Optimalisasi Pembangkitan Listrik Pada PLTG pada PT. Indonesia Power Up Grati Pasuruan

Hanif Ryanas Putra, Yusuf Bilfaqih
Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember (ITS)
E-mail: yusufbf@gmail.com

***Abstrak*—Pada PLTG PT. Indonesia Power Unit Pembangkitan (UP) Grati, daya terbangkit yang tercatat sebesar 3x100MW. Pada Pembangkit Listrik Tenaga Uap (PLTG) umumnya menggunakan bahan baku utama yaitu gas alam sebagai pendorong turbin gas dengan tekanan tinggi. Dengan begitu maka dimanfaatkan untuk dapat menggerakkan turbin hingga listrik pun terbangkit. Untuk meningkatkan efisiensi penggunaan gas alam pada pembangkit PT. Indonesia Power UP Grati ini diperlukan adanya metode untuk meningkatkan efiensinya dengan menimalkan gas alam yang digunakan dengan daya yang terbangkit mendekati daya yang tercatat yaitu maksimal sebesar 100MW untuk masing-masing unitnya. Pada studi ini dilakukan penelitian tentang penerapan algoritma PSO (*Particle Swarm Optimization*) pada identifikasi pemodelan sistem. Dan penerapan kontroler LQG (*Linear Quadratic Gaussian*) pada proses optimalisasi sistem tersebut. Hasil dari pengujian dapat dilihat bahwa penerapan algoritma PSO dapat menghasilkan nilai *fitness* model yang lebih baik yaitu sebesar 0,6901 atau 69,01%. Dan juga hasil LQG dapat meningkatkan efisiensi pada sistem ini.**

***Kata Kunci*—Simulasi *plant*, optimalisasi, PLTG, identifikasi, PSO, LQG.**

I. PENDAHULUAN

TURBIN gas (GT) diketahui memiliki biaya modal yang rendah untuk rasio daya, fleksibilitas tinggi, keandalan yang tinggi. Oleh karena itu, turbin gas sangat kompatibel untuk digunakan sebagai pembangkit tenaga listrik. Pada PLTG PT. Indonesia Power UP Grati Pasuruan mempunyai 3 unit turbin gas dengan daya terbangkit yang tercatat sebesar 100MW untuk tiap unitnya. Sehingga biasa disebut dengan PLTG berkapasitas 3x100 MW. Dalam rangka peningkatan efiensi dari PLTG dengan meminimalkan penggunaan gas alam serta nilai daya yang terbangkit dapat mendekati dari daya yang tercatat.

Dengan menggunakan metode identifikasi sistem didapatkan prediksi nilai parameter dari *flowrate* atau aliran gas terhadap keluaran dari daya yang terbangkit.[1] Setelah itu, didapatkan persamaan pendekatan sistemnya dengan prediksi tersebut. Untuk mengoptimalkan pendekatan sistemnya ditambahkan algoritma PSO (*Particle Swarm Optimatimization*) sehingga pendekatan pemodelan lebih akurat. Setelah didapatkan pendekatan modelnya digunakan metode kontrol optimal LQG (*Linear Quadratic Gaussian)* untuk meminimalkan *cost function* pada sistem ini. Sehingga efisiensi pada pembangkit ini dapat meningkat.

II. IDENTIFIKASI SISTEM

Merupakan proses pengamatan suatu sistem dan menentukan parameter-parameter yang dimiliki sistem tersebut. Identifikasi dilakukan dengan cara mengamati respon sistem terhadap suatu masukan tertentu

Secara umum pendekatan sistem dilakukan menggunakan model matematis seperti pada persamaan 1.[2]

$$\mathbf{y(k)} = \frac{\mathbf{B(q^{-1})}}{\mathbf{A(q^{-1})}}\mathbf{u(k)} + \frac{\mathbf{q^{-d}C(q^{-1})}}{\mathbf{A(q^{-1})}}\boldsymbol{\eta}\mathbf{(k)} \quad (1)$$

