TUGAS AKHIR – TF 091381

PERANCANGAN SISTEM KONTROL *PITCH ANGLE* TURBIN ANGIN SKALA KECIL BERBASIS *IMPERIALIST COMPETITIVE ALGORITHM* (ICA)

AKHMAD BAKHRUL FAUZI
NRP. 2410 100 055

Dosen Pembimbing
Dr.Ir. Ali Musyafa', M.Sc.

JURUSAN TEKNIK FISIKA
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2014

FINAL PROJECT – TF 091381

THE DESIGN OF PITCH ANGLE CONTROL SYSTEM OF SMALL SCALE WIND TURBINE BASED ON IMPERIALIST COMPETITIVE ALGORITHM (ICA)

AKHMAD BAKHRUL FAUZI
NRP. 2410 100 055

Supervisor
Dr.Ir. Ali Musyafa', M.Sc.

DEPARTMENT OF ENGINEERING PHYSICS
Faculty of Industrial Technology
Sepuluh Nopember Institute of Technology
Surabaya 2014

LEMBAR PENGESAHAN

PERANCANGAN SISTEM KONTROL *PITCH ANGLE* TURBIN ANGIN SKALA KECIL BERBASIS *IMPERIALIST COMPETITIVE ALGORITHM* (ICA)

TUGAS AKHIR

Oleh:
AKHMAD BAKHRUL FAUZI
NRP. 2410 100 055

Surabaya, Juni 2014
Mengetahui/Menyetujui

Pembimbing

Dr.Ir. Ali Musyafa', M.Sc
NIP. 19600901 198701 1 001

Ketua Jurusan
Teknik Fisika, FTI-ITS

Dr.Ir. Totok Soehartanto,DEA
NIP. 19650309 199002 1 001

PERANCANGAN SISTEM KONTROL *PITCH ANGLE* TURBIN ANGIN SKALA KECIL BERBASIS *IMPERIALIST COMPETITIVE ALGORITHM* (ICA)

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
pada
Bidang Studi Rekayasa Instrumentasi dan Kontrol
Program Studi S-1 Jurusan Teknik Fisika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Oleh:

AKHMAD BAKHRUL FAUZI
NRP. 2410 100 055

Disetujui oleh Tim Penguji Tugas Akhir:

1. Dr. Ir. Ali Musyafa', M.Sc ..... (Pembimbing)
2. Ir. Ya'umar, MT ..... (Ketua Penguji)
3. Dr. Bambang L. Widjiantoro ,ST, MT ..... (Penguji I)
4. Dr. Ir. Purwadi A.D, M.Sc ..... (Penguji II)
5. Hendra Cordova, ST, MT ..... (Penguji III)
6. Irwansyah, ST, MT ..... (Penguji IV)

SURABAYA
JUNI, 2014

PERANCANGAN SISTEM KONTROL *PITCH ANGLE* TURBIN ANGIN SKALA KECIL BERBASIS *IMPERIALIST COMPETITIVE ALGORITHM* (ICA)

Nama Mahasiswa : Akhmad Bakhrul Fauzi
NRP : 2410 100 055
Jurusan : Teknik Fisika
Dosen Pembimbing : Dr.Ir. Ali Musyafa', M.Sc.

ABSTRAK

Sistem Konversi Energi Angin (SKEA) adalah salah satu energi alternatif yang dikembangkan dari pemanfaatan energi angin. Turbin angin skala kecil adalah SKEA atau turbin angin yang kapasitasnya kurang dari 10 kW. Indonesia memiliki kecepatan angin yang berubah-ubah dan relatif rendah. Untuk mengatasi permasalahan tersebut diperlukan sistem kontrol sudut pitch turbin angin. Pada penelitian ini dirancang sebuah sistem kontrol sudut pitch turbin angin berbasis Imperialist Competitive Algorithm (ICA). Sistem kontrol diharapkan pada kecepatan angin yang bervariasi, dapat memberikan kecepatan sudut rotor tetap stabil dan bekerja pada daerah optimal. ICA adalah sebuah evolusiner algoritma baru untuk mengoptimasi sebuah sistem yang diilhami dari kompetisi kekuasaan (Imperialist Competition). Optimasi berbasis ICA untuk sistem kontrol yang dirancang menggunakan ITAE (Integral Time Absolute Error) sebagai fungsi objektif dalam penentuan parameter kontrol PID. Hasil penelitian menujukkan kontrol PID berbasis ICA memberikan respon yang baik untuk semua uji setpoint, dan pada uji setpoint 10 pps memberikan hasil respon yang terbaik jika dibandingkan dengan setpoint yang lebih tinggi. Dengan Kp, Ki, dan Kd berurutan 1.49, 3.13 dan 0.03 memberikan lonjakan maksimal (maximum overshoot) 11.30 % dengan waktu turun (settling time) selama 1.85 detik dan secara kuantitatif nilai ITAE selama 10 detik yaitu 0.42.

Kata kunci: *Imperialist Competitive Algorithm (ICA), SKEA.*

THE DESIGN OF PITCH ANGLE CONTROL SYSTEM OF SMALL SCALE WIND TURBINE BASED ON *IMPERIALIST COMPETITIVE ALGORITHM* (ICA)

Sudent's Name : Akhmad Bakhrul Fauzi
NRP : 2410 100 055
Department : Engineering Physics FTI-ITS
Supervisor : Dr.Ir. Ali Musyafa', M.Sc.

ABSTRACT

Wind Energy Conversion Systems (WECS) is one of the alternative energy developed from wind energy utilization. Small-scale wind turbine is wind turbine which have capacity less than 10 kW. Indonesia has low wind speed that changes from time to time. To overcome this problem the control system of wind turbine pitch angle is required. In this study a control system of wind turbine pitch angle based on Imperialist Competitive Algorithm (ICA) is designed. The control system is expected that at varying wind speeds, it can provide rotor angular velocity remains stable and works at optimum area. ICA is a novel evolutionary algorithm to optimize a system inspired from competition rule (Imperialist Competition). ICA-based optimization control system is designed for use ITAE (Integral Time Absolute Error) as the objective function in determining the PID control parameters. The results showed that ICA-based PID control responds well to all setpoints tested, and the test results for setpoint of 10 pps provides the best response when compared others. For Kp, Ki, and Kd of 1.49, 3.13 and 0.03, respectively, it provides maximum overshoot of 11.30% with settling time of 1.85 seconds and ITAE value for 10 seconds is 0.42.

Keywords: *Imperialist Competitive Algorithm (ICA), WECS*

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat Allah SWT atas segala pertolongan, rahmat dan hidayah-Nya, serta sholawat dan salam kepada Nabi Muhammad SAW sehingga penulis dapat menyelesaikan Tugas Akhir dengan judul,

"PERANCANGAN SISTEM KONTROL *PITCH ANGLE* TURBIN ANGIN SKALA KECIL BERBASIS *IMPERIALIST COMPETITIVE ALGORITHM* (ICA)"

Tugas Akhir ini merupakan salah satu persyaratan akademik yang harus dipenuhi dalam Program Studi S-1 Teknik Fisika FTI-ITS. Penulis telah banyak mendapatkan bantuan dari berbagai pihak dalam menyelesaikan Tugas Akhir ini. Penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Kedua orang tua dan keluarga tercinta, Bapak San Rois, Ibu Mujiarti, Bapak Soleh Irsyad, Ibu Nur Ulfiyah, dan saudaraku yang telah memberikan doa, motivasi, dan nasehat dalam proses pengerjaan tugas akhir ini.
2. Bapak Dr. Ir. Ali Musyafa', M.Sc. selaku dosen pembimbing Tugas Akhir, yang telah memberikan bimbingan dan pengarahan dalam membantu penulis menyelesaikan Tugas Akhir ini.
3. Bapak Dr. Ir. Totok Soehartanto, DEA selaku Ketua Jurusan Teknik Fisika ITS, atas segala fasilitas pembelajaran yang diberikan di Jurusan Teknik Fisika ITS.
4. Ibu Dyah Sawitri, S.T. M.T. Selaku Dosen Wali, atas segala kebaikan, kesabaran, perhatian, bimbingan, dan arahan dalam proses perencanaan dan pembelajaran selama di Jurusan Teknik Fisika ITS.
5. Bapak Ir. Ya'umar, M.T selaku Kepala Bidang Studi Rekayasa Instrumentasi dan Kontrol Teknik Fisika FTI-ITS.
6. Bapak dan Ibu Dosen Teknik Fisika yang telah memberikan ilmu selam proses perkuliahan. Serta seluruh karyawan Teknik Fisika yang telah membantu dalam proses akademik.

7. Pramandita Ade Hutama. Selaku sahabat yang telah memberikan kebaikan berupa fasilitas kos selama proses menimba ilmu di Surabaya. Teman-teman satu kos, Hanif, Rowi, Widi, Gege, Atma, dan Rafdi di Perumdos ITS, Jl. Teknik Sipil Blok N/13 yang telah membantu dalam semangat dan doa.
8. Teman-teman seperjuangan Teknik Fisika angkatan 2010 yang telah banyak membantu selama proses perkuliahan dalam segala tindakan dan doa.
9. Teman-teman baikku, Mas Diaz, Dipta, dan Sena yang telah bersedia berdiskusi dan memberikan saran maupun kritik dan bantuan dalam proses Tugas Akhir ini.
10. Semua pihak yang telah membantu dalam kesuksesan Tugas Akhir yang tidak dapat disebutkan satu per satu.

Penulis menyadari bahwa kemampuan berfikir dan pengerjaan laporan Tugas Akhir ini tidaklah sempurna. Oleh karena itu diharapkan kritik dan saran yang membangun dari semua pihak untuk perbaikan dan pengembangan laporan penelitian ini. Penulis berharap semoga laporan ini dapat menambah wawasan yang bermanfaat bagi pembancanya dan bidang keilmuan.

Surabaya, Juni 2014

Penulis.

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR TABEL

"Halaman ini memang dikosongkan"

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi pembangkit listrik tenaga angin di Indonesia semakin pesat seiring akan adanya krisis energi dan upaya untuk melakukan usaha pengembangan energi alternatif. Pengembangan energi alternatif itu merupakan pengembangan energi listrik konvensional kearah energi listrik yang berbasis pada energi baru terbarukan (EBT**)**. Pemanfaatan energi angin di Indonesia selain sebagai energi baru terbarukan juga mempunyai peranan penting dalam pemenuhan jaringan listrik bagi daerah-daerah atau pedesaan yang belum terjangkau, terutama penggunaan pembangkit listrik tenaga angin berskala kecil yang ekonomis dan terjangkau untuk diaplikasikan. Pembangkit energi angin yang ekonomis artinya memiliki efisiensi yang tinggi dan biaya yang lebih rendah dibandingkan pembangkit alternatif lain saat ini.

Sistem Konversi Energi Angin (SKEA) terdiri dari turbin angin, generator, daya elektronik, sistem jaringan dan sistem kontrol. Turbin angin skala kecil adalah turbin angin yang kapasitasnya kurang dari 10 kW. Turbin ini cocok untuk Indonesia yang mempunyai kecepatan angin yang rendah dan berubah-ubah. Energi listrik yang dihasilkan oleh turbin angin bergantung pada daya mekanik yang bersumber dari masukan energi kinetik angin. Terdapat beberapa variabel yang mempengaruhi besaran daya mekanik turbin diantaranya adalah koefisisen daya (Cp). Nilai dari koefisien daya dipengaruhi oleh kecepatan angin, sudut *pitch*, dan kecepatan rotor dari turbin.

Agar mendapatkan daya yang optimal maka diperlukan sistem kontrol pada turbin angin. sistem kontrol yang dirancang yaitu sistem kontrol sudut *pitch* turbin angin. Sistem kontrol sudut *pitch* turbin adalah pengendalian sudut dari bilah turbin terhadap arah tiupan angin agar menghasilkan kecepatan sudut rotor yang tetap stabil dan berada pada kerja optimal.

Dalam Tugas Akhir ini dirancang sebuah sistem kontrol *pitch angle* berbasis *Imperialist Competitive Algorithm* (ICA). Metode ICA adalah sebuah metode evolusi optimasi terbaru yang diilhami dari algoritma kompetisi imperialis (Gargari.Atashpaz, 2007). Optimasi (ICA) mampu memberikan nilai parameter kontrol yang optimal dengan pencarian nilai parameter kontrol PID berupa nilai Kp, Ki, dan Kd optimal berbasis kompetisi imperialis.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka permasalahan yang diangkat pada penelitian Tugas Akhir ini antara lain:

- Bagaimana merancang sistem kontrol *pitch angle* turbin angin skala kecil berbasis *Imperialist Competitive Algorithm* (ICA)?
- Bagaimana performansi sistem kontrol ICA terhadap sistem kontrol *pitch angle* turbin angin skala kecil yang dirancang?

1.3 Batasan Masalah

Terdapat beberapa batasan masalah untuk penyelesaian masalah pada penelitian Tugas Akhir ini antara lain:

- Turbin yang digunakan bahan studi adalah prototipe turbin skala kecil yang dibuat oleh peneliti sebelumnya Farid R.M, pada tahun 2011, yaitu turbin dengan tipe bilah NREL S38n.
- Sistem kontrol *pitch angle* turbin angin dirancang berdasarkan kontrol kecepatan putar rotor turbin angin. kecepatan rotor yang maksimal diperoleh dengan sudut optimal pada kecepatan angin tertentu. Kecepatan angin yang digunakan adalah kecepatan angin pada rentang 2.8 m/s – 7.5 m/s.
- Objek yang dijadikan studi hanya pada bagian mekanik turbin angin
- Penelitian dilakukan dengan simulasi menggunakan Matlab 7.12.0 (R2011a).

- Batas parameter kontrol PID yang digunakan untuk implementasi optimasi ICA adalah nilai Kp (0.50 sampai1.50), nilai Ki (1.00 sampai 5.00), dan nilai Kd (0.01 sampai 0.10).

1.4 Tujuan

Berdasarkan latar belakang dan rumusan masalah diatas, tujuan dari penelitian Tugas Akhir ini adalah sebagai berikut:

- Merancang sistem kontrol pada pitch angle turbin angin skala kecil berbasis *Imperialist Competitive Algorithm* (ICA).
- Menganalisa performansi sistem kontrol pada *pitch angle* turbin angin skala kecil berbasis *Imperialist Competitive Algorithm* (ICA) yang telah dirancang.

"Halaman ini memang dikosongkan"

BAB II
DASAR TEORI

2.1 Sumber Energi Angin

Energi Angin adalah aliran angin yang disebabkan oleh perbedaan suhu antara dua tempat dengan kecepatan tertentu. Aliran angin terjadi ketika udara panas yang berada disuatu tempat menjadi lebih ringan dan naik keatas, sedangkan udara di tempat lain yang lebih dingin akan bergerak menuju ke tempat yang lebih panas, sehingga mengakibatkan terjadinya pergerakan atau perpindahan angin. Terdapat energi kinetik didalam energi angin yang dapat dikonversi menjadi energi mekanik. Proses selanjutnya energi mekanik akan dikonversi menjadi listrik melalui sistem generator.

