

IMPLEMENTASI MARKOV CHAIN MONTE CARLO PADA PENDUGAAN HYPERPARAMETER REGRESI PROSES GAUSSIAN

Moch. Abdul Mukid¹, Sugito¹

^{1,2} Staf Pengajar Program Studi Statistika FMIPA Universitas Diponegoro

mamukid@yahoo.com

Abstract

This paper studies the implementation of Markov Chain Monte Carlo on estimating the hyperparameter of Gaussian process. Metropolis-Hasting (MH) algorithm is used to generate the random samples from the posterior distribution that can not be generated by a direct simulation method. This algorithm require only a proposal distribution for generating a candidate point. In this paper uniform distribution is chosen as the proposal distribution.

Keywords: Markov Chain Monte Carlo, Gaussian Process, Metropolis-Hasting Algorithm

1. Pendahuluan

Terdapat dua pendekatan utama dalam mengembangkan sebuah model dengan menggunakan metode regresi. Pendekatan yang pertama adalah parametrik. Dalam pendekatan ini pertama-tama peneliti menganggap bahwa antara peubah tak bebas dengan peubah bebas yang diamati mengikuti suatu fungsi matematik tertentu, misalnya fungsi linear kuadratik maupun fungsi trigonometri^[11]. Dengan menggunakan data dari sampel, kemudian parameter-parameter model diduga. Selanjutnya model dievaluasi dengan memperhatikan ukuran-ukuran kecocokan model. Dalam kasus di mana banyaknya peubah bebasnya hanya satu maupun adanya landasan teori yang cukup kuat untuk menentukan spesifikasi modelnya, regresi parametrik efektif untuk digunakan. Namun ketika peubah bebas yang diamati cukup banyak, akan sangat sulit bagi peneliti untuk mendapatkan informasi tentang fungsi yang layak yang menghubungkan peubah-peubah yang dimilikinya.

Jika informasi tentang bentuk fungsi tidak mencukupi maka pendekatan secara nonparametrik dapat dilakukan. Dalam pendekatan nonparametrik peneliti tidak perlu melakukan spesifikasi model terlebih dahulu, sehingga tidak ada parameter model yang harus diestimasi. Data dibiarkan “berbicara” untuk menemukan model yang relevan. Agar pendekatan nonparametrik ini menghasilkan estimasi terhadap fungsi yang masuk akal, maka hal yang harus diperhatikan adalah bahwa fungsi tersebut diasumsikan mulus dan kontinu pada daerah yang diinginkan^[11].

Banyak metode nonparametrik yang mendasarkan pada konsep pemulusan. Pemulusan telah menjadi sinonim dengan metode-metode nonparametrik yang digunakan untuk mengestimasi fungsi-fungsi. Tujuan dari pemulusan ini adalah untuk membuang variabilitas dari data yang tidak memiliki efek sehingga ciri-ciri dari data akan tampak lebih jelas^[4].

Dalam tulisan ini dikaji tentang pendugaan hyper-parameter regresi proses Gaussian dengan menggunakan pendekatan simulasi tak langsung terhadap distribusi posterior bagi *hyperparameter*. Metode *Markov Chain Monte Carlo* (MCMC) dipilih untuk melakukan simulasi tersebut.

Regresi proses Gaussian dapat dijelaskan dari sudut pandang regresi nonparametrik Bayesian, yaitu dengan menempatkan secara langsung sebaran prior Gaussian bagi fungsi-fungsi regresi f [12]. Pada tulisan terdahulu, penulis menggunakan metode *marginal maximum likelihood* (MML) untuk menduga nilai hyperparameter dari proses Gaussian [7]. Implementasi metode ini dilakukan dengan menggunakan metode *conjugate gradient*. Metode MML sensitive terhadap nilai awal yang berbeda dan biasanya konvergen ke sebuah titik optimum lokal. Simulasi MCMC dilakukan untuk mengatasi masalah-masalah ini.

Sistematika dari tulisan ini adalah sebagai berikut. Bagian pertama dijelaskan tentang berbagai pendekatan dalam pemodelan menggunakan regresi dan posisi regresi proses Gaussian dalam pemodelan tersebut. Selanjutnya bagian 2 dari tulisan ini menjelaskan tentang formulasi model dan prediksi dalam proses Gaussian. Bagian 3 menjelaskan tentang distribusi prior dan posterior untuk *hyperparameter*. Bagian 4 dibahas secara singkat tentang simulasi MCMC beserta algoritma-algoritma yang dapat digunakan. Di bagian 5 dari tulisan ini menjelaskan tentang penerapan model proses Gaussian untuk masalah real. Selanjutnya hasil-hasil yang diperoleh dari penerapan di bagian 5 dibahas dibagian 6. Akhirnya bagian terakhir dari tulisan ini akan disampaikan mengenai kesimpulan yang dapat diperoleh dari kajian dan penerapan yang telah dilakukan pada bagian sebelumnya.