Di mana :
d (faktor *delay*)
$B(q^{-1}) = b_1 q^{-1} + \ldots + b_{nb} q^{-nb}$ (parameter masukan)
$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_{na} q^{-na}$ (parameter keluaran)
$C(q^{-1}) = c_0 + c_1 q^{-1} + \ldots + c_{nc} q^{-nc}$ (parameter noise)

$q^{-1}y(k) = y(k-i); q^{-1}u(k) = u(k-i)$

Dengan menggunakan model ARX (*Auto Regressive eXogenous input*) maka struktur modelnya seperti pada persamaan 2 berikut ini.

$$\mathbf{y(k)} = \frac{\mathbf{b_0}}{\mathbf{A(q^{-1})}}\mathbf{u(k)} + \frac{\mathbf{q^{-d}C(q^{-1})}}{\mathbf{A(q^{-1})}}\boldsymbol{\eta}\mathbf{(k)} \quad (2)$$

Pada struktur ini $c_0 \neq 0$; $c_i = 0$ untuk i = 1,2,3,.; $a_j \neq 0$ untuk j = 1,2,3,..; $b_k \neq 0$ untuk k = 1,2,3,.. Dengan demikian bentuk umum dari struktur ini adalah seperti pada persamaan 2.3 – 2.4 berikut.

$$\mathbf{y(k)} + \mathbf{a_1 y(k-1)} + \cdots + \mathbf{a_{na} y(k-na)} = \mathbf{b_1 u(k-1)} + \ldots + \mathbf{b_{nb} u(k-nb)} + \mathbf{c_0}\boldsymbol{\eta}\mathbf{(k)} \quad (3)$$

$$y(k) = -a_1 y(k-1) - \cdots - a_{na} y(k-na) + b_1 u(k-1) + \ldots + \mathbf{b_{nb} u(k-nb)} + \mathbf{c_0}\boldsymbol{\eta}\mathbf{(k)} \quad (4)$$

Maka dengan tambahan algoritma PSO pada pemodelan menggunakan ARX seperti pada Gambar 1 berikut.

```
for setiap partikel
      Inisialisasi partikel
end

repeat
    for setiap partikel
       Hitung nilai fitness
         if nilai fitness baru lebih
baik
            daripada nilai fitness lama

             Update nilai fitness
      partikel

         end
    end

    Pilih partikel dengan nilai fitness
    terbaik diantara semua partikel
    tetangganya dan simpan nilai
fitness
    tersebut

    for setiap partikel
       Hitung velocity partikel
       Update posisi partikel
    end

until (KriteriaBerhenti = true)
```

Gambar 1. *Pseudocode* Algoritma PSO

Perubahan nilai-nilai koefisien ini bersifat acak dengan syarat batas yang telah diberikan pada algoritma PSO. Setelah mendapatkan nilai acak tiap koefisien parameter maka hasilnya akan dievaluasi dengan nilai *fitness* nya. Koefisien parameter dengan nilai *fitness* terbaik akan disimpan hingga proses perhitungan berhenti. Proses *looping* PSO akan berhenti jika nilai *fitness* dari koefisien parameter telah melebihi nilai *fitness* yang diinginkan. Dalam hal ini, lebih dari nilai *fitness* hasil identifikasi ARX sebelumnya.

Untuk inisialisasi parameter yang dibutuhkan adalah sebagai berikut:

- Jumlah Partikel
  Jumlah partikel didefinisikan sebanyak sepuluh. Hal ini mengacu pada dasar teori yang menyebutkan bahwa nilai 10 partikel sudah cukup besar untuk mendapatkan hasil yang bagus.
- Dimensi Partikel
  Dimensi partikel didefiniskan sebanyak 6 buah dikarenakan sesuai dengan banyaknya parameter *lag* pada model ARX.
- Rentang nilai dari partikel
  Untuk rentang nilai bernilai acak dengan bilangan ± 0.02 untuk parameter *lag* output model dan ± 0.025 untuk parameter *lag* input model.
- *Vmax*
  Untuk kecepatan maksimal juga bernilai acak dengan bilangan ± 0.05.
- *Learning Rate*
  Untuk laju pembelajaran menggunakan yang umumnya digunakan dalam algoritma PSO yaitu $\varphi_1 = \varphi_2 = 2$.
- Iterasi
  Untuk iterasi peng-*update*-an posisi dibatasi hingga mencapai iterasi ke-25.
- Kondisi Berhenti
  Untuk kondisi berhentinya PSO ditentukan saat nilai *fitness* model PSO lebih besar dari nilai *fitness* model ARX.