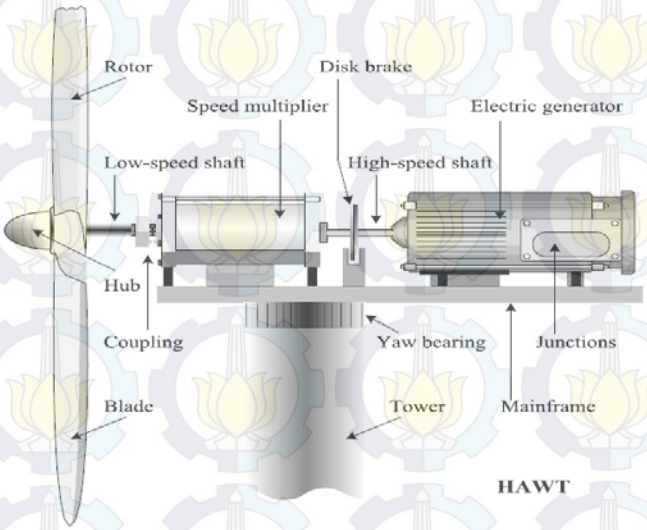
Energi angin adalah salah satu energi alternatif dan terbarukan yang perkembangannya sangat cepat. Energi angin dapat menjadi salah satu solusi dalam krisis energi akibat krisisnya sumber energi fosil dan merupakan sumber energi ramah lingkungan yang tidak menimbulkan polusi. Selain itu, energi angin merupakan energi yang berasal dari alam yang mudah didapatkan, berkelanjutan dan senantiasa tersedia di alam. Energi angin dapat dimanfaatkan sebagai sumber pembangkit listrik, menyalakan pompa air, membantu pengairan sawah atau lahan pertanian, penggilingan padi dan lain-lain. Energi angin ini juga dapat dimanfaatkan oleh daerah-daerah terpencil yang masih belum mendapatkan fasilitas jaringan listrik.

Pemanfaatan energi angin di Indonesia saat ini masih mencapai kurang lebih 1 MW daya terpasang dan direncanakan mencapai 250 MW pada tahun 2025 (P3TKEBT-ESDM). Berdasarkan kecepatan angin yang berada di Indonesia, potensi sumber energi angin yang dapat dimanfaatkan dibagi menjadi tiga kelas tenaga angin yaitu kelas skala kecil dengan kecepatan angin 2.5-4.0 m/detik, kelas skala menengah dengan kecepatan angin 4-5 m/s dan kelas skala besar dengan kecepatan lebih dari 5 m/detik. Variasi kecepatan ini tersebar di berbagai wilayah antara lain Jawa, NTB, NTT, Sulawesi, dan Maluku (ESDM-2010).

Energi angin juga memiliki kelemahan diantaranya perbedaan kecepatan angin diberbagai tempat di Indonesia. Di beberapa tempat terdapat angin dengan kecepatan yang kencang sehingga pemanfaatan energi angin menjadi lebih mudah, sedangkan di beberapa tempat lainnya kecepatan anginnya tidak cukup kuat sehingga tidak mampu dimanfaatkan dengan baik. Selain itu biaya instalasi dan teknologi yang belum maksimal mengakibatkan pemanfaatan energi angin ini belum maksimal. Sehingga, pengembangan dan penelitian mengenai pemanfaatan energi angin sangat diperlukan untuk meningkatkan efektifitas dan efisiensi energi angin. Akibatnya turbin angin menjadi energi alternatif dan terbarukan dan memberikan manfaat untuk masyarakat.

2.2 Turbin Angin

Turbin angin adalah suatu alat atau perangkat yang mengkonversi energi angin menjadi energi mekanik dan selanjutnya akan dikonversi menjadi energi listrik. Konversi energi ini melalui dua tahap. Tahap pertama yaitu proses ektraksi angin, proses ini menggunakan perangkat *blade* atau rotor turbin yang akan menangkap dan mengubah energi kinetik angin menjadi daya mekanik. Tahap selanjutnya poros putar (*shaft*) yang berada dibelakang *blade* turbin terhubung hingga melalui bagian *gearbox*, poros putar keluaran *gearbox* akan memutar generator elektrik sehingga menghasilkan energi listrik. Bagian *gear box* ini berfungsi untuk mengubah kecepatan poros putar kecepatan rendah (*low speed shaft*) menjadi lebih tinggi kecepatan poros putarnya (*high speed shaft*) untuk memutar generator. Komponen turbin angin secara lebih rinci ditunjukkan oleh gambar 2.1



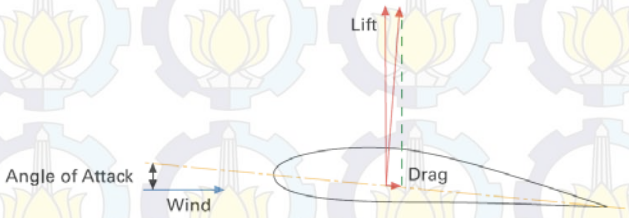
Gambar 2.1. Komponen utama turbin angin horizontal (Munteanu, 2008)

Berdasarkan gambar 2.1 komponen atau perangkat sistem konversi energi angin disusun dalam empat sub sistem:

1. Aerodinamika, terdiri atas bagian utama rotor turbin, berupa bilah turbin atau *blade* dan sebuah hub.
2. *Drive train*, terdiri dari poros putar kecepatan rendah (*low speed shaft*), kecepatan *multiplier* , dan poros putar kecepatan tinggi (*high speed shaft*) yang menggerakkan generator
3. Elektromagnetik, terdiri dari bagian utama generator listrik
4. Elektrik, terdiri dari komponen jaringan listrik.

Secara umum, terdapat dua dasar konfigurasi turbin angin, turbin angin sumbu vertikal (*Vertical Axis Wind Turbine*) dan turbin angin sumbu horisontal (*Horizontal Axis Wind Turbine*). Konfigurasi turbin angin yang paling banyak dikembangkan adalah turbin angin sumbu horizontal (HAWT) dengan *blade* yang jumlahnya 2 atau 3. Turbin angin sumbu horisontal (HWAT) adalah jenis turbin angin yang berbentuk menara dan mempunyai baling-baling yang fungsinya sebagai rotor untuk menangkap angin.

Prinsip kerja turbin angin yang ditinjau dari keluaran daya mekaniknya yaitu terletak pada komponen aerodinamiknya yang berupa *blade* (bilah turbin).



Gambar 2.2 Vektor gaya pada bilah turbin angin (Burton,2001)

Gambar 2.2 dapat dijelaskan bahwa ketika angin bertiup kearah bilah turbin angin yang berupa airfoil dari depan dengan kecepatan tertentu, maka akan menghasilkan vektor gaya lift (F_{Lift}) dan gaya drag (F_{drag}). Resultan gaya drag dan gaya lift menghasilkan gaya total (F_{total}) sehingga menyebabkan turbin berputar. Gaya drag dan gaya lift dipengaruhi oleh bentuk geometri bilah, kecepatan dan arah angin. Perubahan kedua gaya akan mempengaruhi kecepatan sudut dan torsi poros putar (*shaft*). Pengendalian sudut pitch (β) adalah salah satu sistem kontrol pada turbin angin dengan mengendalikan aerodinamis dari blade. Pengendalian aerodinamis dilakukan dengan mengontrol sudut *pitch blade* terhadap arah tiupan angin (*angle of attack* (α) (Zhang.J, 2008). Perubahan sudut bilah mempengaruhi kecepatan sudut dari *shaft* karena jumlah daya tiup agin yang diterima oleh bilah berubah, sehingga mempengaruhi proses konversi angin.

Energi angin diproduksi oleh turbin angin dengan memanfaatkan energi kinetik angin yang mengandung massa dan kecepatan sebagaimana ditunjukkan oleh persamaan (2.1) dan (2.2) dibawah ini (Musyafa'.A, 2012).

$$U = \frac{1}{2}mv^2 = \frac{1}{2}(\rho A_r x)v^2 \qquad (2.1)$$

$$P = \frac{dU}{dt} = \frac{1}{2}\rho A_r v^2 \frac{dx}{dt} = \frac{1}{2}\rho A_r v^3 \qquad (2.2)$$

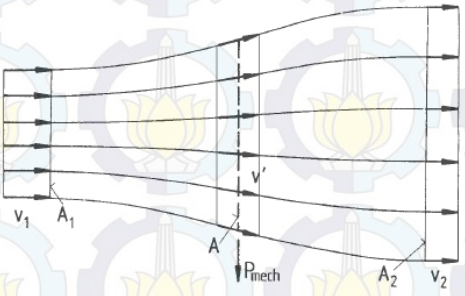
Dimana energi kinetik (U, joule), massa jenis udara (ρ, Kg/m^2), kecepatan angin (v, m/s) dan daya turbin angin (P).

Proses konversi energi oleh sebuah angin dapat dijelaskan dengan teori *Momentum Elementer Belt'z*. Energi mekanik hanya dapat di ekstrak pada besarnya nilai energi kinetik angin. Artinya dengan tanpa mengubah massa aliran, kecepatan aliran angin pada saat sebelum dan sesudah melintasi sebuah konverter energi (rotor turbin) berbeda. Hal ini terjadi dikarenakan sejumlah energi diserap oleh turbin angin. energi mekanik yang dihasilkan akibat oleh perbedaan kecepatan ini ditunjukkan oleh persamaan berikut (Erich Hau,2006).

$$P = \frac{1}{2}\rho A_1 {v_1}^3 - \frac{1}{2}\rho A_2 {v_2}^3 = \frac{1}{2}\rho(A_1 {v_1}^3 - A_2 {v_2}^3) \qquad (2.3)$$

Dengan massa aliran yang tetap, berlaku sebuah persamaan kontinuitas sebagai berikut,

$$\rho A_1 v_1 = \rho A_2 v_2) \qquad (2.4)$$



Gambar 2.3. Kondisi ekstraksi energi angin teori *momentum beltz* (Erich hau, 2006)

Perbandingan antara daya mekanik (P) yang dikonversi oleh turbin dan daya yang terkandung pada angin(P_0), disebut dengan koefisien daya turbin (Cp).

$$Cp = \frac{P}{P_0} = \frac{1}{2}\left|1 - \left(\frac{v_2}{v_1}\right)^2\right|\left|1 + \frac{v_2}{v_1}\right| \quad (2.5)$$

Dengan perbandingan kecepatan v_2/v_1=1/3, maka diperoleh daya koefisien Cp ideal maksimal yaitu sebesar 16/27 yaitu sebesar 0.593 atau sebesar 60% dari daya keseluruhan angin yang dapat dikonversi menjadi energi mekanik.

Performansi turbin angin dipengaruhi oleh daya yang dihasilkan oleh kecepatan angin yang bervariasi. Kecepatan angin yang bervariasi berpengaruh pada besaran *Tip Speed Ratio* turbin angin. *Tip Speed Ratio* (TSR) adalah variabel yang menunjukkan rasio antara kecepatan disekeliling *blade* dan kecepatan angin. TSR disimbolkan dengan λ (harika, 2006):

$$\lambda = \frac{R.\omega}{v} \quad (2.6)$$

Dimana R adalah panjang bilah turbin, ω adalah kecepatan rotor, dan *v* adalah kecepatan angin. TSR adalah variabel utama di dalam kontrol turbin angin. Koefisian daya turbin angin (Cp) adalah fungsi dari TSR (λ) dan sudut *pitch* (β), Cp (λ,β). Koefisien daya Cp (λ,β), secara numerik ditunjukkan oleh persamaan berikut.

$$Cp(\beta,\lambda) = C_1(C_2/\lambda_i - C_3\beta - C_4)e^{-C_5/\lambda_i} + C_6\lambda \quad (2.7)$$

$$\frac{1}{\lambda_i} = \frac{1}{\lambda+0.08\beta} - \frac{0.035}{\beta^3+1} \quad (2.8)$$

Cp maksimal akan diperoleh jika posisi sudut *pitch* berada pada posisi yang tepat dan *tip speed ratio* yang tepat. Cp yang maksimal akan menghasilkan daya turbin angin yang maksimal.

Dengan mensubtitusikan persamaan (2.2) dan (2.7) diperoleh persamaan daya turbin angin sebagai berikut,

$$P = \frac{1}{2} C_p(\lambda, \beta) \rho A_r v^3 \qquad (2.9)$$

Berdasarkan persamaan (2.6), daya maksimal turbin angin dapat diperoleh salah satunya dengan mengasumsikan bahwa *ω* berniai konstan dan dijadikan *setpoint* yang diinginkan, dan jari-jari bilah turbin (*R*) bernilai konstan, maka Cp hanya akan bergantung pada kecepatan angin (*v*) dan sudut *pitch* (*β*), sudut *pitch* akan dijadikan variabel yang dikontrol sebagai reaksi akibat kecepatan angin yang bervariasi, sehingga diperoleh nilai daya turbin yang diinginkan. Sudut *pitch* yang sesuai dengan berbagai kecepatan angin dan hubungannya dengan kecepatan rotor sudah dilakukan oleh penelitian sebelumnya dan akan digunakan sebagai dasar penelitian ini.

2.3 Imperialist Competitive Algorithm

Imperialist Competitive Algorithm (ICA) adalah sebuah evolusi algoritma baru untuk mengoptimasi sebuah sistem yang dasar strateginya bersumber pada kompetisi imperialis. Pada dasarnya optimasi dilakukan untuk membuat sebuah proses menjadi lebih baik dan dapat menyelesaikan permasalahan yang kompleks dalam sebuah sistem. Proses optimasi biasanya dilakukan dengan sebuah fungsi optimasi untuk menemukan nilai solusi yang optimal biasanya berupa nilai minimum. Secara *pseudo code*, algoritma Imperialist Competitive dapat dijelaskan dengan langkah-langkah berikut ini (Gargari.Atashpaz, 2007),

- Pilihlah beberapa titik (nilai) secara random pada fungsi dan inisialisasi *empire* (kerajaan)
- Gerakkan koloni menuju imperialisnya yang relevan (assimilasi)
- Jika terdapat koloni pada sebuah *empire* atau kerajaan yang memiliki harga lebih rendah dari imperialisnya. Tukarkan posisi koloni dan imperialis

- Menghitung *total cost* dari semua *empire* (berhubungan dengan kekuatan dari imperialis dan koloninya.
- Pilih koloni yang paling lemah dari sebuah *empire* yang paling lemah dan berikan mereka pada *empire* yang paling memungkinkan untuk memilikinya (kompetisi imperialis)
- Eliminasi *empire* (kerajaan) yang paling lemah
- Jika hanya terdapat satu *empire* (kerajaan), stop algoritma, jika tidak kembali ke langkah 2.