2. Tinjauan Pustaka

2.1 Regresi Proses Gaussian

Misal untuk setiap output y_i bergantung pada input \mathbf{x}_i di bawah sebuah fungsi f_i yang dimodelkan dengan

$$y_i = f(\mathbf{x}_i) + \varepsilon_i \quad (1)$$

dengan ε_i adalah peubah acak galat yang secara bebas dan identik menyebar secara Gaussian dengan rata-rata nol dan ragam σ_ε^2 . \mathbf{x}_i adalah vektor input ke- i dimana $i = 1, \dots, n$. Apabila fungsi-fungsi f dikumpulkan dalam sebuah vektor $\mathbf{f} = [f_1, \dots, f_n]^T$ maka menurut Proses Gaussian sebaran prior atas vektor \mathbf{f} adalah Gaussian ganda dengan vektor rata-rata $\mathbf{0}$ dan matrik ragam-peragam \mathbf{K} , yaitu

$$\mathbf{f} | \mathbf{X}, \boldsymbol{\theta} \sim N(\mathbf{0}, \mathbf{K}) \quad (2)$$

dengan \mathbf{K} adalah matrik berukuran $(n \times n)$ yang bergantung pada nilai \mathbf{X} dan $\boldsymbol{\theta}$ sedangkan $\boldsymbol{\theta}$ adalah vektor parameter dari fungsi peragam yang kemudian disebut dengan *hyperparameter* [6],[10]. Setiap elemen (i,j) dari matrik \mathbf{K} adalah $k(\mathbf{x}_i, \mathbf{x}_j)$ dimana $k(\dots)$ adalah sebuah fungsi yang memuat parameter $\boldsymbol{\theta}$. Selanjutnya $k(\dots)$ disebut sebagai fungsi peragam. Apabila banyaknya variabel bebas hanya satu maka notasi fungsi peragamnya dapat ditulis dengan $k(x_i, x_j)$. Hal ini berarti input modelnya merupakan sebuah skalar.

Fungsi peragam adalah sebuah fungsi dari dua buah input model yang hasilnya adalah peragam bagi output-output yang bersesuaian dengan dua input tersebut [8]. Dari sudut pandang proses Gaussian, fungsi peragam mendefinisikan tentang kedekatan atau kemiripan. Titik-titik input yang posisinya berdekatan akan memiliki kemiripan didalam output-outputnya. Oleh karena itu titik-titik pelatihan (*training points*) yang posisinya berdekatan dengan titik-titik uji (*test points*) seharusnya dapat menjadi informasi bagi prediksi output pada titik-titik uji tersebut [10]. Beberapa fungsi peragam yang umum digunakan dalam model proses Gaussian adalah fungsi peragam eksponensial kuadrat,

fungsi peragam linear, fungsi peragam kelas Matern, dan lain-lain. Fungsi peragam juga dapat diperoleh dengan mengkombinasikan fungsi-fungsi peragam di atas, karena penjumlahan maupun perkalian dari fungsi-fungsi peragam akan menghasilkan sebuah fungsi peragam juga^[9]. Salah satu fungsi peragam yang digunakan secara luas adalah eksponensial kuadrat yang formula matematisnya sebagai berikut:

$$k(x_i, x_j) = v_0 \exp\left(-\frac{1}{2} \left(\frac{x_i - x_j}{w}\right)^2\right) \quad (3)$$

dengan $\theta = (v_0, w)$ adalah sebuah vektor parameter yang biasa disebut dengan *hyperparameter*. Istilah *hyperparameter* dipilih untuk menjelaskan bahwa ini adalah parameter dari fungsi peragam dan bukan parameter-parameter sebagaimana dalam pendekatan parametrik. Satu-satunya syarat bagi sebuah fungsi peragam adalah mampu membangkitkan sebuah matrik ragam peragam yang definit non negatif^[10].

Terdapat beberapa metode yang dapat digunakan untuk menduga nilai-nilai *hyperparameter*. Nilai θ dapat diduga dengan menggunakan metode *Maximum Marginal Likelihood*, metode aposterior maksimum, dan metode simulasi hybrid Monte Carlo. Metode lain yang bisa digunakan adalah metode *Cross Validation* dan metode *Generalized Cross Validation*^[10].

Dalam tulisan ini nilai *hyperparameter* θ diduga dengan menggunakan pendekatan Bayesian MCMC. Simulasi MCMC sering digunakan dalam inferensi Bayesian, khususnya jika distribusi posterior dari parameter yang diperhatikan memiliki bentuk yang tidak standar dan rumit^[5]. Dalam inferensi Bayesian, parameter dianggap sebagai sebuah peubah acak yang mengikuti sebuah distribusi tertentu. Distribusi ini biasa disebut dengan distribusi prior.

Setelah nilai θ diduga, selanjutnya prediksi dalam regresi proses Gaussian dapat dilakukan. Misal x^* adalah sebuah titik uji dan f^* adalah fungsi yang bersesuaian dengan x^* , maka di bawah kerangka kerja proses Gaussian, sebaran bersama dari \mathbf{f} dan f^* adalah Gaussian Ganda dengan rata-rata nol, yaitu:

$$\begin{bmatrix} \mathbf{f} \\ f^* \end{bmatrix} | \mathbf{X}, \theta \sim N\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{k} \\ \mathbf{k}^T & \kappa \end{bmatrix}\right) \quad (4)$$

dimana $\mathbf{k} = [k(x^*, x_1), \dots, k(x^*, x_n)]^T$ adalah vektor berukuran $(n \times 1)$ yang dibentuk dari peragam antara x^* dan input-input model. Sedangkan $\kappa = k(x^*, x^*)$ adalah sebuah skalar. Apabila peubah galat mengikuti sebaran seperti pada persamaan (1) maka sebaran bersama dari peubah \mathbf{y} dan y^* adalah

$$\begin{bmatrix} \mathbf{y} \\ y^* \end{bmatrix} | \mathbf{X}, \theta \sim N\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\varepsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & \kappa + \sigma_\varepsilon^2 \end{bmatrix}\right) \quad (5)$$