Dengan program yang telah dibuat sesuai *pseudocode* pada Gambar 3.6 dan inisialisasi parameter yang telah ditentukan maka akan didapatkan model PSO dengan nilai *fitness* yang lebih bagus daripada model ARX.

Kemudian agar dapat dikontol menggunakan kontrol optimal maka model PSO diubah menjadi model *state-space* ditambah dengan *disturbance* sebagai *noise* sistem.

III. KONTROL OPTIMAL

Untuk mengoptmalisasi sistem pembangkitan maka dilakukan kontrol optimal pada sistem. Pada kali ini, menggunakan kontrol optimal LQG (*Linear Quadratic Gaussian*) dengan satuan waktu diskrit dengan *state* dinyatakan dalam persamaan 5 dan 6 berikut.

$$x_{k+1} = Ax_k + Bu_k + Gw_k \quad (5)$$
$$y_k = Cx_k + v_k \quad (6)$$

Di mana *filter* Kalman *steady state* ditulis dalam bentuk rekursi priori dengan persamaan 7 dan 8 berikut ini.

$$\hat{x}_{k+1}^- = Ax_k + Bu_k + AK_f(y_k - C\hat{x}_k) \quad (7)$$

atau rekursi priori

$$\hat{x}_{k+1} = Ax_k + Bu_k + AK_f[y_k - C(Ax_k + Bu_k)] \quad (8)$$

Di mana K_f adalah *gain* Kalman *steady-state*. Estimasi *state* priori x_k^- tidak menggunakan pengukuran y_k sedangkan rekursi posteriori $\hat{x}_k$ menggunakan pengukuran y_k.

Maka, kontrol *feedback* dapat dinyatakan pada persamaan 9 dan 10 berikut.

$$u_k = -K\hat{x}_k^- + r_k \quad (9)$$

atau

$$u_k = -K\hat{x}_k + r_k \quad (10)$$

Dengan catatan bahwa *gain filter diskrit* adalah $L = KA_f$.

Keuntungan pemilihan *feedback* posteriori adalah bahwa sinyal u_k dapat dihitung pada waktu $k-1$ sehingga memberikan waktu yang lebih banyak untuk melakukan komputasi.

Gain feedback K dan *gain observer* dapat dipilih agar memberikan perilaku yang cocok dari $(A-BK)$ dan $(A-LC)$. Dalam desain LQG digital, *gain feedback* dipilih menggunakan ARE LQR diskrit dan *gain observer* dipilih dengan menggunakan ARE filter Kalman diskrit.

Untuk persamaan desain *state feedback* LQR adalah seperti pada persamaan 11 dan 12 berikut.

$$0 = A^T PA - P + Q - A^T PB(R + B^T PB)^{-1} B^T P \quad (11)$$

$$K = (R + B^T PB)^{-1} B^T PA \quad (12)$$

Untuk persamaan desain *gain filter* Kalman adalah seperti pada persamaan 13 dan 14 berikut.

$$0 = APA^T - P + GQ_f G^T - APC^T(CPC^T + R_f)^{-1} CPA^T \quad (13)$$

$$L = APC^T(CPC^T + R_f)^{-1} \quad (14)$$

Di mana Q_f dan R_f adalah matriks kovarian *noise*. Untuk desain regulator dinamik LQG adalah seperti pada persamaan 15 dan 16 berikut.

$$\hat{x}_{k+1}^- = Ax_k^- + Bu_k + L(y_k - C\hat{x}_k^- \quad (15)$$

$$u_k = -K\hat{x}_k + r_k \quad (16)$$

IV. HASIL DAN SIMULASI DATA

Pada tahap ini hasil identifikasi pemodelan sistem disimulasikan untuk validasi error yang didapatkan. Error merupakan selisih antara hasil keluaran PLTG berupa daya MW dengan hasil simulasinya. Setelah didapatkan persamaan model diskritnya seperti pada persamaan 17 – 19 berikut.