Langkah-langkah algoritma ICA diatas, dapat dijelaskan lebih rinci sebagaimana berikut (Gargari.Atashpaz, 2007),

1. Inisialisasi Kerajaan (*Initial Empire*)

Tahap 1 ini akan dibentuk terlebih dahulu sebuah kerajaan yang terdiri dari imperialis dan koloni. ICA mengawalinya dengan membentuk sebuah array dari nilai variabel yanag akan dioptimasi. Sebagaimana pada algoritma seperti GA, array ini disebut dengan "kromosom", maka di ICA dikenal dengan istilah negara atau "*country*". Sebuah negara adalah 1 x Nvar array. Array ini ditunjukkan persamaan sebagai berikut,

$$country = [p_1, p_2, p_3, \dots p_{N_{var}}] \qquad (2.10)$$

Variabel *(P1,P2,P3,...,PNvar)* merupakan variabel yang akan dioptimisasi sejumlah *Nvar*, *cost* dari tiap *country* dapat diketahui dengan mengevaluasi fungsi *cost* pada masing-masing variabel optimasi, berikut persamaannya,

$$cost = f(country) = f(p_1, p_2, p_3, \dots p_{N_{var}}) \qquad (2.11)$$

Algoritma optimasi diawali dengan menciptakan sebuah populasi yang berupa negara. Terdapat dua tipe negara, yaitu imperialis dan koloni. Untuk membentuk inisial *empire*, maka Pembagian koloni harus dilakukan berdasarkan kekuatan dari imperialis. Untuk membagi koloni berdasarkan imperialis

dengan tepat, maka *cost* imperialis harus dinormalisasi terlebih dahulu dengan persamaan berikut.

$$C_n = c_n - \min_i\{c_i\} \tag{2.12}$$

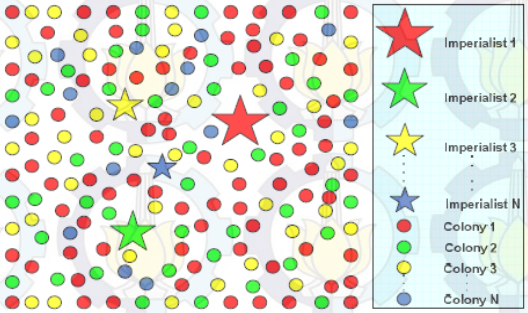
Dengan c_n merupakan *cost* dari imperialis ke-*n*, dan c_n adalah *cost* yang sudah dinormalisasi. Kemudian selanjutnya, kekuatan masing-masing imperialis didefinisikan sebagaimana berikut,

$$p_n = \left| \frac{c_n}{\sum_{i=1}^{N_{imp}} c_i} \right| \tag{2.13}$$

Kemudian diperoleh jumlah koloni awal untuk sebuah *empire* ke-*n* ,

$$N.C_n = round\{p_n N_{col}\} \tag{2.14}$$

Dengan *N.C.n* adalah jumlah awal koloni dari *empire* ke-*n* dan N_{col} merupakan jumlah koloni awal. Koloni dengan imperialis ke-*n* akan membentuk *empire* ke-*n*. Proses pembentukan empire pada awalnya ditunjukkan oleh Gambar 2.4.

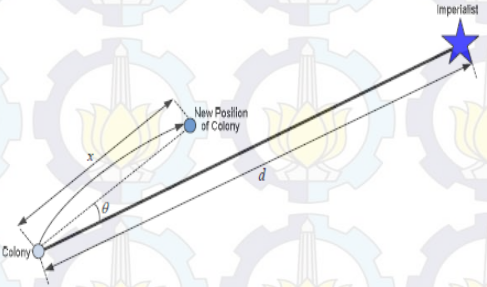


Gambar 2.4. Pembentukan inisialisasi *Empire* (Gargari.Atashpaz, 2007)

2. Pergerakan Koloni Sebuah Kerajaan Menuju Imperialis

Tahap 2 ini, Negara *imperialist* akan memulai dengan memperbaiki atau meningkatkan koloninya dengan cara menggerakkan semua koloni untuk mendekati imperialis. Sebagaiamana ditunjukkan oleh gambar 2.5, koloni mendekati imperialist sejauh x unit, dimana nilai x adalah variabel random yang terdistribusi seragam. Sehingga dari nilai x tersebut, kita mendapatkan persamaan berikut,

$$x \sim U(0, \beta x d) \tag{2.15}$$



Gambar 2.5. Pergerakan koloni menuju imperialis dalam kondisi acak (Gargari.Atashpaz, 2007)

Nilai β adalah sebuah angka yang lebih dari 1 dan d adalah jarak koloni dan imperialis. Nilai $\beta > 1$ akan mengakibatkan koloni bergerak lebih dekat dengan imperialisnya ditinjau dari kedua sisinya. θ adalah parameter yang terdistribusi seragam.

$$\theta \sim U(-\gamma, \gamma) \tag{2.16}$$

Dimana γ adalah parameter yang mengatur penyimpangan dari arah awal. Namun, nilai β dan γ tidak dipilih sembarangan, dalam sebagian besar impelementasi, nilai β sekitar 2 dan nilai γ sekitar $\pi/4$ (rad) untuk menghasilkan konvergensi yang baik untuk menuju global minimum.

3. Pertukaran Posisi Imperialis dan Koloni

Tahap 3 ini, sementara bergerak mendekat kearah imperialis, koloni akan menjangkau daerah atau posisi yang harganya lebih rendah dari imperialis. Pada tahap ini Imperialis bergerak ke arah posisi koloni dan begitu sebaliknya. Algoritma akan terus berlanjut oleh imperialis yang berpindah ke posisi yang baru dan koloni mulai mendekat ke posisi tersebut. Imperialis dan koloni berpindah posisi sebagaimana ditunjukkan gambar 2.6 dibawah ini.

Gambar 2.6. Pertukaran posisi koloni dan *Imperialist* (Gargari.Atashpaz, 2007)

4. Total Kekuatan Dari Kerajaan

Tahap 4 ini, ditentukan total harga dari keseluruhan kekuatan kerajaan atau *empire* yang utamanya dipengaruhi oleh kekuatan negara imperialis tetapi kekuatan koloni juga memiliki pengaruh walaupun diabaikan karena nilainya kecil. Total Cost ditunjukkan oleh persamaan berikut,

$$T.C_n = Cost(Imperialist_n) + \xi mean\{cost(colonies\ of\ empire_n)\} \quad (2.17)$$

Nilai ξ menunjukkan pengaruh kontribusi dari koloni. Dengan *T.C.n* adalah total *cost* dari *empire* ke-n dan ξ adalah nilai positif kurang dari satu, sehingga menyebabkan kekuatan total *empire* lebih dipengaruhi oleh imperialis daripada koloni.

5. Kompetisi Imperialis

Tahap 5 ini, merupakan proses dasar filosofi terjadinya kompetisi imperialis, semua kerajaan (*empire*) mencoba untuk memiliki koloni dari kerajaan yang lain dan mengontrol mereka. Kompetisi imperialis ini secara perlahan akan menurunkan kekuatan dari kerajaan yang lemah dan akan meningkatkan kekuatan dari kerajaan yang lebih kuat. Sebagaimana diilustrasikan dengan mengambil beberapa koloni (satu koloni) pada kerajaan yang paling lemah, kemudian dibuat sebuah kompetisi kerajaan mana yang akan memilikinya. Berdasarkan total kekuatan mereka (*empire*), pada kompetisi ini, masing-masing kerajaan memiliki kemungkinan (*likelihood*) untuk mengambil dan memiliki koloni yang lemah tersebut. Sehingga koloni yang terlemah itu tidak selalu akan dimiliki oleh kerajaan yang paling kuat, tetapi kerajaan yang paling memungkinkan yang akan memilikinya.

6. Eliminasi Kerajaan Yang Terlemah

Tahap 6 ini kerajaan yang tidak berdaya akan runtuh dan pada kompetisi imperialis, koloni mereka akan dibagikan kepada kerajaan yang lain. Kriteria penentuan kerajaan terlemah dapat didefinisikan berbeda-beda. Namun dalam kebanyakan implementasi algoritma ICA mengasumsikan dan menganalogikan, kerajaan dikatakan tidak berdaya adalah kerajaan yang kehilangan semua koloninya. Kerajaan ini akan dieliminasi atau dihilangkan.

7. Konvergensi

Tahap 7 ini merupakan tahap dimana semua kerajaan yang lemah akan jatuh dan tereliminasi, sehingga hanya ada satu yang paling kuat yang akan mengontrol semua koloni. Akhirnya dalam kondisi yang baru, semua koloni berada pada posisi yang sama dan mempunyai harga yang sama dan dikontrol secara keseluruhan oleh imperialis dengan posisi yang sama dan harga dirinya sendiri, dengan kondisi demikian

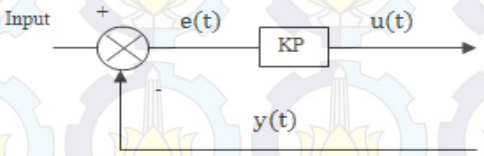
maka itulah tahap terakhir kompetisi imperialis dan berhentinya algoritma (Gargari.Atashpaz, 2007).

2.4 PID Kontrol

PID (Proportional-Integral-Derivative) kontrol adalah kontroller untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya *feedback* (umpan balik) pada sistem tersebut. PID kontrol tediri dari 3 jenis komponen kontrol yaitu Proportional, integral dan derivative, masing-masing jenis kontrol memiliki kharakteristik yang berbeda-beda (Ogata.K, 1995).

Kontrol Proportional (Kp) adalah kontroller yang memberikan efek mengurangi *rise time* (waktu naik), tidak menghapus *error steady state*, dan menghasilkan *overshoot* pada sistem, hal ini dikarenakan *output* proportional adalah hasil kali antara konstanta proportional dengan nilai errornya, sehingga ketika terjadi perubahan pada sinyal *input* akan mengakibatkan perubahan *output* sebesar konstanta pengalinya secara langsung, berikut diagram blok dan fungsi matematis kontrol proportional.

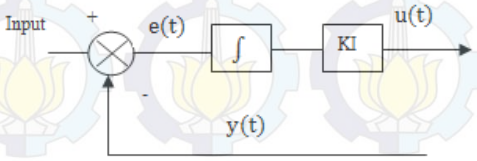
$$U(t)= Kp\ e(t) \tag{2.18}$$



Gambar 2.7. Diagram blok kontrol Proportional

Kontrol Integral (Ki) adalah kontroller yang akan memberikan efek mengahapus *error steady state* yang biasanya dihasilkan oleh kontrol proportional. Sehingga kontroller ini akan membantu menaikkan respon sehingga menghasilkan *output* yang diinginkan. Berikut adalah diagram blok dan fungsi matematis kontrol integral,

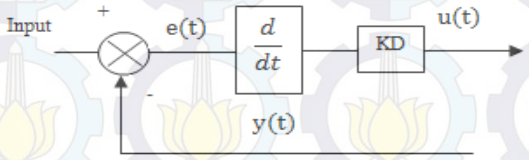
$$u(t) = Ki \int_0^t e(t)dt \quad (2.18)$$



Gambar 2.8. Diagram blok kontrol Integral

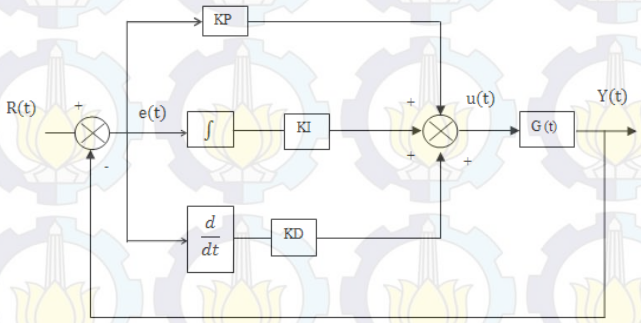
Kontrol derivative (Kd) adalah kontrol yang akan memberikan efek meningkatnya stabilitas sistem, mengurangi *overshoot*, dan menaikkan respon transfer. Kontrol Derivatif tidak dapat digunakan dalam kondisi sendiri tanpa kontroller lain, hal ini dikarenakan kontrol derivative akan berubah ketika terjadi perubahan error, sehinga ketika tidak ada perubahan error maka kontroller ini tidak beraksi. Berikut diagram blok dan fungsi matematika kontrol derivatif,

$$u(t) = Kd \frac{de(t)}{dt} \quad (2.19)$$



Gambar 2.9. Diagram blok kontrol Derivative

Kontrol PID (Proportional-Integral-Derivative) adalah gabungan dari ketiga kontrol diatas Proportional, Integral, dan derivative. Gabungan ketiganya akan mempengaruhi satu sama lain sehingga mampu meredam kekurangan masing-masing dan diperoleh nilai keluaran kontrol yang diingankan. Diagram blok dari kontrol PID adalah ditunjukkan oleh gambar 2.11 dibawah,



Gambar 2.10 Diagram blok kontrol PID

Berdasarkan gambar 2.10 dapat diperoleh sinyal keluaran kontrol PID (u) adalah sebagai berikut.

$$u = Kp.e + Ki\int_0^t e\,dt + Kd\frac{de}{dt} \tag{2.20}$$

PID kontrol bekerja pada loop tertutup dengan variabel e (error), nilai masukan (R(t)), dan Keluaran (Y(t)). Prinsip kerjanya adalah sinyal error akan dikirim ke PID dan kontroller akan menghitung keseluruhan turunan dan integral dari sinyal error. Keluaran kontroller berupa sinyal (u) setelah mengalami penguatan oleh kontrol proportional (Kp), dikalikan dengan ukuran kesalahannya, ditambah dengan penguatan integral (Ki), dikalikan ukuran kesalahan integralnya ditambah penguatan turunan (Kd) dikalikan ukuran kesalahan derivasinya. Selanjutnya sinyal (u) akan dikirim ke Plant, sehingga akan mendapatkan keluaran baru (y) dan keluaran ini akan dikirim kembali melalui sensor untuk mendapatkan error baru. Proses ini akan berlangsung secara terus menerus seperti semula.

Berikut efek dari setiap kontrol (Kp, Ki, Kd) dalam sistem tertutup jika nilainya dinaikkan.