Sehingga sebaran marginal dari y^* adalah juga Gaussian, yaitu :

$$y^* | \mathbf{y}, \mathbf{X}, \theta \sim N(m(x^*), v(x^*)) \quad (6)$$

dengan rata-rata dan ragamnya adalah

$$m(x^*) = \mathbf{k}^T (\mathbf{K} + \sigma_\varepsilon^2 \mathbf{I})^{-1} \mathbf{y} \quad (7)$$

$$v(x^*) = \kappa + \sigma_\varepsilon^2 - \mathbf{k}^T (\mathbf{K} + \sigma_\varepsilon^2 \mathbf{I})^{-1} \mathbf{k} \quad (8)$$

Nilai dugaan untuk y^* adalah $m(x^*)$ dan ragam untuk dugaan y^* adalah $v(x^*)$.

Kinerja dari model regresi proses Gaussian dalam melakukan prediksi diukur berdasarkan nilai RMSE (*Root Mean Square Error*) yang diperoleh. Makin kecil nilai RMSE maka makin baik pula kinerja dari model tersebut.

2.2 Sebaran Prior dan Posterior Hyperparameter

Dalam makalah ini sebaran prior untuk hiperparameter θ mengadopsi pada penelitian yang dilakukan oleh Neal dan Rasmussen dengan melakukan beberapa perubahan pada nilai parameternya^{[8],[2]}. Prior untuk v_0 adalah Gaussian $(-1,1)$. Prior untuk

w^{-1} adalah distribusi invers gamma $\left(\frac{1}{2}, \frac{1}{2}\right)$, yaitu

$$p(w^{-1}) = \frac{\left(\frac{1}{2}\right)^{1/2}}{\Gamma\left(\frac{1}{2}\right)} (w^{-1})^{-3/2} \exp(-w/2) \quad , \quad w^{-1} > 0$$

Perubahan nilai pada v_0 dan w^{-1} akan mempengaruhi nilai parameter pada sebaran peubah acak galat. Oleh karena itu parameter σ_ε^2 dianggap sebagai peubah acak dengan memberikan sebaran prior invers gamma $\left(\frac{1}{2}, 1\right)$. Untuk selanjutnya hyperparameter θ akan memuat komponen σ_ε^2 juga.

Sebaran posterior bagi peubah acak θ memiliki nilai yang sebanding dengan hasil kali antara sebaran prior setiap parameter dalam θ dan distribusi dari data (*likelihood*). Secara matematik dapat ditulis dengan

$$p(\theta | \mathbf{X}, \mathbf{y}) \propto \left(-w^{-1}\sigma_\varepsilon^2\right)^{1/2} \exp\left(-\frac{1}{2}(v_0+1)^2 - \frac{1}{2}\mathbf{y}^T(\mathbf{K} + \sigma_\varepsilon^2)^{-1}\mathbf{y} - 0,5w - 1/\sigma_\varepsilon^2\right)$$

Sebaran posterior ini berguna untuk menentukan peluang penerimaan bagi titik kandidat yang dipilih sebagai anggota sampel acak untuk hyperparameter θ .

2.3 Markov Chain Monte Carlo (MCMC)

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 $\{\theta_j\}$ disampling dari sebuah distribusi peluang yang bergantung pada nilai sebelumnya $\{\theta_{j-1}\}$. Distribusi eksak dari $\{\theta_j\}$ umumnya tidak diketahui, namun distribusi pada setiap iterasi dalam barisan nilai sampel tersebut akan konvergen pada distribusi yang sesungguhnya untuk nilai j yang cukup besar. Oleh karena itu, jika ukuran sampel yang diperbaharui cukup besar maka kelompok terakhir dari nilai yang disampling dalam barisan tersebut, misal $\{\theta_{p+1}, \theta_{p+2}, \dots\}$ akan mendekati sebuah sampel yang berasal dari distribusi yang diinginkan^[3]. Notasi P biasanya disebut sebagai *burn in period*.

Terdapat dua algoritma utama dalam MCMC, yaitu algoritma Metropolis-Hastings dan algoritma Gibbs Sampling^[3]. Dalam tulisan ini, algoritma Metropolis-Hasting (MH) digunakan untuk membantu membangkitkan sampel-sampel acak dari distribusi posterior yang diinginkan. Dalam algoritma MH dibutuhkan sebuah *proposal distribution* $p(\theta | \theta_{j-1})$

untuk membangkitkan kandidat sampel acak. Langkah-langkah dasar dari algoritma ini adalah^[2]:

Langkah 1: Ambil nilai awal, yaitu θ_0 . Untuk iterasi $j = 1$,

$$\text{Bangkitkan } \theta^* \sim p(\theta | \theta_{j-1}).$$

Langkah 2: Bangkitkan sampel acak u dari distribusi uniform $U[0,1]$.