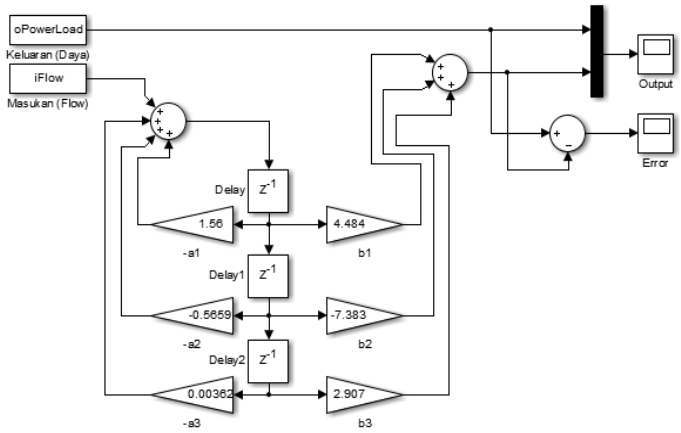
$$A(z)y(t) = B(z)u(t) + e(t) \quad (17)$$

Di mana,

$$A(z) = 1 - 1.56\,z^{-1} - 0.5659\,z^{-2} - 0.00362\,z^{-3} \quad (18)$$

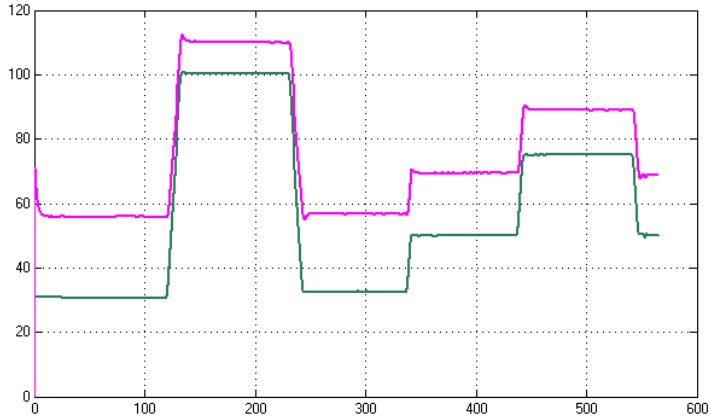
$$B(z) = 4.484\,z^{-1} - 7.383\,z^{-2} + 2.907\,z^{-3} \quad (19)$$

Maka, dirancang simulasinya pada *Simulink* seperti pada Gambar 2 berikut.



Gambar 2. Rancangan *Simulink* Model ARX

Pada Gambar 2, nilai dari enam parameter model diubah menjadi *gain* sebagai pengali sesuai dengan faktor *delay* dari masing-masing parameternya. Rancangan ini dirancang dalam bentuk kanonik. Untuk data masukan dan keluaran digunakan dari *workspace* dengan menambahkan waktu sampelnya pada kolom pertama dari masing-masing data. Setelah itu hasil simulasi dapat dilihat pada Gambar 3 berikut.



Gambar 3. Hasil Simulasi Model ARX

Pada Gambar 3, respon yang berwarna ungu merupakan keluaran dari model ARX sedangkan yang berwarna hijau merupakan data keluaran pengukuran. Terlihat pada Gambar 4.2 terdapat selisih atau *error* dari keluaran model ARX dan data keluaran yang terukur.

Pada tahap ini algoritma PSO ditambahkan dalam proses identifikasi. Jadi nilai parameter yang telah didapatkan model ARX akan diacak menggunakan 10 partikel pada tiap parameternya. Setiap parameter ini disebut dengan dimensi jadi pada model ini terdapat 6 dimensi dengan total 60 partikel.

Kemudian program akan dijalankan setelah melakukan inisialisasi pada program. Program berjalan hingga nilai *fitness* didapatkan melebihi nilai *fitness* dari model ARX sebelumnya.

