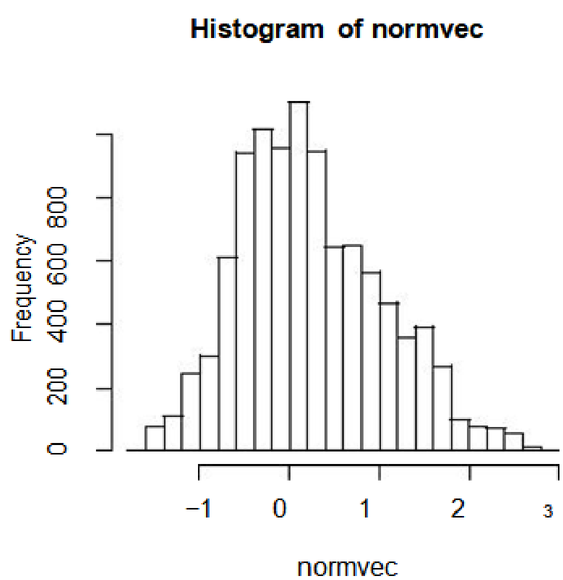
Algoritma *Metropolis-Hastings* berguna untuk membangkitkan barisan sampel dari su-atu distribusi probabilitas yang sulit untuk dilakukan penarikan sampel dengan menggunakan mekanisme penerimaan dan penolakan. Barisan ini dapat digunakan untuk mengaproksimasi distribusi dengan histogram, atau untuk menghitung integral.

Algoritma Metode Metropolis Hastings :

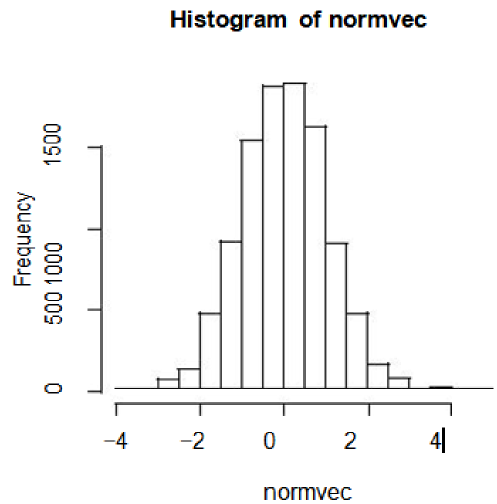
1. Ambil nilai awal, yaitu $\boldsymbol{\theta}_0$. Untuk iterasi j = 1,
2. Bangkitkan $\boldsymbol{\theta}* \sim p(\boldsymbol{\theta} \mid \boldsymbol{\theta} j - 1)$.
3. Bangkitkan sampel acak *u* dari distribusi uniform U[0,1]
4. Jika $u < \min\left(1, \frac{p(\theta*|\, X,y)\, p(\theta_{j-1}|\theta*)}{p(\theta_{j-1}|\, X,y)\, p(\theta*|\theta_{j-1})}\right)$ ambil $\theta_j = \theta *$

   Namun jika $u > \min\left(1, \frac{p(\theta*|\, X,y)\, p(\theta_{j-1}|\theta*)}{p(\theta_{j-1}|\, X,y)\, p(\theta*|\theta_{j-1})}\right)$ ambil $\theta_j = \theta_{j-1}$
5. Ulangi langkah 1-3 sampai dengan jumlah yang diinginkan.

Gambar 1 dan 2 menunjukkan histogram dari sampling dengan Algoritma M-H



Gambar 1: Histogram hasil simulasi $N(0; 1)$, dengan N=10000 dan a=0.1



Gambar 2: Histogram hasil simulasi $N(0; 1)$, dengan N=10000 dan a=1

III. GIBBS SAMPLER

Gibbs Sampling bisa diterapkan apabila distribusi probabilitas bersama *(joint proba-bility distribution)* tidak diketahui secara eksplisit, tetapi distribusi bersyarat *(conditional dis-tribution)* dari tiap-tiap variabel diketahui. Algoritma *Gibbs sampling* bisa dituliskan sebagai berikut:

1) Tentukan nilai awal $x^0 = x_1^0 ; \dots ; x_p^0)$
2) Ulangi langkah untuk $j = 1; 2; \dots ; M$