Langkah 3: Jika $u < \min\left(1, \frac{p(\theta^* | \mathbf{X}, \mathbf{y})p(\theta_{j-1} | \theta^*)}{p(\theta_{j-1} | \mathbf{X}, \mathbf{y})p(\theta^* | \theta_{j-1})}\right)$, ambil $\theta_j = \theta^*$. Namun jika

$$u > \min\left(1, \frac{p(\theta^* | \mathbf{X}, \mathbf{y})p(\theta_{j-1} | \theta^*)}{p(\theta_{j-1} | \mathbf{X}, \mathbf{y})p(\theta^* | \theta_{j-1})}\right), \text{ ambil } \theta_j = \theta_{j-1}$$

Langkah 4: Ulangi langkah 1-3 sampai dengan jumlah yang diinginkan.

Statistik yang digunakan untuk mengukur derajat ketergantungan antar pengambilan secara berurutan dalam sebuah rantai markov adalah autokorelasi. Autokorelasi mengukur korelasi antara dua buah himpunan dari nilai-nilai yang tersimulasi $\{\theta_j\}$ dan $\{\theta_{j+L}\}$ dengan L adalah lag atau bilangan yang memisahkan dua himpunan tersebut. Untuk sebuah hyperparameter θ_i tertentu, nilai autokorelasi pada lag ke- L dapat dihitung dengan formula

$$r_{iL} = \left(\frac{M}{M-L} \right) \left(\frac{\sum_{i=1}^{M-L} (\theta_i - \bar{\theta})(\theta_{i+L} - \bar{\theta})}{\sum_{i=1}^M (\theta_i - \bar{\theta})^2} \right)$$

dengan M adalah ukuran sampel acaknya.

3. Data dan Metode

3.1 Sumber Data

Eksperimen berikut ini diberikan untuk memperjelas implementasi MCMC pada pendugaan hyperparameter regresi proses Gaussian. Data yang digunakan adalah data sekunder yang berasal dari Bank Indonesia (BI) dari periode Juni 2003 sampai dengan Mei 2005. Data diperoleh melalui situs www.bi.go.id. Variabel-variabel yang diperhatikan adalah nilai kurs rupiah terhadap dolar USA (Y) dan tingkat suku bunga (X). Dalam hal ini X dianggap mempengaruhi nilai Y .

3.2 Metode Analisis

Langkah-langkah dalam pemodelan regresi proses Gaussian secara garis besar adalah sebagai berikut:

1. Menguji distribusi dari peubah tak bebas (nilai tukar rupiah).
2. Menetapkan jenis fungsi peragam yang akan digunakan.
Dalam makalah ini dipilih fungsi peragam eksponensial kuadrat yang formulanya seperti pada persamaan (3).
3. Memilih distribusi prior bagi hyperparameter.
Distribusi prior yang dipilih telah dijelaskan pada bagian 3 dari makalah ini.
4. Menduga nilai-nilai hyperparameter proses Gaussian.
Hyperparameter proses Gaussian diduga dengan menggunakan simulasi MCMC dengan menggunakan algoritma MH. Dalam makalah ini *proposal distribution* yang

dipilih adalah Uniform (0,1). Sedangkan nilai awal bagi hyperparameter θ adalah (1, 2, 0.5). Menurut Johson dan Albert^[5] serta Gamerman^[3], karena *proposal distribution* yang dipilih merupakan sebaran yang simetrik maka peluang penerimaan dari titik kandidat untuk terpilih menjadi anggota sampel acak merupakan perbandingan antara nilai posterior dari titik kandidat dengan nilai posterior dari sampel acak sebelumnya.

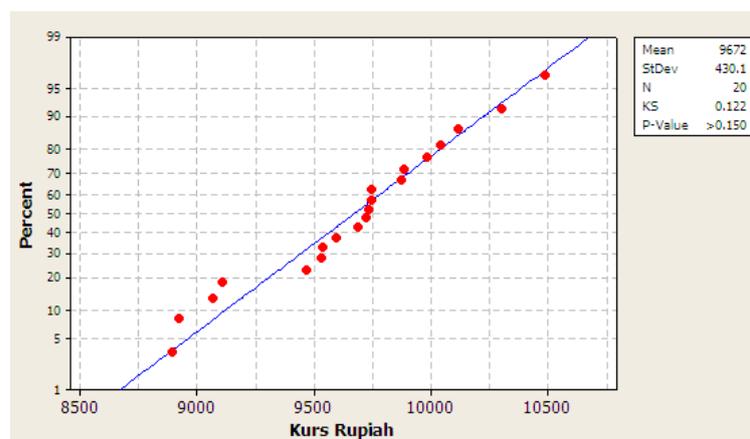
Peluang tersebut adalah $\min\left(1, \frac{p(\theta^*|\mathbf{X}, \mathbf{y})}{p(\theta_{j-1}|\mathbf{X}, \mathbf{y})}\right)$.