Setelah itu, hasil dari nilai *fitness* yang didapatkan sebesar 0.6901 atau 69,01%. Hasil nilai *fitness* ini lebih besar daripada nilai *fitness* pada model ARX.

Dengan persamaan modelnya adalah seperti pada persamaan 20–22 berikut.

$$A(z)y(t) = B(z)u(t) + e(t) \quad (20)$$

Di mana,

$$A(z) = 1 - 1.543\,z^{-1} - 0.5546\,z^{-2} - 0.01006\,z^{-3} \quad (21)$$

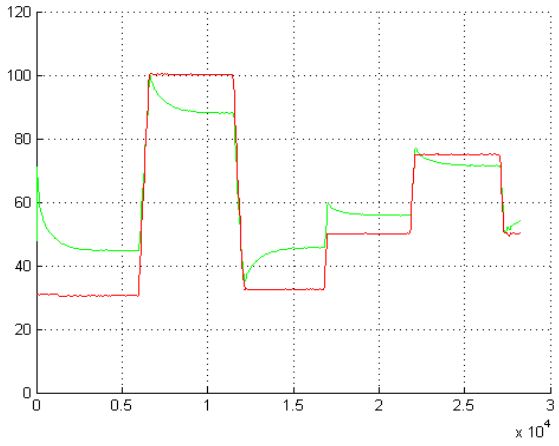
$$B(z) = 4.509\,z^{-1} - 7.357\,z^{-2} + 2.91\,z^{-3} \quad (22)$$

Kemudian model diubah menjadi persamaan *state-space* agar dapat dikontrol menggunakan kontrol optimal untuk selanjutnya. Setelah itu, model tambahan algoritma PSO diubah menjadi model *state-space* diskrit menjadi seperti pada persamaan 23 dan 24 berikut.

$$\dot{x} = \begin{bmatrix} 0 & 0 & -0.08047 \\ 0.125 & 0 & -0.5546 \\ 0 & 1 & 1.543 \end{bmatrix} x + \begin{bmatrix} 5.819 \\ -1.839 \\ 1.127 \end{bmatrix} u + \begin{bmatrix} -0.02012 \\ -0.1386 \\ 0.3857 \end{bmatrix} w \quad (23)$$

$$y = [0\ \ 0\ \ 4]\,x + [0]\,u \quad (24)$$

Pada Gambar 4, respon yang berwarna hijau merupakan keluaran dari model *state-space* dengan tambahan algortima PSO sedangkan yang berwarna merah merupakan data keluaran yang terukur.