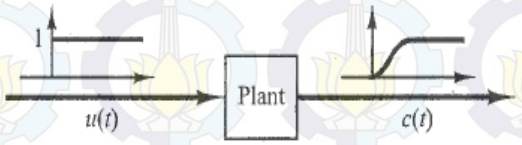
Tabel 2.1. Efek dari setiap kontol (Kp Ki,Kd)

Respon *Close loop*	***Rise Time***	***Maximum Overshoot***	***Settling Time***	***Error Steady State***
Kp	Menurun	Meningkat	Perubahan kecil	Menurun
Ki	Menurun	Meningkat	Meningkat	Hilang
Kd	Perubahan kecil	Menurun	Menurun	Perubahan kecil

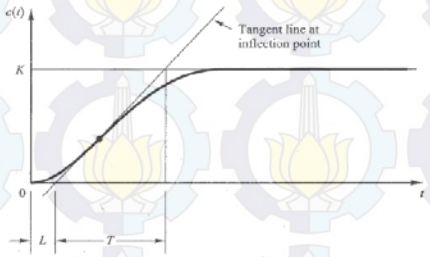
2.5 Tuning parameter PID kontrol

Salah satu metode yang digunakan untuk menala (tuning) parameter PID kontrol adalah dengan metode Zighler-Nichols. Zighler Nichols menentukan nilai gain (penguat) Kp, Ti, Td berdasarkan respon transien kharakteristik yang diberikan plant. Metode ini bertujuan untuk menentukan nilai *Maximum Overshoot*nya sebesar 25% dengan inputannya berupa sinyal step

Dalam penerapannya pada plant orde pertama, maka digunakan metode Zighler-Nichols yang pertama yaitu menggunakan Kurva-S. Kurva-S dicirikan dengan dua konstanta, yaitu waktu tunda atau *delay time* (L) dan waktu konstan atau *time constant* (T). Waktu tunda dan waktu konstan ditentukan dengan menggambar garis *tangent* pada titik belok dari bentuk kurva–S dan menentukan perpotongan antara garis *tangent* dengan sumbu vertikal waktu dan garis c(t)=K, sebagaimana ditunjukkan oleh gambar 2.12. Kurva-S diperoleh dari respon *open loop plant* dengan inputan sinyal step.



Gambar 2.11. Respon *Unit-step* dari *Plant* (Ogata.K, 1995)



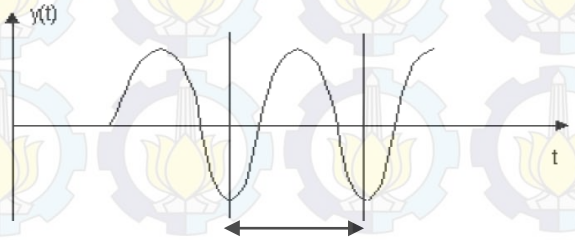
Gambar 2.12 Kurva-S respon (Ogata.K, 1995)

Kemudian dimasukkan nilai T dan L yang diperoleh kedalam tabel metode pertama (Kurva-S) Zighler-Nichols sebagai berikut untuk mendapatkan parameter Kp, Ti, Td,

Tabel 2.2. Tuning Zighler-Nichols dengan Metode Pertama

Tipe Kontroller	Kp	Ti	Td
P	T/L	∞	0
PI	0.9 T/L	L/0.3	0
PID	1.2 T/L	2L	0.5L

Metode Zighler Nichols yang kedua adalah metode osilasi, Metode osilasi dilakukan pada sistem *close loop*. Parameter Integrator (Kontrol Integral) di berikan nilai tak terhingga dan parameter differensial (Kontrol *Derivative*) diberikan nilai nol. Selanjutnya, parameter proportional dinaikkan secara bertahap mulai dari nol hingga mencapai nilai yang mengakibatkan respon sistem berosilasi dengan besar yang tetap. Nilai penguat proportional saat sistem mencapai kondisi osilasi dengan besar yang tetap disebut dengan *ultimate gain* Kcr. Sedangkan periode dari osilasi disebut *ultimate periode* Pcr. Berikut gambar respon sistem yang berosilasi dengan besar yang sama,



Gambar 2.13. Respon sistem metode Osilasi

Zighler Nichols menyarankan untuk menggunakan rumus berikut ini untuk tuning PID dengan metode osilasi.

Tabel 2.3. Tuning Zighler-Nichols dengan Metode Osilasi

Tipe Kontroller	Kp	Ti	Td
P	0.5 Kcr	-	-
PI	0.45 Kcr	0.5 Pcr	-
PID	0.6 Kcr	0.5 Pcr	0.125 Pcr

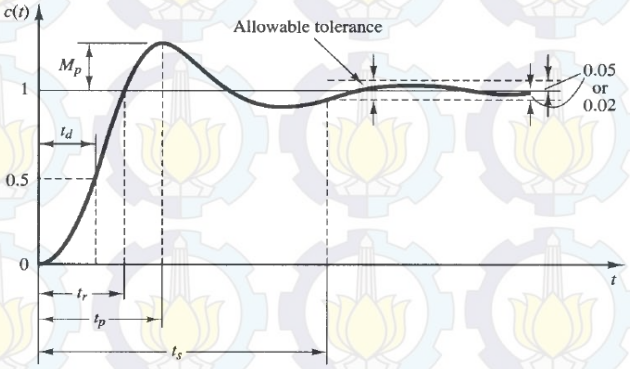
Imperialist Competitive Algorithm (ICA) dalam optimasi sistem kontrol ini dilakukan dengan fungsi evaluasi atau fungsi objektifnya menggunakan fungsi ITAE (*Integral Time Absolute Error*), hal ini dikarenakan ITAE merupakan suatu kriteria indeks performansi yang didasarkan pada integral waktu dan harga mutlak dari error, berikut persamaan ITAE,

$$ITAE = \int_0^\infty t|e(t)|\,dt \qquad (2.21)$$

Dimana t adalah waktu dan e adalah error,sehingga diperoleh nantinya parameter nilai Kp, Td, Ti, terbaik.

2.6 Respon PID Kontrol

Terdapat spesifikasi respon untuk menganalisa performansi dari keluaran sebuah sistem kontrol yang diberikan suatu sinyal masukan tertentu. Dalam menganalisa performansi respon PID kontrol digunakan beberapa ukuran kualitas respon, diantaranya:



Gambar 2.14. Respon sinyal uji step (Ogata.K, 1995)

Berdasarkan gambar 2.14 diatas. Beberapa ukuran kualitas respon dapat dijelaskan,

1. *Maximum overshoot*, Mp adalah nilai puncak maksimal dari kurva respon yang diukur dari nilai *setpoint*. Pada umumnya persentase *maximum overshoot* didefinisikan sebagai berikut,

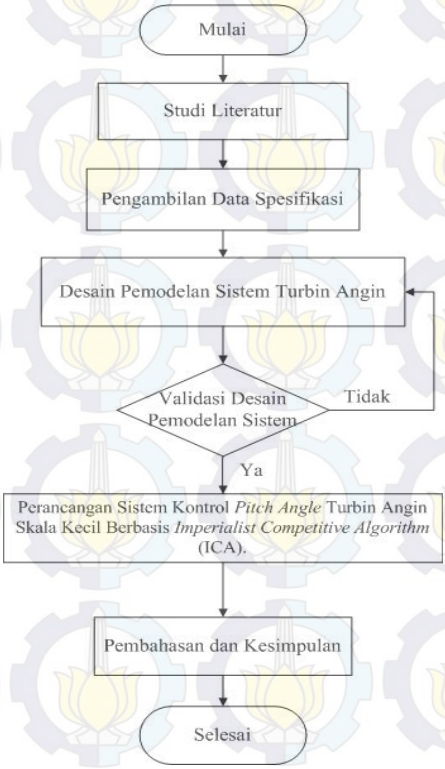
$$Maximum\ overshoot\ (\%) = \frac{c(t_p) - c(\infty)}{c(\infty)} x\ 100\% \quad (2.22)$$

2. *Settling time*, t_s adalah waktu yang dibutuhkan kurva respon untuk mencapai nilai akhir yang besarannya di spesifikasikan sebagai persentase *absolute* dari nilai akhir (biasanya 2% atau 5%).

"Halaman ini memang dikosongkan"

BAB III
METODOLOGI PENELITIAN

Bab ini akan menjelaskan tentang alur pengerjaan tugas akhir ini, yaitu study literatur, pengambilan data spesifikasi, desain pemodelan sistem turbin angin, validasi model sistem. desain sistem kontrol *pitch angle* turbin angin *berbasis Imperialist Competitive Algorithm* (ICA), Pembahasan dan kesimpulan. Berikut Gambar 3.1 yang merupakan diagram alir (*flowchart*) dalam pengerjaan tugas akhir.



Gambar 3.1. Diagram alir penelitian tugas akhir

Berikut penjelasan secara rinci mengenai diagram alir penelitian tugas akhir pada gambar 3.1.

3.1 Pengambilan Data Spesifikasi

Data yang diperlukan dalam penelitian ini adalah data mengenai spesifikasi prototipe turbin angin skala kecil yang telah dibangun oleh peneliti sebelumnya. Prototipe turbin angin tersebut merupakan objek penelitian dari penelitian tugas akhir ini. Data meliputi spesifikasi bilah turbin, sudut pitch (β), *tip speed ratio* (λ), kecepatan putar rotor *shaft* turbin (ω), dan kecepatan angin (v).

Prototipe turbin angin berupa turbin angin sumbu horizontal dengan tipe bilah *non uniform* Airfoil NREL S83n dengan tiga buah bilah (*blade*), panjang *blade* 1 m, dengan massa masing-masing bilah 1297 gram. Turbin angin tersebut telah dirancang oleh peneliti sebelumnya (Farid RM, 2011). Penelitian pada saat itu adalah melakukan percobaan pencarian sudut *pitch* optimal untuk prototipe turbin angin, sehingga diperoleh hubungan antara kecepatan angin dengan sudut *pitch* tertentu yang menghasilkan putaran rotor yang paling optimal.

Tabel 3.1. Sudut *pitch* optimal untuk kecepatan angin yang berbeda (Farid R.M, 2011)

Kecepatan angin (m/s)	Sudut pitch Optimal	RPM Maksimum
2.80	10.35	39.58
3.80	10.37	54.28
4.10	13.10	64.38
4.80	10.15	68.96
6.50	13.16	112.83
7.00	16.19	99.02
7.50	10.87	168.09

Hasil penelitian diatas menunjukkan bahwa pada kecepatan angin tertentu agar menghasilkan kecepatan rotor yang maksimal dilakukan dengan sudut *pitch* tertentu yang optimal. Hasil

percobaan pada penelitian ini yang kita gunakan untuk memperoleh fungsi alih *plant* turbin angin yang selanjutnya untuk merancang sistem kontrol pada penelitian ini.

3.2 Desain Pemodelan Sistem Turbin Angin

3.2.1 Pemodelan *plant* turbin angin

Desain pemodelan sistem turbin angin dalam penelitian tugas akhir ini hanya ditinjau dari bagian mekanik turbin angin saja. Desain pemodelan sistem turbin angin diawali dengan memodelkan persamaan matematika mekanika turbin yang diperoleh dari beberapa referensi jurnal (Sneckenberger, 2003 dan Abbas furrat, 2010).

Pemodelan matematika atau fungsi transfer turbin angin diperoleh dari hubungan torsi aerodinamika turbin (T_A), kecepatan putar rotor (ω), dan momen inersia (J_T**)** bilah turbin angin dalam menghasilkan daya mekanik turbin. Daya mekanik turbin (P_T), diperoleh dari hasil perkalian torsi aerodinamika turbin (T_A) dengan kecepatan putar rotor (ω), sebagaimana ditunjukkan oleh persamaan berikut.

$$P_T = \omega . T_A \tag{3.1}$$

Turbin angin yang dinamik dapat dimodelkan melalui persamaan matematika berikut,

$$J_T \omega_T = T_A - T_E \tag{3.2}$$

Dimana, (J_T) merupakan momen inersia rotor turbin, (ω_T) merupakan kecepatan sudut rotor, T_A adalah torsi aerodinamika turbin angin, dan T_E adalah torsi mekanik turbin angin yang terhubung kearah generator dan dalam hal ini Q_E diasumsikan bernilai konstan, karena dalam penelitian ini yang ditinjau adalah aerodinamika turbin. Persamaan torsi aerodinamika turbin ditunjukkan oleh persamaan (3.3).

$$T_A = \frac{1}{2} \rho A R C_T(\beta, \lambda) v^2 \tag{3.3}$$

$$C_p(\beta,\lambda) = \lambda . C_T(\beta,\lambda) \quad (3.4)$$

Pada tugas akhir ini digunakan pendekatan nilai Cp, dengan persamaan numerik berikut ini,

$$C_P(\lambda,\beta) = (0.44 - 0.0167\beta)sin\frac{\pi(\lambda-3)}{15-0.3\beta} - 0.00184(\lambda-3) \quad (3.5)$$

Berdasarkan persamaan (3.3) dan (3.4), nilai dari torsi aerodinamika turbin dipengaruhi oleh besaran koefisien torsi turbin (C_T). Sedangkan besaran nilai C_T dipengaruhi oleh nilai C_p yang berdasarkan pada nilai sudut pitch (β) dan tip speed ratio (λ) turbin angin.

Sistem kontrol yang akan dirancang adalah sistem kontrol turbin yang menggunakan PID kontroller. PID Kontroller adalah kontrol linear, sehingga dalam perancangannya turbin angin yang dinamis akan dilakukan proses linearisasi pada titik operasi tertentu. Dengan mengasumsikan, $T_A|op = T_E|op$, pada titik operasi tertentu, sehingga linearisasi persamaan (3.2) dan (3.3) menghasilkan persamaan turbin angin sebagai berikut,

$$J_t\Delta\dot{\omega} = \frac{\partial\dot{\omega}}{\partial u}\Delta v + \frac{\partial\dot{\omega}}{\partial\omega}\Delta\omega + \frac{\partial\dot{\omega}}{\partial\beta}\Delta\beta \quad (3.6)$$

$$J_t\Delta\dot{\omega} = \alpha\Delta v + \gamma\Delta\omega + \delta\Delta\beta \quad (3.7)$$

Berdasarkan persamaan (3.7), nilai α, γ, δ merupakan koefisien linearisasi turbin angin. besaran nilai α, γ, δ turunan parsial dari torsi aerodinamika turbin angin (T_A) terhadap kecepatan putar rotor turbin (ω), sudut *pitch* (β) dan kecepatan angin (v) pada titik operasi tertentu. Nilai α,γ, dan δ dapat dicari dengan menggunakan persamaan berikut.

$$\gamma = \frac{\partial\tau_A}{\partial\omega}\Big|_{op} = \frac{\partial}{\partial\omega}(J_t\omega)\Big|_{op} = \frac{1}{2}\rho A{v_{op}}^3\frac{\partial}{\partial\omega}\left[\frac{C_P(\lambda,\beta)}{\omega}\right]\Big|_{op} = \frac{\partial\tau_m}{\partial\omega}\Big|_{op} = K11 + K12 + 13 \quad (3.8)$$

$$\alpha = \frac{\partial \tau_A}{\partial v}\Big|_{op} = \frac{\partial}{\partial v}(J_t \omega)\Big|_{op} = \frac{1}{2}\rho A \frac{1}{\omega_{op}} \frac{\partial}{\partial v}[C_P(\lambda, \beta) * v^3]\Big|_{op} = \frac{\partial \tau_m}{\partial v}\Big|_{op} = K21 + K22 + K23 \quad (3.9)$$

$$\delta = \frac{\partial \tau_A}{\partial \beta}\Big|_{op} = \frac{\partial}{\partial \beta}(J_t \omega)\Big|_{op} = \frac{1}{2}\rho A \frac{{v_{op}}^3}{\omega_{op}} \frac{\partial}{\partial \beta}[C_P(\lambda, \beta)]\Big|_{op} = \frac{\partial \tau_m}{\partial \beta}\Big|_{op} = K31 + K32 + K33 \quad (3.10)$$