5. Memprediksi nilai tukar rupiah dan menentukan variasi dari hasil prediksi.
Prediksi terhadap nilai tukar rupiah dilakukan dengan menggunakan formula yang merupakan kombinasi antara formula pada persamaan (7) dan (8) dengan teknik prediksi berdasarkan sampel-sampel acak Monte Carlo. Formula tersebut adalah $E(y^*) = \frac{1}{M} \sum_{m=1}^M (E_m(y^*))$ dan $\text{var}(y^*) = \frac{1}{M} \sum_{m=1}^M \text{var}_m(y^*) + \frac{1}{M} \sum_{m=1}^M (E_m(y^*) - E(y^*))^2$ dengan $E_m(y^*)$ dan $\text{var}_m(y^*)$ dihitung berdasarkan persamaan (7) dan (8). M adalah ukuran sampel acak yang digunakan untuk memprediksi.
6. Menguji distribusi dari peubah acak galat..
7. Menghitung nilai RMSE (*Root Mean Square Error*).

Proses komputasi dari model ini dikerjakan dengan menggunakan program R. Listing programnya dapat dilihat di lampiran.

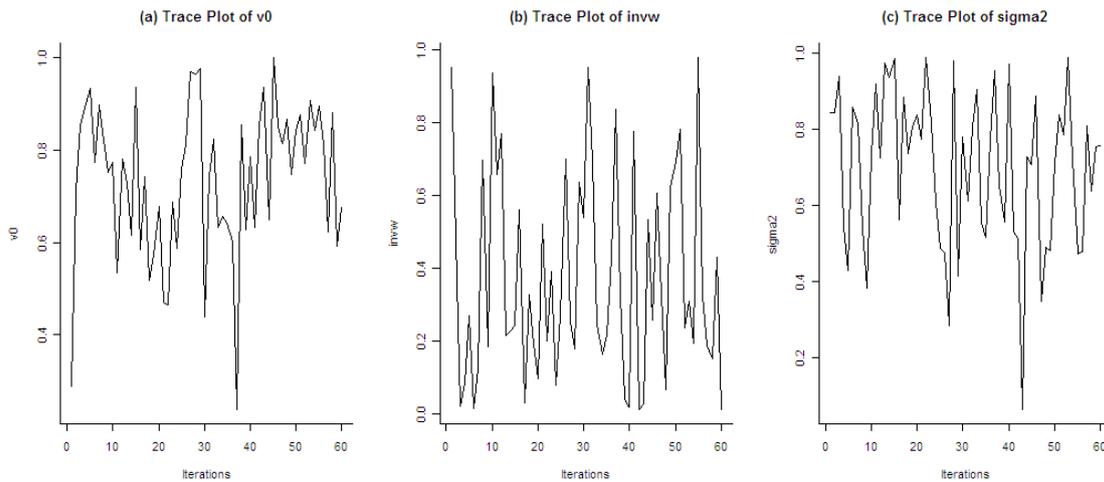
4. Hasil dan Pembahasan

Terdapat beberapa asumsi pada pemodelan dengan menggunakan proses Gaussian. Asumsi yang pertama adalah bahwa peubah tak bebas (nilai kurs rupiah terhadap dolar USA) dan galat mengikuti distribusi Gaussian (normal). Asumsi yang kedua adalah bahwa sampel acak yang diperoleh saling bebas. Berdasarkan Gambar 1, dengan menggunakan uji Kolmogorov-Smirnov pada taraf nyata 5%, dapat disimpulkan bahwa nilai kurs rupiah terhadap dolar USA menyebar normal dengan rata-rata Rp. 9672,- dan simpangan baku Rp. 430,-. Nilai p-value yang diperoleh pada pengujian sebaran ini lebih besar dari 0,150.



Gambar 1. Plot Peluang Normal Kurs Rupiah Terhadap Dolar USA

Simulasi MCMC dengan menggunakan algoritma Metropolis-Hasting dilakukan untuk menduga nilai hyperparameter $\theta = (v_0, w^{-1}, \sigma_\epsilon^2)$. Secara keseluruhan simulasi ini dilakukan dengan iterasi sebanyak 50.000 kali.

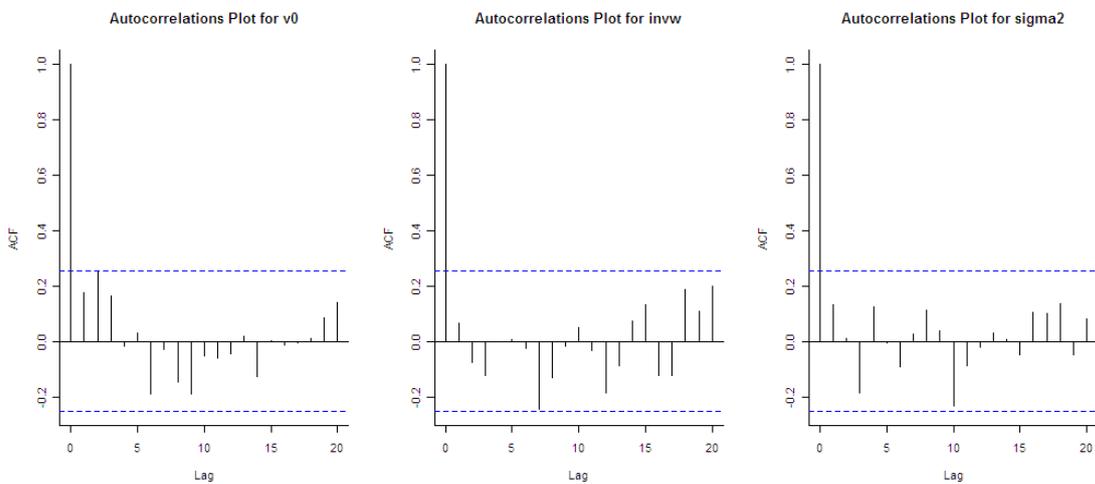


Gambar 2. Trace Plot Sampel Acak yang Terpilih dari Hyperparameter $\theta = (v_0, w^{-1}, \sigma_\epsilon^2)$

Nilai *burn in period* dalam makalah ini ditetapkan 20.000. Angka 20.000 ini dianggap cukup besar. Artinya sampel acak yang terpilih pada iterasi ke 20.001 dan seterusnya dapat diasumsikan berasal dari distribusi posterior $p(\theta|\mathbf{X}, \mathbf{y})$.