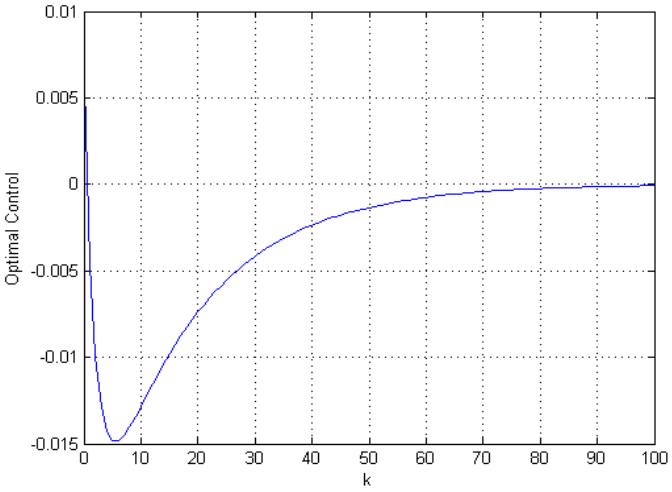


Gambar 4. Hasil Simulasi pada *State-Space* dengan PSO

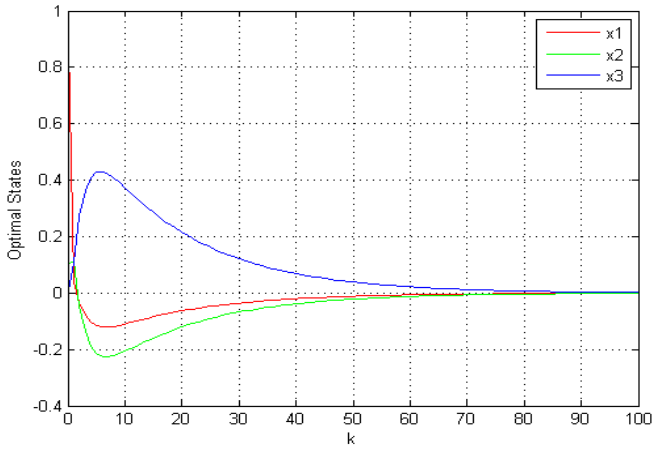
Selanjutnya untuk melihat perbandingan model PSO dengan model ARX sebelumnya. Pada Gambar 4 ditampilkan grafik dari kedua model tersebut. Jadi, model PSO lebih baik dikarenakan nilai *fitness*-nya sebesar 0,6901 atau 69,01% sedangkan untuk model ARX nilai *fitness*-nya sebesar 0,6712 atau 67,12%.

Kemudian Simulasi *plant* ini menggunakan model *state-space* PSO bukan pada model regresi dikrit karena nantinya matriks *state-space* digabungkan dengan kontroler LQR dengan tambahan *filter* Kalman yang berupa matriks *gain feedback* dan *gain* Kalman. Maka dari itu gabungan dari keduanya disebut dengan LQG.

Untuk hasil rancangan LQR adalah seperti pada Gambar 5 untuk hasil sinyal kontrol dan Gambar 4.6 untuk plot *state* sistem berikut ini.



Gambar 5. Hasil *Plot* Sinyal Kontrol dengan Kontroler LQR



Gambar 6. Hasil *Plot State* Optimal dengan Kontroler LQR

Terlihat pada Gambar 5 dan 6 hasil nilainya mendekati nol maka kontroler LQR telah berhasil diterapkan pada *state-space* sistem.

Kemudian didapatkan nilai *gain* Kalman (L) dan *gain feedback* (K). Berikut pada Gambar 4.18 nilai *gain* K dan L didapatkan dengan program *dare*.m pada Matlab.

Nilai *gain feedback* dan *gain* Kalman didapatkan dalam persamaan 25 dan 26 berikut.

$$K = [-0.008 \quad 0.00840 \quad 0.0367] \quad (25)$$

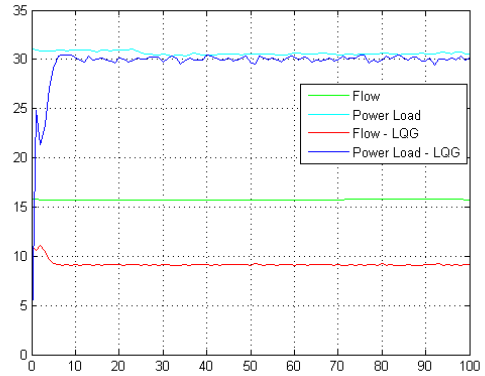
$$L = \begin{bmatrix} -0.0201 \\ -0.1349 \\ 0.6322 \end{bmatrix} \quad (26)$$

Dengan memasukkan semua nilai parameter seperti matriks A, B, C, D, G, K, L, N dan *noise variance* pada *simulink* LQG seperti pada Gambar 4.5 dan 4.6. Maka didapatkan nilai optimal sinyal kontrol dan keluaran. Hal ini nantinya dibandingkan dengan sinyal kontrol sebelum diberikan kontroler LQG ini atau pada model sebelumnya.

Nilai referensi atau *set point* dari LQG merupakan representasi dari nilai pengaturan daya keluaran (*power load*)

pada PLTG. Pada PLTG daya keluaran diatur sesuai dengan perintah P2B (Pusat Pengatur Beban) ke PLTG.