Nilai K11, K12, K13, K21, K22, K23, K31, K32, dan K33 diperoleh dengan persamaan pada hal lampiran (Lampiran A). Persamaan (3.7) akan di *laplace* kan sehingga menjadi persamaan berikut.

$$J_t s \Delta\omega(s) = \alpha \Delta v(s) + \gamma \Delta\omega(s) + \delta \Delta\beta(s) \quad (3.11)$$

$$J_t \left(s - \frac{\gamma}{Jt}\right) \Delta\omega(s) = \alpha \Delta v(s) + \delta \Delta\beta(s) \quad (3.12)$$

$$\Delta\omega(s) = \left[\frac{\alpha}{Jt} \Delta v(s) + \frac{\delta}{jt} \Delta\beta(s)\right] \frac{1}{s - \frac{\gamma}{Jt}} \quad (3.13)$$

$\Delta\omega(s)$ adalah keluaran dari sistem kontrol, yang berupa kecepatan putar rotor (shaft) sedangkan $\alpha\Delta v(s)$ dan $\delta\Delta\beta(s)$ merupakan masukan linear turbin yang berupa kecepatan angin dan sudut *pitch*. Desain kontrol yang dirancang merupakan kontrol dengan mengekspresikan gangguan sudut *pitch* ($\Delta\beta$). Desain kontrol tersebut ditunjukkan oleh persamaan berikut ini dengan persamaan PID kontrol standar,

$$\Delta\beta(t) = K_p \Delta\omega(t) + K_I \int \Delta\omega(t)\ dt + K_D \Delta\dot{\omega}(t) \quad (3.14)$$

Tujuan dari desain kontrol akan menentukan nilai gain Kp, Ki, Kd yang tepat untuk mempertahankan kestabilan sistem close-loop dan mencapai respon yang baik. Pertama dilakukan pemodelan sistem *close loop* ke dalam *laplace* atau domain s. Sehingga persamaan 3.15 menjadi,

$$\Delta\beta(s) = K_p\Delta\omega(s) + K_I\frac{1}{s}\Delta\omega(s) + K_D s\Delta\omega(s) \quad (3.15)$$

Dengan mengintegrasikan persamaan (3.17) dengan persamaan (3.14), maka diperoleh persamaan berikut,

$$(s - A)\Delta\omega(s) = B\left(\mathrm{K_p}\Delta\omega(\mathrm{s}) + \mathrm{K_I}\frac{1}{\mathrm{s}}\Delta\omega(\mathrm{s}) + \mathrm{K_D}\,\mathrm{s}\Delta\omega(\mathrm{s})\right) + Bd\Delta v(s) \quad (3.16)$$

Pada operasinya, agar turbin angin mendapatkan daya yang maksimal, maka diharapkan turbin bekerja pada kondisi optimalnya, berdasarkan tabel 3.1 diatas, perancangan kontrol ini menggunakan kecepatan angin 4.8 m/s, karena pada kecepatan ini merupakan kecepatan yang berada di rentang tengah antara kecepatan angin 2.8 m/s – 7.5 m/s, sehingga untuk merespon kecepatan angin bawah dan tinggi tidak terlalu berat. Turbin bekerja dengan sudut *pitch* optimalnya 10.15^{o} dan menghasilkan kecepatan rotor sebesar 68.96 RPM. Selain itu berdasarkan spesifikasi turbin yang ada, turbin angin memiliki moment inersia 1.297 kgm^2 dengan panjang bilah 1 m dan massa jenis udara standar adalah 1.25 kg/m^3.

Nilai A, B, dan Bd, diperoleh dengan memasukkan nilai-nilai atau parameter desain tersebut kedalam persamaan (3.16) diatas dan diperoleh nilai A, B, dan Bd dengan persamaan berikut,

$$A = \frac{\gamma}{J_t} =, B = \frac{\delta}{J_t}, Bd = \frac{\alpha}{J_t} \quad (3.18)$$

Dengan mengintegrasikan persamaan (3.16) dan hasil dari persamaan (3.18), maka diperoleh fungsi alih turbin angin sebagai berikut.

$$Ft(\mathrm{s}) = \frac{\Delta\omega(s)}{\Delta v(s)} = \frac{Bds}{(1-BK_D)s^2+(-A-BK_p)s+(-BK_I)} \quad (3.19)$$

$$Ft(s) = \frac{\Delta\omega(s)}{\Delta v(s)} = \frac{Bds}{(1-0.0835K_D)s^2+(-(-0.1816)-0.0835K_p)s+(-0.0835K_I)} \quad (3.20)$$

agar sistem stabil maka akar-akarnya harus bernilai negatif maka nilai dari $(1 - 0.1801\mathrm{K_D}) > 0$, $(-(-0.1816) - 0.0835\mathrm{Kp} > 0$, dan $-0.0835\mathrm{KI} > 0$, maka nilai KD≤11.98, $\mathrm{K_p} \leq 0.46$, $\mathrm{K_I} \geq 0$. Dengan A=-0.1816, B=0.0835, Bd=1.2103 maka diperoleh fungsi alih turbin angin sebagai berikut.

$$Ft(s) = \frac{\Delta\omega(s)}{\Delta v(s)} = \frac{1.2103s}{s^2+0.1816s} \quad (3.21)$$

3.2.2 Pemodelan *Opto Interrupter*

Pada *plant* turbin angin skala kecil yang dirancang dalam perancangan sistem kontrol ini terdapat *opto interrupter* tipe ITR8105 (Sunarto,2012). *Opto interrupter* ini bagian *plant* turbin angin yang befungsi untuk merekam kecepatan rotor turbin yang berasal dari *rotary encoder*. *Rotary encoder* berupa piringan yang mempunyai 20 lubang yang berada pada poros turbin angin. *Opto interrupter* ini akan menghasilkan input sebesar 1 jika pada *rotary* tidak terhalang oleh piringan dan akan bernilai 0 jika terhalang oleh piringan *rotary encoder* yang terdiri dari 20 lubang.

Keluaran dari *opto interupter* berupa tegangan sebesar 0-5 volt untuk masukan ke sistem digital. Desain *opto interrupter* yang akan digunakan mempunyai waktu konstan (τ) sebesar 10 ms. *Opto interuppter* dapat dimodelkan berupa orde satu, nilai *gain* (K) merupakan perbandingan antara span *output* dengan span inputnya. Berikut model *opto interrupter* dalam orde 1.

$$\frac{Output}{Input} = \frac{K}{\tau s+1} \quad (3.22)$$

Dengan memasukkan nilai-nilai parameternya diatas diperoleh nilai gain (K) sebagai berikut,

$$K = \frac{Span\ \ 0utput}{Span\ \ Input} = \frac{5-0}{1-0} = 5 \tag{3.23}$$

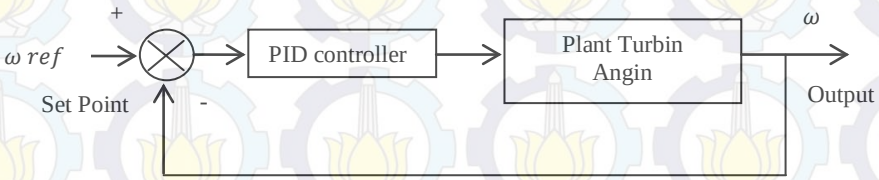
Sehingga diperoleh fungsi alih *opto interrupter* sebagai berikut,

$$F_{opto} = \frac{Output}{Input} = \frac{5}{0.01s+1} \tag{3.24}$$

Dengan mengintegrasikan persamaan (3.21) dengan persamaan (3.24), maka fungsi alih *plant* turbin angin secara keseluruhan untuk desain sistem kontrol yang dirancang menjadi,

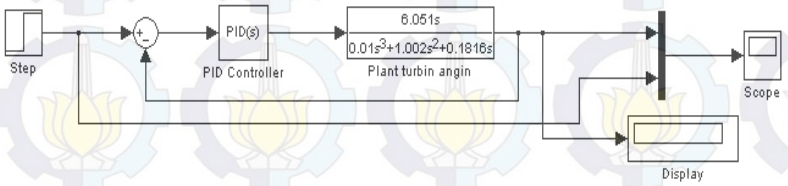
$$F_T(s) = \frac{\Delta\omega(s)}{\Delta v(s)} = \frac{6.051s}{0.01s^3+1.002s^2+0.1816s} \tag{3.25}$$

Berdasarkan pemodelan fungsi alih turbin angin linear, maka perancangan sistem kontrol *pitch angle* turbin angin dapat ditunjukan dengan diagram blok berikut ini,



Gambar 3.2. Diagram blok sistem kontrol turbin angin

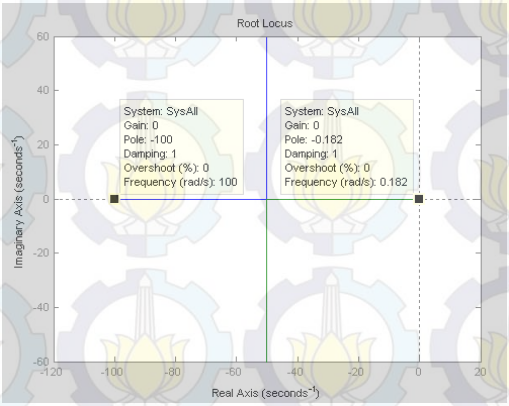
Berdasarkan gambar 3.2, sistem kontrol sudut *pitch* turbin angin direpresentasikan dengan setpoint berupa kecepatan putar rotor turbin pada sudut *pitch* optimalnya yang ingin dicapai, menggunkan kontrol PID yang masukannya berupa error dari kecepatan putar rotor. Berikut adalah desain simulasi sistem kontrol *pitch angle* turbin angin skala kecil pada Simulink Matlab 7.12.0 (R2011a).



Gambar 3.3. Desain simulasi sistem kontrol pada Simulink Matlab 7.12.0 (R2011a).

3.3 Validasi Desain Pemodelan Sistem

Validasi perancangan pemodelan sistem kontrol *pitch angle* turbin angin dilakukan dengan uji *root locus* pada fungsi alih *plant* linear turbin angin. berikut gambar 3.4. hasil uji *root locus*.

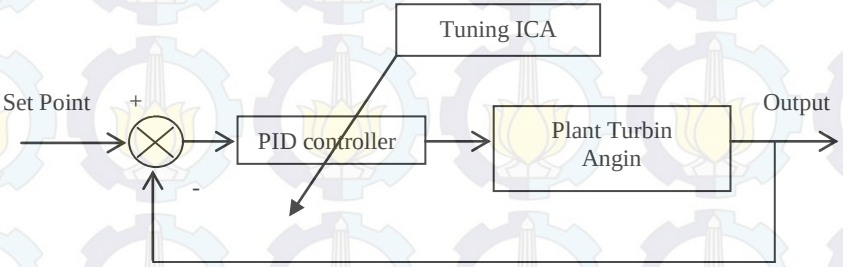


Gambar 3.4. Hasil Plot Root Locus

Berdasarkan plot *root locus* diatas, terlihat bahwa pole-pole sistem mempunyai akar-akar real negatif yang menunjukkan sistem bekerja di daerah stabil.

3.4 Perancangan Sistem Kontrol *Pitch Angle* Turbin Angin Skala Kecil Berbasis *Imperialist Competitive Algorithm* (ICA).

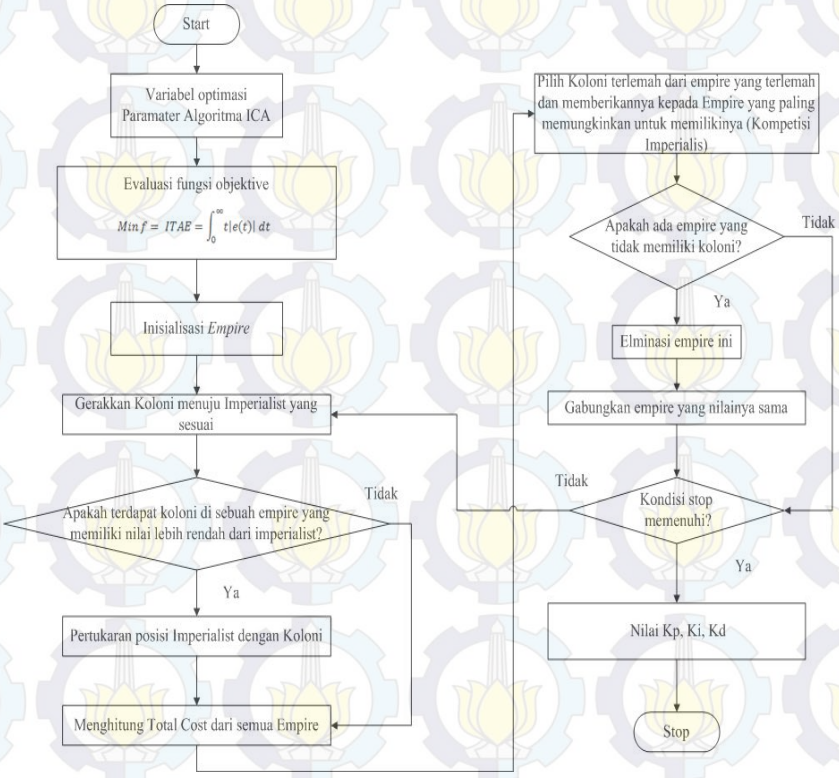
Perancangan sistem kontrol pitch angle berbasis algoritma kompetisi imperialis (ICA) dapat dijelaskan dengan diagram blok dan *flowchart* optimasi ICA berikut ini,



Gambar 3.5. Diagram blok sistem kontrol berbasis ICA

Berdasarkan gambar 3.5 diatas, sistem kontrol berbasis algoritma kompetisi imperialist (ICA) berupa sistem kontrol yang menggunakan kontrol PID. Nilai parameter kontrol Kp, Ki, dan Kd dioptimasi dengan algoritma ICA, sehingga memberikan solusi nilai parameter kontrol yang optimal untuk sistem kontrol yang dirancang. ICA akan menala nilai Kp, Ki, Kd dengan memberikan terlebih dahulu batasan nilai Kp, Ki, Kd yang ingin diperoleh. Prosesnya optimasi didasarkan pada fungsi objektif berupa nilai minimal dari ITAE (*Integral Time Absolute Error*). *Error* adalah selisih antara *output* dengan *input* dari sistem kontrol.