Agar sampel-sampel acak yang dipilih saling bebas antara satu dengan lainnya maka pemilihan sampel dilakukan dengan menggunakan *systematic sampling*. Dalam makalah ini sampel yang dipilih adalah setiap 500 iterasi setelah proses *burn in period*. Sehingga diperoleh sampel acak yang berukuran 60. Gambar 2 adalah trace plot sampel acak yang terpilih. Tampak bahwa trace plot dari masing-masing parameter tidak membentuk pola tertentu. Ini adalah sebuah indikasi bahwa sampel-sampel acak tersebut berasal dari sebaran posteriornya.

Selanjutnya fungsi autokorelasi digunakan untuk mengidentifikasi apakah sampel acak yang diperoleh tersebut dapat dianggap bebas diantara anggotanya. Gambar 3 berikut ini adalah plot fungsi autokorelasi dari lag 1 sampai dengan lag 20 untuk masing-masing hyperparameter. Tampak bahwa setelah lag pertama nilai korelasi antar variabel relative kecil. Artinya setelah lag pertama sampel-sampel acak yang terpilih untuk masing-masing parameter dapat diasumsikan bebas antara satu dengan yang lainnya.



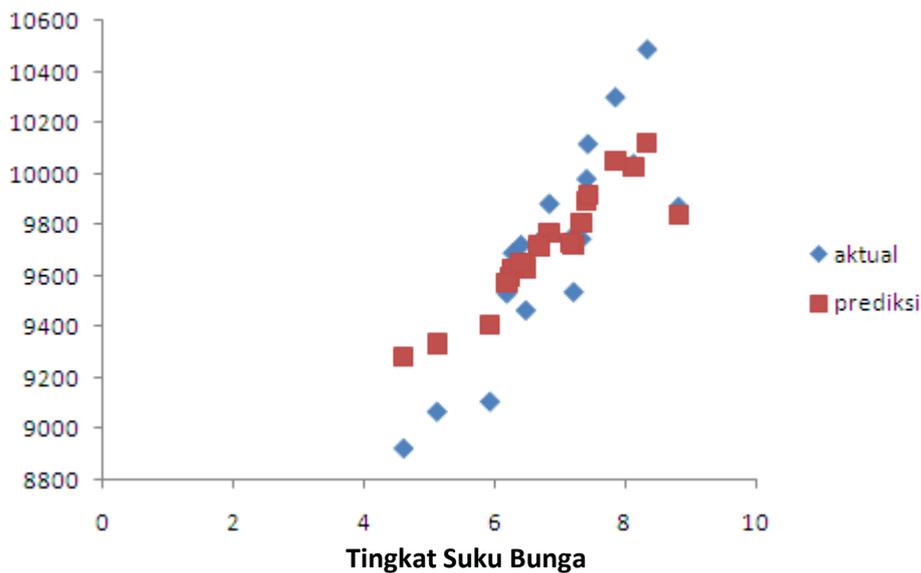
Gambar 3. Fungsi Autokorelasi Sampel-Sampel Acak dari Hyperparameter $\theta = (v_0, w^{-1}, \sigma_\epsilon^2)$

Sampel yang terpilih digunakan untuk menduga nilai hyperparameter. Nilai dugaan untuk hyperparameter $\theta = (v_0, w^{-1}, \sigma_\varepsilon^2)$ diperoleh dengan mengambil nilai rata-rata dari sampel acak tersebut. Tabel 1 berikut adalah ringkasan statistik bagi hyperparameter $\theta = (v_0, w^{-1}, \sigma_\varepsilon^2)$.

Tabel 1. Ringkasan Statistik Bagi Hyperparameter $\theta = (v_0, w^{-1}, \sigma_\varepsilon^2)$