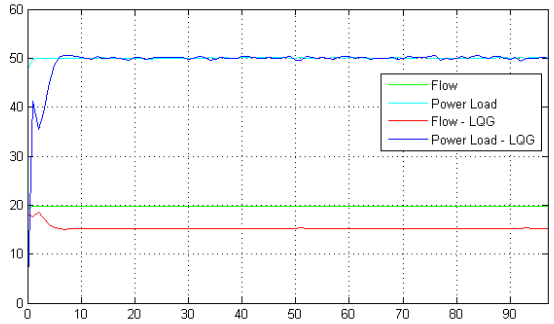
Hasil simulasi saat PLTG dengan referensi daya keluaran 30MW. Dikarenakan unit PLTG maksimal dapat menghasilkan 100MW maka daya yang terbangkit sebesar 30% dapat dilihat pada Gambar 4.7 berikut.



Gambar 7. Hasil Perbandingan Simulasi LQG dengan Kondisi Nyata PLTG 30MW

Pada Gambar 7 terlihat bahwa nilai simulasi dari LQG lebih efisien dikarenakan energi yang dibutuhkan dalam hal ini *flow* sebagai masukan sistem lebih sedikit dengan hasil *Power Load* yang melebihi. Namun hasil output berupa *Power Load* mengalami *overshoot* dan *osilasi* yang berlebih.

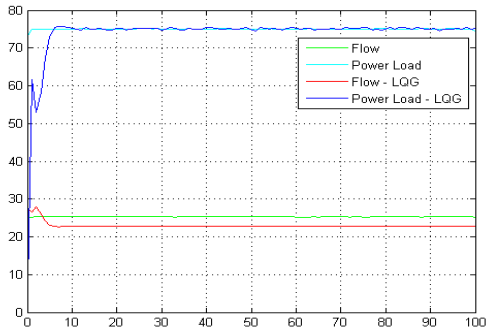
Hasil simulasi saat PLTG dengan referensi daya keluaran 50MW. Dikarenakan unit PLTG maksimal dapat menghasilkan 100MW maka daya yang terbangkit sebesar 50%. Untuk mengetahui perbandingannya dengan sistem sebelumnya dapat dilihat pada Gambar 8



Gambar 8. Hasil Perbandingan Simulasi LQG dengan Kondisi Nyata PLTG 50MW

Pada Gambar 8 terlihat bahwa nilai simulasi dari LQG lebih efisien dikarenakan energi yang dibutuhkan dalam hal ini *flow* sebagai masukan sistem lebih sedikit dengan hasil *Power Load* yang melebihi. Namun hasil *output* berupa *Power Load* mengalami *overshoot* dan *osilasi* yang berlebih.

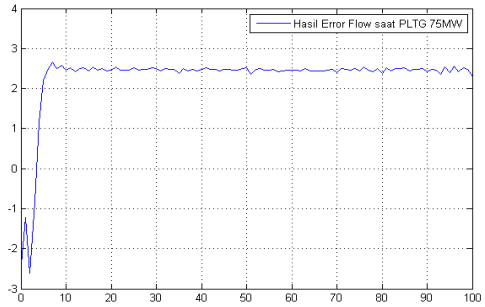
Hasil simulasi saat PLTG dengan referensi daya keluaran 75MW. Dikarenakan unit PLTG maksimal dapat menghasilkan 100MW maka daya yang terbangkit sebesar 75%. Untuk mengetahui perbandingannya dengan sistem sebelumnya dapat dilihat pada Gambar 9



Gambar 9. Hasil Perbandingan Simulasi LQG dengan Kondisi Nyata PLTG 75MW

Pada Gambar 9 terlihat bahwa nilai simulasi dari LQG lebih efisien dikarenakan energi yang dibutuhkan dalam hal ini *flow* sebagai masukan sistem lebih sedikit dengan hasil *Power Load* yang sesuai dengan *set point*.

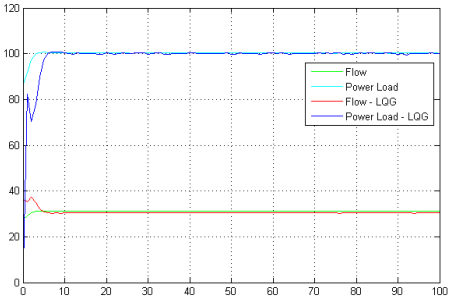
Dikarenakan hasil sinyal kontrol LQG dan nilai *Flow* berselisih sedikit maka pada Gambar 4.10 ditampilkan error atau selisih dari keduanya. Di mana nilai *Flow* dikurangkan dengan hasil sinyal kontrol LQG.