   Bangkitkan X_1^{j+1} dari $f_1(x_1|x_2^{(j)}, x_3^{(j)}. \dots, x_p^{(j)})$

   Bangkitkan X_2^{j+1} dari $f_2(x_2|x_1^{(j)}, x_3^{(j)}. \dots, x_p^{(j)})$

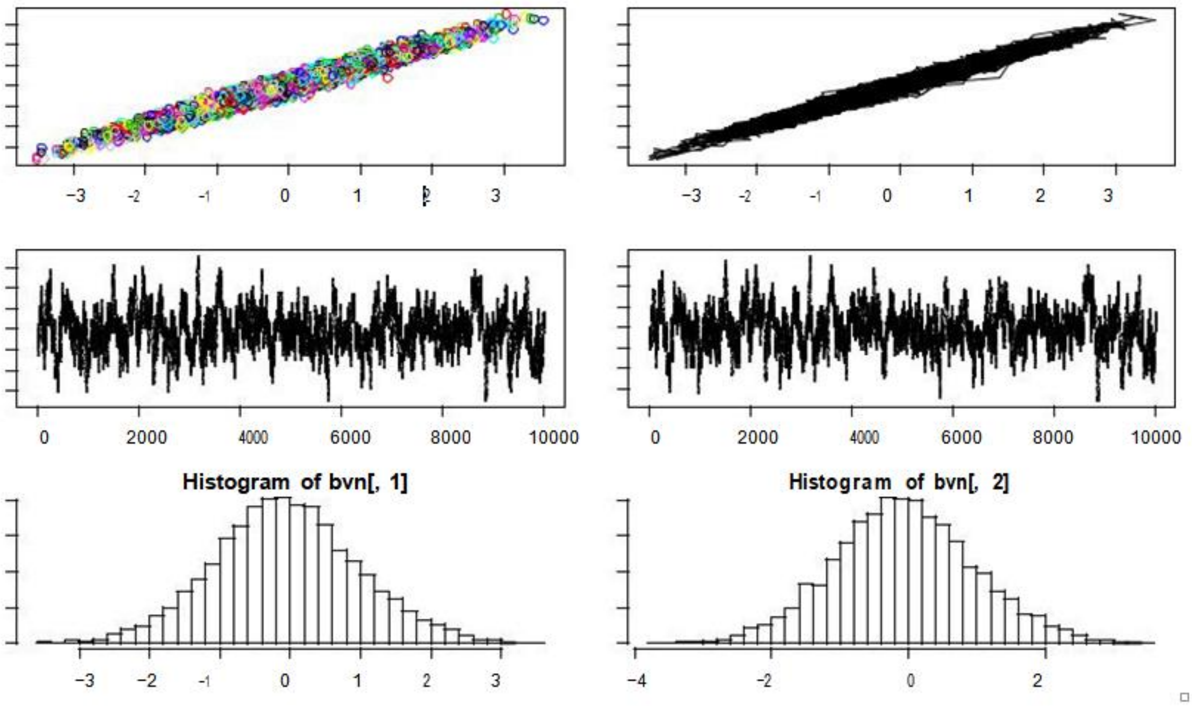
   $\vdots$

   Bangkitkan X_p^{j+1} dari $f_p(x_p|x_1^{(j)}, . \dots, x_p^{(j)})$
3) Kembalikan nilai $x^1; x^2; \dots; x^M$

densitas $f1; f2; \dots; fp$ disebut distribusi bersyarat penuh, dan densitas yang digunakan untuk simulasi. Walaupun dalam dimensi tinggi semua simulasi adalah univariate. Dalam *gibbs sam-pling* tidak ada mekanisme penerimaan dan penolakan semua sampel hasil simulasi diterima.

berikut adalah kode program dengan menggunakan R-Statistik.

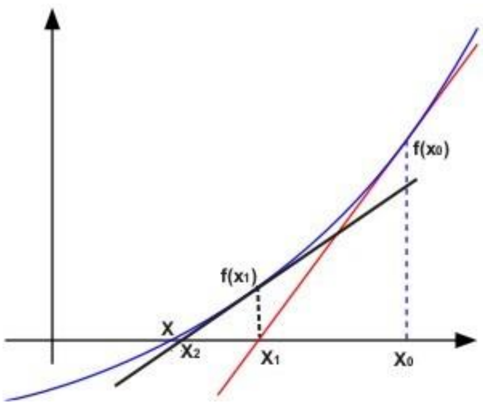
```
# Gibbs sampling untuk kasus bivariate normal...
# Gibbs sampling dengan n=banyaknya iterasi dan rho=parameter
        korelasi gibbs<-function(n, rho) {


        #Alokasi matriks

        mat <- matrix(ncol = 2, nrow
        = n) #nilai awal

        x <- 0
        y <- 0
        mat[1, ] <- c(x, y)

        #Loop

        for (i in 2:n) {

                #Komponen x

                x <- rnorm(1, rho * y, sqrt(1 - rho^2))

                #Komponen y

                y <- rnorm(1, rho * x, sqrt(1 - rho^2))

                #Matriks hasil simulasi

                mat[i, ] <- c(x, y)

                        }

                Mat

                        }

                #Cara Pemanggilan

                        bvn<-gibbs(10000,0.98)
```