Berdasarkan gambar 3.6 dibawah ini, menunjukkan *flowchart* yang menjelaskan langkah-langkah algoritma optimasi ICA bekerja, Tahapan optimasi nilai Kp, Ki, dan Kd untuk sistem kontrol dengan algoritma ICA adalah sebagai berikut,



Gambar 3.6. *flowchart* optimasi kontrol PID berbasis ICA

- Menentukan Parameter Algoritma

Sebelum algoritma optimasi dijalankan, maka ditentukan terlebih dahulu parameter algoritma yang akan digunakan. Parameter algoritma ICA yang digunakan adalah antara lain adalah jumlah Negara (*Countries*), jumlah negara adalah besaran dimensi masalah yang akan dioptimasi. Jumlah Imperialis awal, merupakan banyaknya *empire* (kerajaan) awal yang ingin dibentuk. Jumlah koloni adalah selisih jumlah Negara dengan jumlah Imperialis awal. Dekade, merupakan banyaknya iterasi yang dilakukan untuk

melakukan proses optimasi. Kecepatan revolusi (*revolution rate*), merupakan kecepatan revolusi koloni untuk berubah posisi secara random. Asimilasi (β) dan Sudut Asimilasi (γ) adalah nilai yang menyebabkan koloni bergerak mendekati imperialis. Dan zetta (ξ) adalah nilai yang digunakan untuk menghitung *Total Cost* sebuah *empire*. Berikut parameter optimasi

Tabel 3.2. Parameter Algoritma ICA

Parameter	Nilai
Jumlah Negara	50.00
Jumlah imperialis awal	5.00
Jumlah koloni	45.00
Dekade	50.00
Kecepatan Revolusi	0.30
Asimilasi (β)	2.00
Sudut Asimilasi (γ)	0.50
Zetta (ξ)	0.02

- Membuat Inisial Empire

Inisialisasi empire (kerajaan) adalah tahapan untuk menentukan nilai variabel yang akan dioptimasi. Variabel yang akan dioptimasi berupa tiga variabel yaitu nilai Kp, Ki, dan Kd kontrol PID, variabel optimasi ditentukan batas atas dan bawahnya. Pada tugas akhir ini digunakan batas nilai Kp pada 0.5 sampai 1.5, nilai Ki pada 1 sampai 5, dan batas nilai Kd pada 0.01 sampai 0.1, batas ini diperoleh secara praktik dicobakan pada sistem untuk tiap-tiap rentang batasan nilai Kp, Ki, kd, yang menghasilkan respon yang diinginkan. Selanjutnya variabel optimasi tersebut digunakan untuk membentuk matrik inisial Negara (*Countries*) berukuran jumlah Negara dan banyaknya variabel optimasi (50 x 3) menggunakan fungsi *Generate new Country* pada program matlab.

Kemudian dilakukan evaluasi fungsi objektif (ITAE) dengan menggunakan Inisialisasi Negara yang telah dibentuk,

masing-masing nilai Kp, Ki, dan Kd pada baris data ke-n sampai data ke-50 secara bergantian dimasukkan kedalam sistem kontrol yang dirancang sehingga mendapatkan inisial *cost* (nilai ITAE sistem kontrol) berukuran matrik (50x1). Inisial Cost dan Inisial Negara disusun berdasarkan *cost* nya, nilai *cost* yang lebih baik berada diposisi yang lebih atas. Data ke-1 hingga data ke-5 nilai inisial cost dan Inisial Negara adalah *cost* dan posisi imperialis, sedangkan data ke-6 sampai data ke-50 adalah *cost* dan posisi koloni.

Berdasarkan Inisial Negara dan Inisial *cost* yang diperoleh. Dapat diperoleh jumlah kerajaan (*empire*) yang dibentuk dengan fungsi *create initial empires*. Karena jumlah imperialis awal ditentukan sebanyak 5, maka terbentuk 5 kerajaan *empire*, yang masing-masing *empire* terdiri dari posisi imperialis, *cost* imperialis, posisi koloni, *cost* koloni, dan total *cost*. Pada masing-masing Empire ke-n, memiliki posisi dan *cost* imperialis data ke-n, sedangkan posisi dan *cost* koloni disebar ke setiap *empire* dengan jumlah yang berbeda-beda secara random berdasarkan kekuatan masing-masing *empire*.

- Algoritma Utama ICA

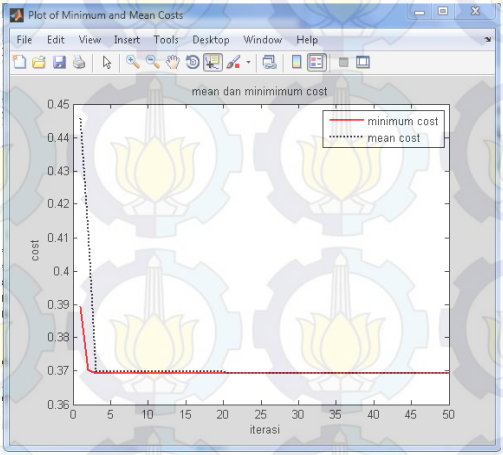
Setelah dibuat inisialisi *empire*, selanjutnya akan dilakukan proses algoritma utama yaitu menggunakan proses iterasi sebanyak jumlah nilai *decade* yang ditentukan. Proses ini dilakukan untuk mengevaluasi, menentukan posisi, dan perubahan koloni pada masing-masing empire. Proses evaluasi dan penentuan koloni pada masing-masing empire dilakukan berdasarkan fungsi *assimilate colonies* dan *revolve colonies*. Selanjutnya dihitung *cost* dan posisi koloni dengan fungsi evaluasi ITAE sistem. berdasarkan cost nya. Koloni dan imperialis saling bertukar posisi. Selanjutnya total cost pada tiap-tiap *empire* dihitung. Setelah diketahui nilai total *cost* pada tiap-tiap *empire* maka dilakukan proses kompetisi imperialis antar *empire*. Pada tahap ini digunakan beberapa

fungsi yaitu fungsi *posses* empire, *unite similar empires*, dan *imperialistic competition*,

Kompetisi imperialis berlangsung dengan diambilnya koloni terlemah dari *empire* terlemah oleh *empire* yang paling berpeluang untuk mengambilnya. Peluang kepemilikan ini didasarkan pada kekuatan *empire* tersebut terhadap total kekuatan semua *empire*. Jika terdapat *empire* yang tidak memiliki koloni maka *empire* ini akan tereliminasi dan jika ada *empire* yang *cost* nya sama maka kedua empire ini digabungkan. Sehingga akhirnya tersisa satu empire yang memiliki kekuatan yang terbesar dan memiliki banyak koloni.

- Konvergensi (Selesainya Optimasi)

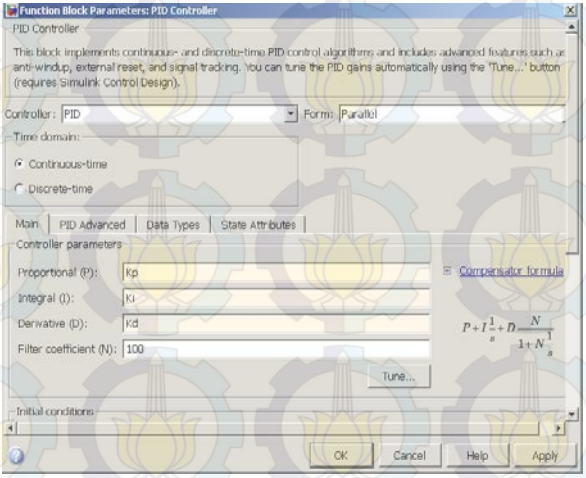
Berhentinya optimasi ini didasarkan pada jika dan hanya jika terdapat satu *empire* yang tersisa dan program dihentikan pada kondisi tersebut. Artinya telah terjadi konvergensi nilai *minimum cost* dan *mean cost*. Selain itu proses optimasi berhenti dikarenakan oleh selesainya proses iterasi sebanyak yang telah ditentukan yaitu 50 kali iterasi. Kemudian empire yang terkuat tadi menunjukkan nilai Kp, Ki, dan Kd hasil optimasi.



Gambar 3.7. Tampilan plot *minimum cost* dan *mean cost* ICA

Berdasarkan gambar 3.7, terlihat bahwa terjadi konvergensi nilai pada iterasi yang ke-21 dari 50 iterasi.

Pada tugas akhir ini, perancangan sistem kontrol PID berbasis ICA dilakukan dengan mengintegrasikan desain sistem kontrol pada simulink dan beberapa m.file program optimasi ICA. Proses optimasi dilakukan dengan software Matlab dan simulink 7.12.0 (R2011a). Program m.file terdiri dari 10 bagian yang berupa fungsi-fungsi yang membantu proses optimasi berdasarkan algoritma ICA. Proses integrasi simulink dengan program matlab (m.file) dilakukan dengan cara mengupload data simulink kedalam program utama ICA. Upload data simulink dilakukan dengan menuliskan listing program sim (‘nama simulasi simulink’) pada evaluasi nilai ITAE di program utama ICA yang menginputkan nilai Kp, Ki, Kd ke dalam simulink. Nilai Kp, Ki, dan Kd diinputkan kedalam Simulink dengan cara mensetting parameter *block* PID controller pada *Function Block Parameter*: PID *controller* berikut ini.



Gambar 3.8. Setting pada *Function Block Paramater*: *PID controller*

"Halaman ini memang dikosongkan"

BAB IV
ANALISA DATA DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai performansi perancangan sistem kontrol *pitch angle* turbin angin berbasis *Imperialist Competitive Algorithm* (ICA), pembahasan meliputi respon hasil pengujian pada setpoint, dan optimasi metode iterasi.

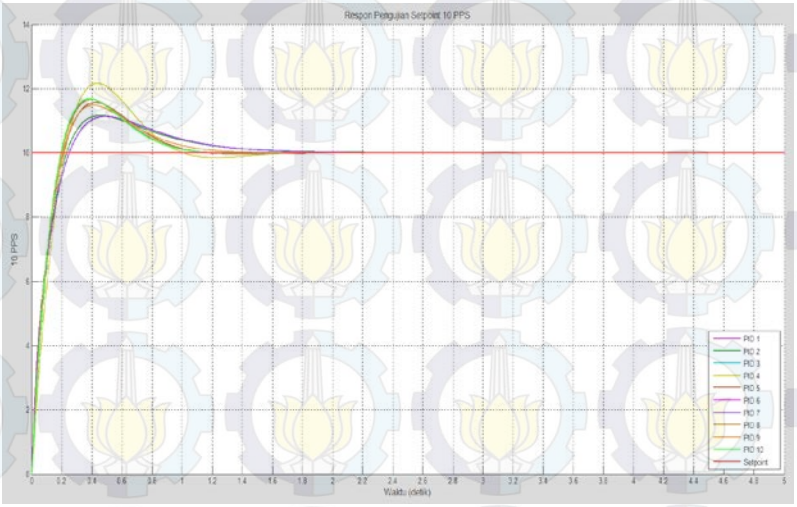
4.1 Respon Hasil Pengujian *Setpoint*

Pengujian *setpoint* ini dilakukan dengan tiga kondisi *set point* yang merepresentasikan variasi kecepatan putar rotor turbin angin, kecepatan putar rotor berupa nilai pulsa pps (*pulse per second*), pps adalah sejumlah pulsa yang dihasilkan oleh rangkaian *transmitter* kecepatan rotor turbin selama satu detik yang diterima oleh kontrol digital. *Pulse per second* (pps) digunakan untuk menghitung kecepatan putar turbin dalam satuan putaran per detik atau putaran per menit. Dalam hal ini nilai 1 pps setara dengan 3 rpm. *Setpoint* nilai pps berupa sinyal step yang ukurannya disesuaikan dengan nilai pps yang diinginkan. Tiga nilai setpoint adalah 10 pps, 20 pps, dan 40 pps, yang masing-masing ketiganya mewakili untuk kecepatan rendah, sedang, dan tinggi.

Setiap *setpoint* dilakukan pengambilan data sebanyak 10 kali pengambilan data menggunakan optimasi ICA. Sehingga nantinya terdapat perbandingan hasil untuk masing-masing 1 kali pengambilan data dan diambil respon terbaik untuk masing-masing uji *setpoint* berdasarkan data kuantitatif berupa nilai minimum ITAE , dan data kualitatif yang berdasarkan *maximum overshoot*, dan *settling time* kriteria 2% dari respon yang diberikan. *Maximum overshoot* adalah nilai puncak maksimal respon saat berusaha mencapai *setpoint*. *Settling time* adalah waktu yang dibutuhkan respon yang telah masuk ± 5% atau ± 2% dari respon *steady state*

4.1.1 Respon Hasil Uji *Setpoint* Sebesar 10 PPS

Berikut respon hasil uji *setpoint* 10 pps ini sebanyak 10 kali pengambilan data menggunakan optimasi ICA,

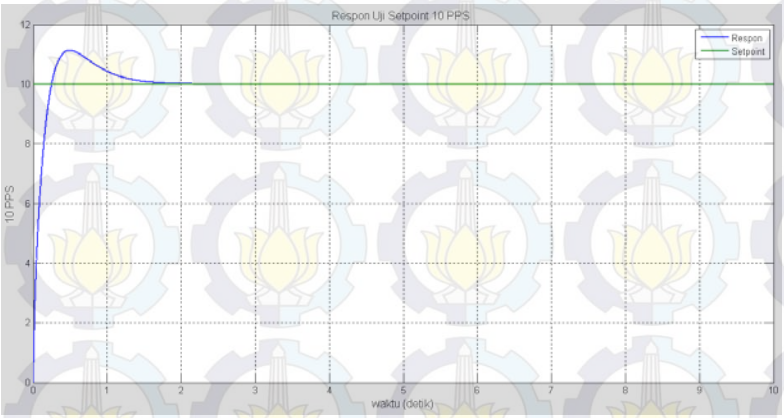


Gambar 4.1. 10 Respon hasil pengujian *setpoint* 10 pps

Tabel 4.1. Respon hasil uji *setpoint* 10 pps

Data-Ke	Parameter PID Kontrol			*Settling time*,t_s (detik)	*Maximum overshoot*, Mp (%)	ITAE
	Kp	Ki	Kd			
1	1.50	4.96	0.01	1.13	16.70	0.42
2	1.50	3.08	0.01	1.94	11.60	0.42
3	1.50	5.00	0.01	1.12	16.80	0.42
4	1.15	4.59	0.01	1.67	21.70	0.44
5	1.50	5.00	0.03	1.64	15.70	0.42
6	1.50	4.97	0.01	1.13	16.70	0.42
7	1.49	3.13	0.03	1.85	11.30	0.42
8	1.49	5.00	0.02	1.11	16.60	0.42
9	1.50	4.20	0.01	1.37	14.70	0.42
10	1.50	5.00	0.01	1.12	16.80	0.42

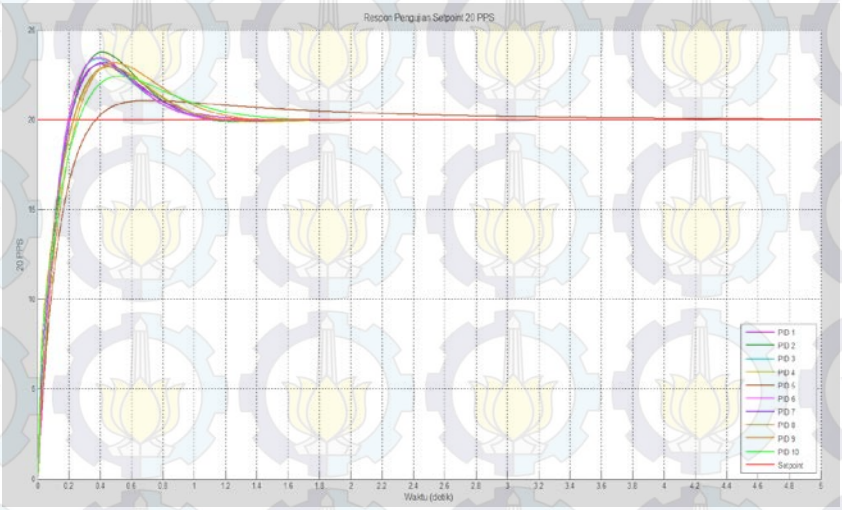
Berdasarkan gambar 4.1, menunjukan respon hasil pengujian 10 pps sebanyak 10 kali pengambilan data, optimasi ICA menghasilkan respon sistem yang sama namun berbeda dalam menghasilkan nilai *maximum overshoot* dan *settling time* nya. Berdasarkan tabel 4.1, optimasi ICA memberikan *best solution* berupa nilai parameter kontrol Kp, Ki, dan Kd pada setiap pengambilan data berdasarkan fungsi objektifnya, ITAE minimal (*best cost*) yang dapat diperoleh oleh ICA sebesar 0.42. Pengambilan data sebanyak 10 kali hampir sepenuhnya memberikan nilai minimal ITAE yang konsisten, terkecuali pada data ke-4 yang mengalami perbedaan dengan nilai minimal ITAE nya sebesar 0.44. Berdasarkan analisa kuantitatif nilai minimal ITAE dan analisa kualitatif pada respon sistem yang dihasilkan diperoleh nilai parameter PID kontrol yang optimal pada *setpoint* 10 pps dengan nilai Kp=1.49, Ki= 3.13 dan Kd= 0.03. Respon memberikan *maximum overshoot* sebesar 11. 30 % dan *settling time*nya 1.85 detik. Berikut respon sistem yang dihasilkan oleh parameter PID optimal.