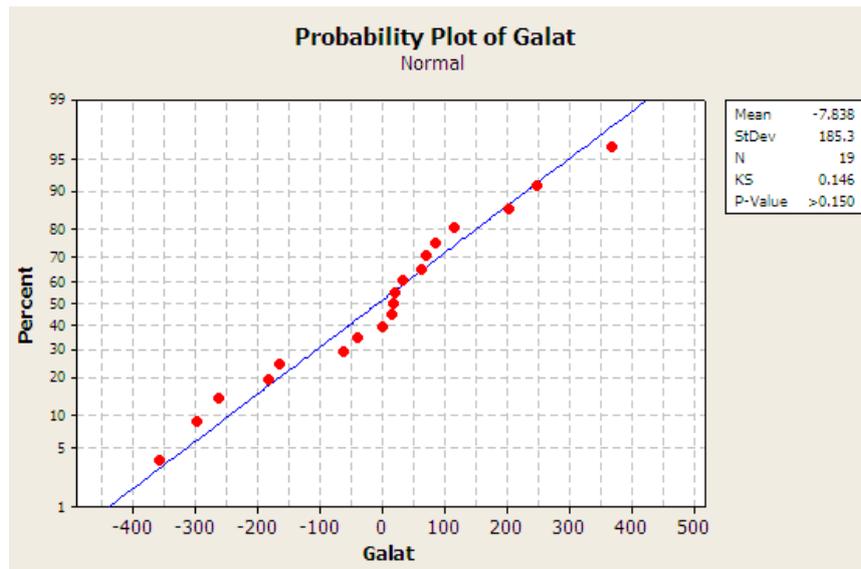
Statistik	Hyperparameter		
	v_0	w^{-1}	σ_ε^2
Minimum	0,2387	0,0126	0,0637
Kuantil 1	0,6301	0,1749	0,5388
Median	0,7648	0,2705	0,7347
Rata-rata	0,7361	0,3725	0,6991
Kuantil 3	0,8533	0,6099	0,8446
Maksimum	0,9988	0,9785	0,9888

Implementasi regresi proses Gaussian dengan menggunakan fungsi peragam eksponensial kuadrat untuk model prediksi nilai tukar rupiah terhadap dolar USA berdasarkan tingkat suku bunga Bank Indonesia menghasilkan RMSE sebesar 180,531. Plot antara nilai aktual dan prediksinya dapat dilihat di Gambar 4 berikut.



Gambar 4. Plot Antara Nilai Aktual dan Dugaan Nilai Tukar Rupiah

Selanjutnya untuk menjamin bahwa pemodelan nilai tukar rupiah dengan menggunakan proses Gaussian sah secara statistik, perlu dilakukan uji terhadap sebaran galat yang dihasilkan. Berdasarkan Gambar 4, dengan menggunakan uji Kolmogorov-Smirnov pada taraf nyata 5%, dapat disimpulkan bahwa peubah acak galat menyebar normal. Nilai p-value yang diperoleh pada pengujian sebaran ini lebih besar dari 0,150.



Gambar 5. Plot Peluang Normal Peubah Acak Galat

5. Kesimpulan

Regresi proses Gaussian adalah salah satu metode yang dapat digunakan untuk melakukan pemodelan. Agar model ini dapat diimplementasikan maka hyperparameter dari fungsi kovariannya harus diduga terlebih dahulu. MCMC adalah salah satu metode yang dapat digunakan dengan keunggulannya terletak pada performanya yang tidak sensitif pada penggunaan nilai awal. Salah satu algoritma MCMC yang sering digunakan adalah Metropolis-Hastings. Algoritma ini tidak terlalu rumit karena hanya memerlukan sebuah *proposal distribution* untuk membangkitkan sampel-sampel acak yang akan menjadi kandidat untuk diterima atau ditolak sebagai sampel dari distribusi posterior hyperparameter.

DAFTAR PUSTAKA

1. Casella G, Berger R.L., *Statistical Inference*, Thomson Learning, Duxbury, 2002.
2. Chen T, Morris J, Martin E., Gaussian Process Regression for Multivariate Spectroscopic Calibration, *Chemometrics and Intelligent Laboratory Systems*, 2007, 87: 85-97.
3. Gamerman D, *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*, Chapman & Hall, London, 1997.
4. Halim S, Bisono I., Fungsi-Fungsi Kernel Pada Metode Regresi Nonparametrik dan Aplikasinya Pada *Priest River Experimental Forest's Data*. *Jurnal Teknik Industri*, 2006, Vol.8, No.1. 73-81.
5. Johnson V.E., Albert J.H., *Ordinal Data Modeling*, Springer-Verlag, New York, 1999.
6. MacKay D.J.C., *Introduction to Gaussian Processes*. Di dalam: Bishop C.M., Editor. *Neural Networks and Machine Learning*, Springer-Verlag, New York, 1998.
7. Mukid M.A., Pemodelan Regresi Proses Gaussian Menggunakan Fungsi Peragam Eksponensial Kuadrat. *Media Statistika*, 2010, Vol.3, No.1:1-7.
8. Neal R.M., *Bayesian Learning For Neural Network*. New York, Springer-Verlag, New York, 1995.
9. Rasmussen C.E., Evaluation Of Gaussian Processes And Other Methods For Non-Linear Regression, *Disertasi Department of Computer Science, University of Toronto*, 1996.

10. Rasmussen C.E., Williams C.K.I., *Gaussian Process for Machine Learning*, MIT Press, Massachusetts, 2006.
11. Takezawa K., *Introduction to Nonparametric Regression*, John Wiley & Sons, New Jersey, 2006.
12. Williams C.K.I, Rasmussen C.E., Schwaighofer A, and Tresp V., *Observations on the Nyström Method for Gaussian Process Prediction*, Technical Report, University of Edinburgh, 2002.