Gambar 1: Hasil simulasi Gibbs sampling pada distribusi bivariate Normal

IV. NEWTON RAPHSON

Metode Newton-Raphson adalah metode pencarian akar suatu fungsi f(x) dengan pendekatan satu titik, dimana fungsi f(x) mempunyai turunan. Metode ini dianggap lebih mudah dari **Metode Bagi-Dua** (Bisection Method) karena metode ini menggunakan pendekatan satu titik sebagai titik awal. Semakin dekat titik awal yang kita pilih dengan akar sebenarnya, maka semakin cepat konvergen ke akarnya.



Algoritma Metode Newton Raphson :

1. Tentukan Harga fungsi $f(xi)$
2. Tentukan Harga Awal (xi)
3. Tentukan Interval = [a ; b} dengan jumlah pembagi Δh
4. Tentukan toleransi kesalahan (εs) dan iterasi maksimum (n)
5. Hitung nilai fungsi $f(xi)$ dan turunannya $f'(xi)$
6. Hitung nilai Xi+1 menggunakan rumus : $X_{i+1} = X_i - \frac{f(x_i)}{f'(x_i)}$
7. Hitung kesalahan dan bandingkan dengan toleransi kesalahan yang diizinkan
   a. Jika εa > εs , maka ulangi langkah ke-2
   b. Jika εa < εs ,maka iterasi selesai dan $Xi + 1$ sebagai akar persamaan
8. Akar persamaan adalah Xi terakhir yang diperoleh.

Contoh Soal :

Tentukan akar dari persamaan $4x^3 - 15x^2 + 17x - 6 = 0$ menggunakan Metode Newton-Raphson.

Penyelesaian :

$f(x) = 4x^3 - 15x^2 + 17x - 6$

$f'(x) = 12x^2 - 30x + 17$

iterasi 1 :

ambil titik awal $x_0 = 3$

$f(3) = 4(3)^3 - 15(3)^2 + 17(3) - 6 = 18$

$f'(3) = 12(3)^2 - 30(3) + 17 = 35$

$$x_1 = 3 - \frac{18}{35} = 2.48571$$

iterasi 2 :

$f(2.48571) = 4(2.48571)^3 - 15(2.48571)^2 + 17(2.48571) - 6 = 5.01019$

$f'(2.48571) = 12(2.48571)^2 - 30(2.48571) + 17 = 16.57388$

$$x_2 = 2.48571 - \frac{5.01019}{16.57388} = 2.18342$$

iterasi 3 :

$f(2.18342) = 4(2.18342)^3 - 15(2.18342)^2 + 17(2.18342) - 6 = 1.24457$

$f'(2.18342) = 12(2.18342)^2 - 30(2.18342) + 17 = 8.70527$

$x_3 = 2.18342 - \frac{1.24457}{8.70527} = 2.04045$

iterasi 4 :

$f(2.04045) = 4(2.04045)^3 - 15(2.04045)^2 + 17(2.04045) - 6 = 0.21726$

$f'(2.04045) = 12(2.04045)^2 - 30(2.04045) + 17 = 5.74778$

$x_4 = 2.04045 - \frac{0.21726}{5.74778} = 2.00265$

iterasi 5 :

$f(3) = 4(2.00265)^3 - 15(2.00265)^2 + 17(2.00265) - 6 = 0.01334$

$f'(2.00265) = 12(2.00265)^2 - 30(2.00265) + 17 = 5.04787$

$x_5 = 2.00265 - \frac{0.01334}{5.04787} = 2.00001$

iterasi 6 :

$f(2.00001) = 4(2.00001)^3 - 15(2.00001)^2 + 17(2.00001) - 6 = 0.00006$

$f'(2.00001) = 12(2.00001)^2 - 30(2.00001) + 17 = 5.00023$

$x_6 = 2.00001 - \frac{0.00006}{5.00023} = 2.00000$

iterasi 7 :

$f(2) = 4(2)^3 - 15(2)^2 + 17(2) - 6 = 0$

jika disajikan dalam tabel, maka seperti tabel dibawah ini.

n	x_n	$f(x_n)$	$f'(x_n)$
0	3	18	35
1	2.48571	5.01019	16.57388
2	2.18342	1.24457	8.70527
3	2.04045	0.21726	5.74778
4	2.00265	0.01334	5.04787
5	2.00001	0.00006	5.00023
6	2.00000	0.00000	5.00000

karena pada iteasi ketujuh $f(x_6) = 0$ maka akar dari persamaan tersebut adalah x = 2.