Gambar 4.2. Respon optimal hasil uji *setpoint* 10 pps

4.1.2 Respon Hasil Uji *Setpoint* Sebesar 20 PPS

Setelah dilakukan pengambilan data sebanyak 10 kali pada *setpoint* 20 pps, berikut respon hasil pengujian *setpoint* 20 pps,

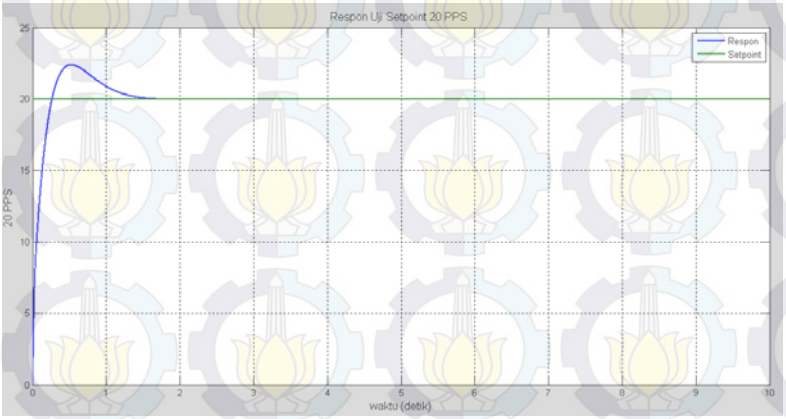


Gambar 4.3. 10 Respon hasil pengujian *setpoint* 20 pps

Tabel 4.2. Respon hasil uji *set point* 20 pps

Data-Ke	Parameter PID Kontrol			*Settling time*,t_s (detik)	*Maximum overshoot*, Mp (%)	ITAE
	Kp	Ki	Kd			
1	1.50	4.44	0.01	1.34	15.40	0.85
2	1.32	4.66	0.01	1.72	18.65	0.85
3	1.50	5.00	0.01	1.17	16.80	0.85
4	1.49	5.00	0.06	1.87	14.60	0.85
5	1.27	1.00	0.01	5.18	5.35	0.85
6	1.47	5.00	0.01	1.12	17.25	0.85
7	1.50	5.00	0.03	1.72	15.80	0.85
8	1.50	5.00	0.05	1.85	14.80	0.85
9	1.25	3.52	0.02	1.38	15.75	0.85
10	1.44	3.43	0.05	1.67	12.05	0.85

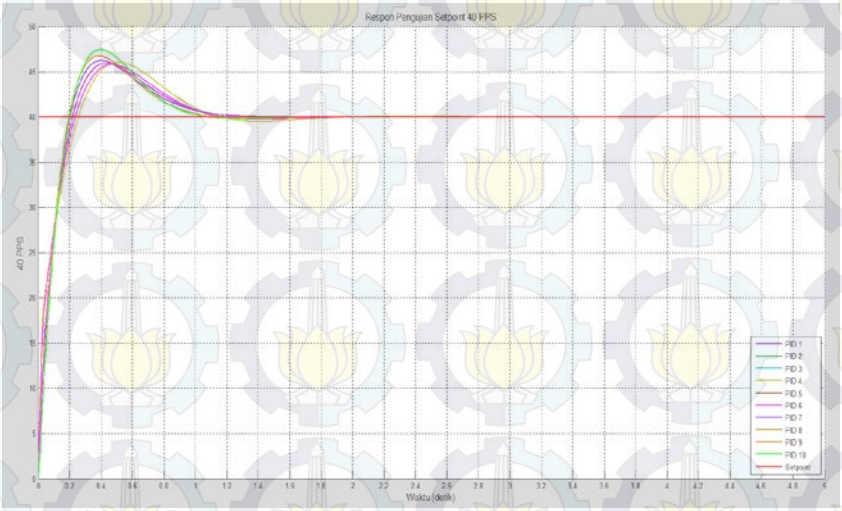
Berdasarkan gambar 4.2, terlihat respon hasil pengujian 20 pps sebanyak 10 kali pengambilan data, optimasi ICA menghasilkan respon sistem yang sama namun berbeda dalam menghasilkan nilai *maximum overshoot* dan *settling time* nya. Berdasarkan tabel 4.2, optimasi ICA memberikan *best solution* berupa nilai parameter kontrol Kp, Ki, dan Kd pada ITAE minimal (*best cost*) yang dapat diperoleh oleh ICA sebesar 0.85. Pengambilan data sebanyak 10 kali memberikan nilai minimal ITAE yang konsisten. Berdasarkan analisa kuantitatif nilai minimal ITAE dan analisa kualitatif pada respon sistem yang dihasilkan, diperoleh nilai parameter PID kontrol yang optimal pada *setpoint* 20 pps dengan nilai Kp=1.44, Ki= 3.43 dan Kd= 0.05. Respon memberikan *maximum overshoot* sebesar 12.05 % dan *settling time*nya 1.67 detik. Berikut respon sistem yang dihasilkan oleh parameter PID optimal.



Gambar 4.4. Respon optimal hasil uji *setpoint* 20 pps

4.1.3 Respon Hasil Uji *Setpoint* Sebesar 40 PPS

Pada *setpoint* 40 pps setelah dilakukan pengambilan data sebanyak 10 kali, diperoleh nilai parameter Kp, Ki, Kd dan respon yang diberikan oleh optimasi PID berbasis ICA.



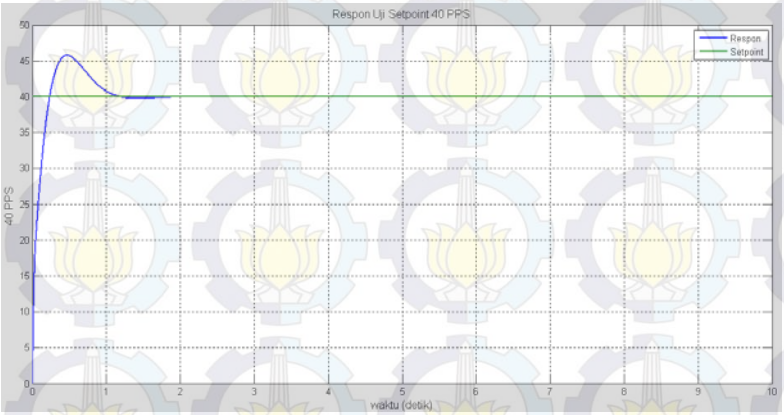
Gambar 4.5. 10 Respon hasil pengujian *setpoint* 40 pps

Tabel 4.3 Respon hasil uji *set point* 40 pps

Data-Ke	Parameter PID Kontrol			*Settling time*,t_s (detik)	*Maximum overshoot*, Mp (%)	ITAE
	Kp	Ki	Kd			
1	1.50	4.55	0.03	1.92	14.80	1.97
2	1.50	5.00	0.01	1.73	16.80	1.97
3	1.50	5.00	0.01	1.73	16.80	1.97
4	1.38	5.00	0.09	1.99	15.10	1.97
5	1.50	5.00	0.01	1.73	16.80	1.97
6	1.50	5.00	0.06	1.98	14.38	1.97
7	1.50	4.49	0.01	1.36	15.55	1.97
8	1.50	5.00	0.01	1.73	16.80	1.97
9	1.50	5.00	0.01	1.73	16.80	1.97
10	1.38	5.00	0.01	1.75	18.63	1.97

Berdasarkan gambar 4.5, terlihat respon hasil pengujian 40 pps sebanyak 10 kali pengambilan data, optimasi ICA

menghasilkan respon sistem yang sama namun berbeda dalam menghasilkan nilai *maximum overshoot* dan *settling time* nya. Berdasarkan tabel 4.3, terlihat optimasi ICA memberikan *best solution* berupa nilai parameter kontrol Kp, Ki, dan Kd pada ITAE minimal (*best cost*) yang dapat diperoleh oleh ICA sebesar 1.70. Pengambilan data sebanyak 10 kali memberikan nilai minimal ITAE yang konsisten. Berdasarkan analisa kuantitatif nilai minimal ITAE dan analisa kualitatif pada respon sistem yang dihasilkan, diperoleh nilai parameter PID kontrol yang optimal pada *setpoint* 40 pps dengan nilai Kp=1.5, Ki= 5.00 dan Kd= 0.06. Respon memberikan *maximum overshoot* sebesar 14.38 % dan *settling time*nya 1.98 detik. Berikut respon sistem yang dihasilkan oleh parameter PID optimal.



Gambar 4.6 Respon optimal hasil uji *setpoint* 40 pps

Berdasarkan hasil respon masing-masing *setpoint* uji yang diberikan selama waktu 10 detik untuk tiap simulasi, sistem kontrol yang telah dirancang memberikan respon sistem yang baik pada semua uji *setpoint* , dan memberikan hasil yang terbaik pada uji *setpoint* 10 pps, jika dibandingkan dengan *set point* yang lain, karena memberikan nilai *maximum overshoot* sebesar 11.30 % dan *settling time* (2%) 1.85 detik dengan nilai ITAE sebesar 0.42. sedangkan pada *setpoint* 20 pps menghasilkan respon yang

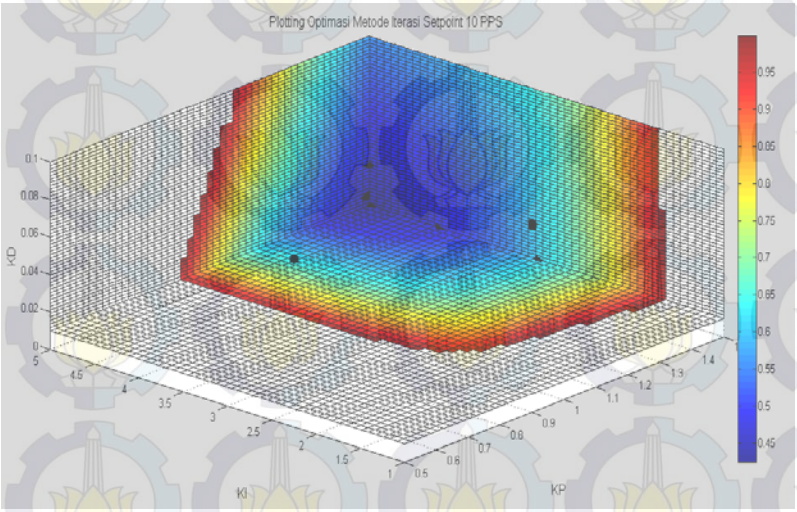
memiliki *maximum overshoot* sebesar 12.05 % dan *settling time* (2%) 1.67 detik dengan nilai ITAE sebesar 0.85, serta pada setpoint 40 pps, menghasilkan respon yang memiliki *maximum overshoot* sebesar 14.38 % dan *settling time* (2%) 1.98 detik dengan nilai ITAE sebesar 1.97. Pada *setpoint* 20 pps dan 40 pps, sistem kontrol memberikan respon yang memiliki *maximum overshoot* lebih besar pada uji *setpoint* yang lebih tingg hal ini dikarenakan pada waktu yang sama, error yang dihasilkan cukup besar sehingga nilai ITAE yang dihasilkan pada *setpoint* 20 pps dan 40 pps lebih besar jika dibandingkan dengan *setpoint* yang lebih kecil.

Perancangan kontrol PID dengan menggunakan *zighler Nichols* tidak dapat dilakukan pada sistem ini, baik itu menggunakan metode yang pertama kurva-s, maupun yang menggunakan metode osilasi. karena tidak semua sistem kontrol dapat dirancang dengan metode *zighler Nichols*. Sedangkan untuk perancangan kontrol PID dengan ICA, ICA mampu memberikan nilai parameter kontrol Kp, Ki, dan Kd pada batasan nilai tertentu, sehingga ICA merupakan salah satu metode optimasi heuristik (*heuristic optimization*) yang dapat digunakan untuk menala (*mentuning*) parameter kontrol dengan sebuah metode strategi optimasi berdasarkan fungsi evaluasinya yang dalam penelitian ini menggunakan ITAE (*Integral Time Absolute Error*).