Lampiran: Kode Program R Untuk Proses Gaussian Menggunakan MCMC

```
library(fields)
```

```
library(base)
```

```
## Data
```

```
x<-c(4.6,5.11,5.92,6.47,6.83,7.2,6.67,6.27,6.22,6.18,6.4,7.32,7.15,8.81,8.12,7.4,7.42,7.84,8.33)
```

```
z<-c(8925.2,9068.8,9108.3,9465.3,9882.4,9536.9,9735.4,9689.5,9596.2,9531.5,9721,9744.9,  
9744.9,9870.5,10039.4,9979.8,10116.5,10299.3,10486.2)
```

```
y<-z-mean(z)
```

```
iterations<-50000
```

```
hyper<-matrix(nrow=iterations, ncol=3)
```

```
## Setting Nilai Awal
```

```
current.hyper<-c(1,2,0.5)
```

```
current.v0<-current.hyper[1]
```

```
current.invw<-current.hyper[2]
```

```
current.sigma2<-current.hyper[3]
```

```
## Matrik Varian Kovarian
```

```
K01<-current.v0*Exp.cov(x, theta=current.invw, p=2, C=NA, marginal=FALSE)
```

```
K1<-K01+jitter(K01,1)
```

```
plus1<-current.sigma2*diag(19)
```

```
## Current
```

```
likelihood<--0.5*t(y)%*%solve(K1+plus1)%*%y
```

```
p.v0<-exp(-0.5*(current.v0+1)^2)
```

```
p.invw<-current.invw^(-3.5)*exp(-0.5*current.invw^(-1))
```

```
p.sigma2<-current.sigma2^(-3.5)*exp(-current.sigma2^(-1))
```

```
A<-p.v0*p.invw*p.sigma2
```

```
## Proposal
```

```
for (t in 1:iterations){
```

```
  prop.hyper<-runif(3,0,1)
```

```
  prop.v0<-prop.hyper[1]
```

```
  prop.invw<-prop.hyper[2]
```

```
  prop.sigma2<-prop.hyper[3]
```

```
  K02<-prop.v0*Exp.cov(x, theta=prop.invw, p=2, C=NA, marginal=FALSE)
```

```
  K2<-K02+jitter(K02,1)
```

```
  plus2<-prop.sigma2*diag(19)
```

```
  prop.likelihood<--0.5*t(y)%*%solve(K2+plus2)%*%y
```

```
  prop.p.v0<-exp(-0.5*(prop.v0+1)^2)
```

```
  prop.p.invw<-prop.invw^(-3.5)*exp(-0.5*prop.invw^(-1))
```

```
  prop.p.sigma2<-prop.sigma2^(-3.5)*exp(-prop.sigma2^(-1))
```

```
  B<-prop.p.v0*prop.p.invw*prop.p.sigma2
```

```
  p<-B*exp(prop.likelihood-likelihood)/A
```

```
  if(p!='NaN'){
```

```
    acc.prob<-min(1,p)
```

```

u<-runif(1)
  if(u<acc.prob){
    current.hyper<-prop.hyper
  }
  hyper[t,]<-current.hyper
  }}
t<-seq(20001,50000,500)
data<-hyper[t,]
new.hyper<-na.omit(data)

```

Trace Plot

```

par( mfrow=c(1,3), xaxs='r', yaxs='r', bty='l')
plot(new.hyper[,1], type='l', ylab='v0', xlab='Iterations', main='(a) Trace Plot of v0')
plot(new.hyper[,2], type='l', ylab='invw', xlab='Iterations', main='(b) Trace Plot of invw')
plot(new.hyper[,3], type='l', ylab='sigma2', xlab='Iterations', main='(c) Trace Plot of sigma2')

```

Plot ACF

```

par( mfrow=c(1,3), xaxs='r', yaxs='r', bty='l')
acf(new.hyper[,1], lag.max=20, main='Autocorrelations Plot for v0')
acf(new.hyper[,2], lag.max=20, main='Autocorrelations Plot for invw')
acf(new.hyper[,3], lag.max=20, main='Autocorrelations Plot for sigma2')

```

Prediksi

```

baris<-19
yduga<-matrix(nrow=baris)
variany duga<-matrix(nrow=baris)

for (j in 1:baris){
  g<-x[j]
  r<-nrow(new.hyper)
  yduga1<-matrix(nrow=r)
  variany duga1<-matrix(nrow=r)
  for (h in 1:r){
    v0<-new.hyper[h,1]
    invw<-new.hyper[h,2]
    sigma2<-new.hyper[h,3]
    k<-v0*Exp.cov(g,x, theta=invw, p=2, C=NA, marginal=FALSE)
    plus<-sigma2*diag(19)
    K<-v0*Exp.cov(x, theta=invw, p=2, C=NA, marginal=FALSE)
    yprediksi<-k%*%solve(K+plus)%*%y
    yduga1[h]<-yprediksi
    kappa<-v0*Exp.cov(g, theta=invw, p=2, C=NA, marginal=FALSE)
    varianprediksi<-kappa+sigma2-k%*%solve(K+plus)%*%t(k)
    variany duga1[h]<-varianprediksi
  }
  ratayduga<-mean(yduga1)
  selisih<-yduga1-ratayduga
  ratavariany duga<-mean(variany duga1)+mean(selisih^2)
  yduga[j]<-ratayduga
  variany duga[j]<-ratavariany duga
}

zduga<-yduga+mean(z)
error<-z-zduga
rmse<-sqrt(mean(error^2))

```