Gambar 10. Hasil *Error* pada nilai *Flow* dan sinyal kontrol LQG pada PLTG 75MW

Pada Gambar 10 terlihat bahwa nilai *error* sebagian besar bernilai positif maka dapat disimpulkan bahwa nilai sinyal kontol dari LQG lebih sedikit dibandingkan dengan nilai *Flow* sebenarnya.

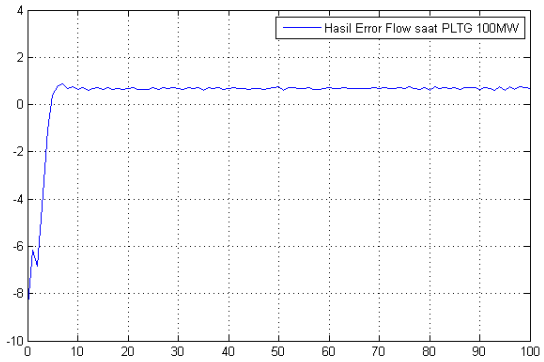
Hasil simulasi saat PLTG dengan referensi daya keluaran 100MW. Dikarenakan unit PLTG maksimal dapat menghasilkan 100MW maka daya yang terbangkit sebesar 100% atau dapat dikatakan dengan beban penuh. Untuk mengetahui perbandingannya dengan sistem sebelumnya dapat dilihat pada Gambar 11



Gambar 11. Hasil Perbandingan Simulasi LQG dengan Kondisi Nyata PLTG 100MW

Pada Gambar 11 terlihat bahwa nilai simulasi dari LQG lebih efisien dikarenakan energi yang dibutuhkan dalam hal ini *flow* sebagai masukan sistem lebih sedikit dengan hasil *Power Load* yang melebihi. Namun hasil output berupa *Power Load* mengalami *overshoot* dan *osilasi* yang berlebih.

Dikarenakan hasil sinyal kontrol LQG dan nilai *Flow* berselisih sangat sedikit maka pada Gambar 12 ditampilkan error atau selisih dari keduanya. Di mana nilai *Flow* dikurangkan dengan hasil sinyal kontrol LQG.



Gambar 12. Hasil *Error* pada nilai *Flow* dan sinyal kontrol LQG pada PLTG 100MW

Pada Gambar 12 terlihat bahwa nilai *error* sebagian besar bernilai positif maka dapat disimpulkan bahwa nilai sinyal kontol dari LQG lebih sedikit dibandingkan dengan nilai *Flow* sebenarnya.

V. KESIMPULAN

Dari penelitian yang telah dilakukan, dapat ditarik kesimpulan bahwa penerapan algoritma PSO (*Particle Swarm Intelligence*) pada identifikasi sistem dengan ARX memberikan hasil yang lebih baik dengan nilai *fitness* sebesar 0,6901 atau 69,01% yang sebelumnya mempunyai nilai *fitness* sebesar 0,6712 atau 67,12%

Dan juga penerapan kontroler LQG pada kontrol optimal memberikan hasil yang lebih efisien terhadap sistem karena dengan nilai masukan yang lebih sedikit dapat menghasilkan keluaran daya yang bernilai sama atau lebih. Dengan selisih masukan akan lebih kecil apabila nilai referensi semakin besar. Namun hasil keluarannya terjadi *overshoot* dan *osilasi* yang berlebih.

DAFTAR PUSTAKA

[1] T. Hongjin, Zhang; Yizuo, Zhang; Weidong, Zhang; Tao, “Swarm Intelligence Algorithm on Combustion Optimization of Coal-fired Boiler,” in *Chinese Control Conference*, 2016.

[2] K. Ogata, *Modern Control Engineering Third Edition*. Prentice-Hall International, Inc, 1997.