4.2 Optimasi Metode Iterasi dan Optimasi ICA

Telah dilakukan pula pada penelitian ini sebuah metode optimasi sederhana yaitu metode iterasi. Metode iterasi ini dilakukan dengan membuat sebuah metrik berukuran 50x50x50 yang merepresentasikan banyaknya data berupa nilai Kp, Ki, dan Kd dengan batasan yang sama pada optimasi ICA dan dilakukan proses iterasi sebanyak 50 kali, selanjutnya hasil dari proses iterasi dilakukan *plotting* gambar menggunakan fungsi *plotting slice* yang ada pada matlab. *Slice* adalah fasilitas *plotting* volumetrik tiga dimensi, tujuan dilakukan ini adalah ini mengetahui persebaran nilai berdasarkan pada fungsi (persamaan) objektif yang sama dengan ICA berupa menghitung nilai ITAE

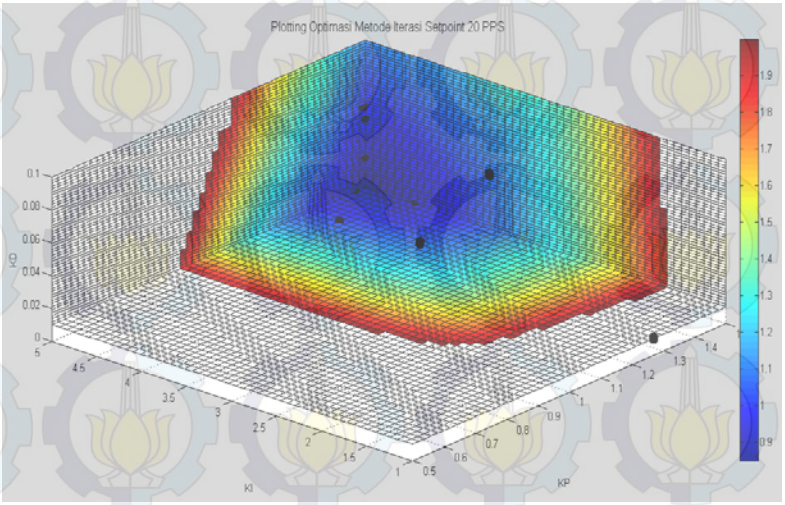
pada nilai Kp, Ki, dan Kd tertentu, sehingga kita memperoleh informasi nilai ITAE minimal dan maksimal dari data dan daerah nilai Kp, Ki, dan Kd yang dapat terlihat secara visual. Berikut hasil *plotting* slice hasil optimasi menggunakan metode iterasi pada masing-masing uji *setpoint* 10 pps, 20 pps, dan 40 pps.



Gambar 4.7 *Plotting slice* pada uji *set point* 10 PPS

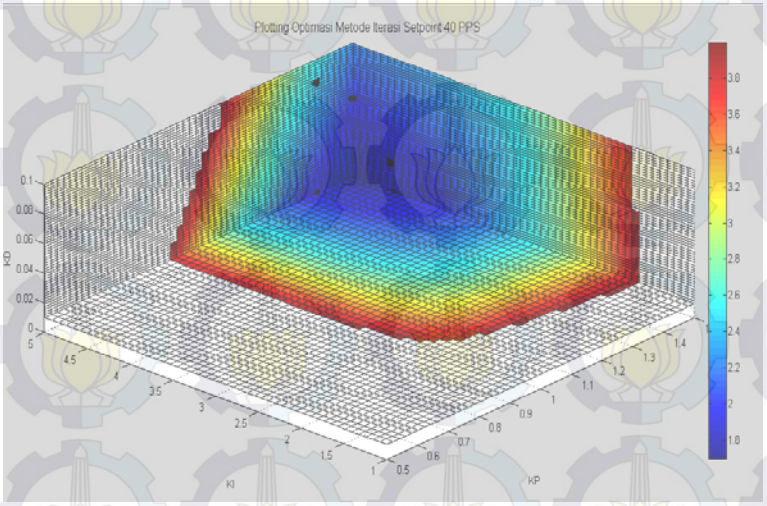
Berdasarkan gambar 4.7 diatas, nilai minimal ITAE hasil optimasi metode iterasi *setpoint* 10 pps pada nilai 0.42 dengan nilai Kp= 1.5, Ki=5, dan Kd=0.01, nilai ITAE minimal tersebar pada daerah yang berwarna biru. Untuk titik-titik yang berwarna hitam yang tersebar pada gambar merupakan *plotting* nilai Kp, Ki, dan Kd yang diperoleh dari optimasi ICA pada *setpoint* 10 pps. Berdasarkan persebarannya nilai Kp, Ki, dan Kd yang diberikan oleh optimasi berada pada daerah minimal, walaupun nilainya tidak selalu tetap pada titik tertentu, tetapi masih berada pada area yang menghasilkan nilai ITAE minimal. Dengan demikian ICA mampu memberikan solusi yang memungkinkan mendekati nilai minimal yang menjadikan solusi terbaik (*best*

cost) berupa nilai minimum ITAE dan *best solution* berupa nilai Kp, Ki, dan Kd.



Gambar 4.8 *Plotting slice* pada uji *set point* 20 PPS

Berdasarkan gambar 4.8 diatas, nilai minimal ITAE hasil optimasi metode iterasi *setpoint* 20 pps pada nilai 0.85 dengan nilai Kp= 1.5, Ki=5, dan Kd=0.01, nilai ITAE minimal tersebar pada daerah yang berwarna biru. Untuk titik-titik yang berwarna hitam yang tersebar pada gambar merupakan *plotting* nilai Kp, Ki, dan Kd yang diperoleh dari optimasi ICA pada *setpoint* 20 pps. Berdasarkan persebarannya nilai Kp, Ki, dan Kd yang diberikan oleh optimasi berada pada daerah minimal, walaupun nilainya tidak selalu tetap pada titik tertentu dan terdapat satu yang berada diluar area minimal, tetapi secara keseluruhan data yang lain masih berada pada area yang menghasilkan nilai ITAE minimal. Dengan demikian ICA mampu memberikan solusi yang memungkinkan mendekati nilai minimal.



Gambar 4.9 *Plotting slice* pada uji *set point* 40 PPS

Berdasarkan gambar 4.9 diatas, nilai minimal ITAE hasil optimasi metode iterasi *setpoint* 40 pps pada nilai 1.70 dengan nilai Kp= 1.5, Ki=5, dan Kd=0.01, nilai ITAE minimal tersebar pada daerah yang berwarna biru. Untuk titik-titik yang berwarna hitam yang tersebar pada gambar merupakan *plotting* nilai Kp, Ki, dan Kd yang diperoleh dari optimasi ICA pada *setpoint* 40 pps. Berdasarkan persebarannya nilai Kp, Ki, dan Kd yang diberikan oleh optimasi berada pada daerah minimal, walaupun nilainya tidak selalu tetap pada titik tertentu, tetapi secara keseluruhan data masih berada pada area yang menghasilkan nilai ITAE minimal.

Berdasarkan gambar 4.7 hingga 4.9, terlihat bahwa nilai ITAE minimum tersebar didaerah yang berwarna biru. Sedangkan solusi hasil optimasi menggunakan ICA juga berada di daerah tersebut yang ditunjukkan oleh tanda titik hitam yang ada pada gambar. Meskipun ICA tidak mampu memberikan nilai data yang tepat pada titik tertentu, namun ICA mampu mendekatinya dengan memberikan hasil di daerah tersebut. Selain itu ICA memiliki kelebihan dalam proses pencariannya nilai minimum,

ICA tidak membutuhkan waktu yang relatif lama jika dibandingkan dengan optimasi metode iterasi, dengan menggunakan perangkat komputer yang sama, Proses Iterasi ICA mampu melakukan optimasi dalam waktu kurang lebih 5-10 detik, sedangkan pada metode optimasi iterasi diperlukan waktu kurang lebih 3-5 jam, tingkat kecepatan proses ini bergantung pada spesifikasi perangkat komputer yang digunakan. Optimasi metode Iterasi membutuhkan waktu yang lebih lama dikarenakan dalam prosesnya metode iterasi harus membentuk banyak data sejumlah ukuran matrik yang telah dibuat sesuai dengan parameter yang ditentukan. Sehingga dengan demikian optimasi ICA adalah metode optimasi heuristik yang dapat dijadikan referensi untuk proses optimasi, meskipun ICA memberikan nilai solusi terbaik yang memungkinkan sebagai nilai optimal.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah dilakukan penelitian tentang perancangan sistem kontrol *pitch angle* turbin angin berbasis *Imperialist competitive Algorithm* (ICA), diperoleh beberapa kesimpulan diantaranya:

1. Berdasarkan hasil pengujian pada berbagai variasi *setpoint*, sistem kontrol yang dirancang menggunakan PID yang dioptimasi menggunakan ICA mampu memberikan nilai parameter PID kontrol, berdasarkan nilai minimal fungsi objektif ITAE (*Integral Time Absolut Error*).
2. Ditinjau dari analisa kuantitatif berupa nilai minimal ITAE sistem dan analisa kualitatif yang didasarkan pada lonjakan maksimal dan *settling time* yang diberikan . Sistem kontrol PID berbasis ICA yang dirancang bekerja dengan baik pada semua uji *set point*, dan pada uji *setpoint* 10 pps, sistem kontrol yang dirancang memberikan hasil terbaik dengan nilai ITAE yang lebih kecil yaitu 0.42, dan menghasilkan respon yang memiliki lonjakan maksimal yang lebih kecil, yaitu 11.30% dan *settling time* 1.85 detik.
3. Berdasarkan visualisasi hasil optimasi metode iterasi, optimasi ICA sebagai metode optimasi, mampu memberikan solusi yang mendekati nilai optimalnya yang didasarkan pada nilai minimal fungsi objektifnya (ITAE).

5.2 Saran

1. Perancangan sistem kontrol dengan ICA selanjutnya dapat ditambahkan fungsi objektif (fungsi evaluasi) lainnya selain ITAE untuk memperbaiki respon sistem.

"Halaman ini memang dikosongkan"

DAFTAR PUSTAKA

____. 2010. Blueprit Pengelolaan Energi Nasional 2006-2025, **Kementrian Energi dan Sumber daya Mineral**. Pemerintah Republik Indonesia.

Musyafa'. A. 2012. "Comparative Analysis Of Small-Scale Wind Turbine Design for the Low Rate Wind Speed". **Asian Journal Of Natural & Applied Science**. Vol 1. No.3.

Munteanu, I., Bratcu, A.I., Cutululis, N.-A., Ceanga, E. 2008. **Optimal Control of Wind Energy Systems**. Springer

Zhang. J, Cheng. M, Chen. Z, Fu X. 2008. "Pitch Angle Control for Variable Speed Wind Turbines". Nanjing, China.

Harika.A, A. Musyafa', I. M. Y. Negara, I. Robandi. Pitch Angle Control of variabel Low Rtaed Speed Wind Turbine Using Fuzzy Logic Controller**. International Journal of Engineering & Technology IJET-IJENS** Vol:10 No:05.

Farid Ridha Muttaqin, Ali Musyafa'.2011. "Traceability of pitch angle position for small scale wind turbine to get a maximum energy extraction'. Surabaya: Institut Teknologi Sepuluh Nopember. **Workshop and annual meeting** University of Dar es Salaam Preliminary Program, 2011.

Sunarto.2012. Rancang Bangun Sistem Pengendalian Sudut *Pitch* Turbin Angin Horizontal Axis Berbasis *Particle Swarm Optimization*. Surabaya: Institut Teknologi Sepuluh Nopember, 2012.

Abbas. F.A.R, Abdulsada. M. A. 2010. "Simulation of Wind Turbine Speed Control by MATLAB". **International Journal of Computer and Electrical Engineering**, Vol 2.No.5, October, 2010 1793-8163.

Malinga. B, Sneckenberger. Jhon. E, Feliachi. Ali. 2003. "Modeling and Control of Wind Turbine as a Distributed Resource". **IEEE, Proceedings of the 35th**

Southeastern Symposium on System Theory, 16-18 March 2003, 108 - 112.

Gargari. Atashpaz. E, Lucas. C. 2007. "Imperilaist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition". **IEEE Congress on Evalutionary Computation (CEC 2007), 4661-4667**.

Erich Hau. 2006. **Wind turbines, Fundamental , Technologies, Aplication, Economics 2nd edition**. United Kingdom: Springer

Ogata. Katsuhiko. 1995. **System Dynamic Third Edition**. New-Jersey-USA: Pearson Education International Inc.

Tony Burton, Nick Jenkins, David Sharpe, Ervin Bossanyi. 2001. **Wind Energy Handbook**. New York: Jhon Willey and Sons, LTD, 2001.

LAMPIRAN-1

PERHITUNGAN NILAI K11, K12, K13, K21, K22, K23, K31, K32, K33

$$K11 = \left\{\frac{Kv_{op}{}^3}{R.\,\omega_{op}} * \left(0.44 - 0.0167\beta_{op}\right) * \frac{\pi R}{v_{op}\left(15 - 0.3\beta_{op}\right)} * \cos\left[\pi\left(\frac{\lambda_{op} - 3}{15 - 0.3\beta_{op}}\right)\right]\right\}$$

$$K12 = -\frac{Kv_{op}{}^3}{R\omega_{op}{}^2} * \left(0.44 - 0.0167\beta_{op}\right) * \sin\left[\pi\left(\frac{\lambda_{op} - 3}{15 - 0.3\beta_{op}}\right)\right]$$

$$K13 = -0.00184 * \left(\beta_{op} v_{op}{}^2 + \frac{3\beta_{op} v_{op}{}^3}{R\omega_{op}{}^2}\right) * K$$

$$K21 = \left(0.44 - 0.0167\beta_{op}\right) * \frac{3Kv_{op}{}^2}{R\omega_{op}} * \sin\left(\frac{\pi(\lambda_{op} - 3)}{15 - 0.3\beta_{op}}\right)$$

$$K22 = -\left(0.44 - 0.0167\beta_{op}\right) * \frac{Kv_{op}{}^3}{R\omega_{op}} * \frac{\pi\lambda_{op}}{v_{op}{}^2(15 - 0.3\beta_{op})} * \cos\left(\frac{\pi(\lambda_{op} - 3)}{15 - 0.3\beta_{op}}\right)$$

$$K23 = -0.00184 * K * \left(2v_{op}\beta_{op} - \frac{9\beta_{op} v_{op}}{\lambda_{op}}\right)$$

$$K31 = -\frac{0.0167Kv_{op}{}^2}{\lambda_{op}} * \sin\left[\pi\left(\frac{\lambda_{op} - 3}{15 - 0.3\beta_{op}}\right)\right]$$

$$K32 = \frac{0.0167 K {v_{op}}^2}{\lambda_{op}} * \left(0.44 - 0.0167\beta_{op}\right) * \left(0.3\pi\left(\frac{\lambda_{op} - 3}{\left(15 - 0.3\beta_{op}\right)^2}\right)\right) * \cos\left[\pi\left(\frac{\lambda_{op} - 3}{15 - 0.3\beta_{op}}\right)\right]$$

$$K33 = -\frac{-0.00184 K(\lambda_{op} - 3){v_{op}}^2}{\lambda_{op}}$$

LAMPIRAN-2
LISTING PROGRAM OPTIMASI ICA

PID_ICA_main_program_Turbin_Angin.m

```
clear all;
clc;
tic;
setpoint=10; % Disesuaikan berdasarkan setpoint
% parameter                kp   ki  kd
ProblemParams.VarMin = [0.5 1.0 0.01]; % Batas
bawah untuk tiap parameter, kolom pertama
parameter pertama, kolom kedua parameter kedua,
dst
ProblemParams.VarMax = [1.5 5.0 0.10]; %
Modifying the size of VarMin and VarMax to have
a general form
if numel(ProblemParams.VarMin)==1

ProblemParams.VarMin=repmat(ProblemParams.VarMin
,1,ProblemParams.NPar);

ProblemParams.VarMax=repmat(ProblemParams.VarMax
,1,ProblemParams.NPar);
end

ProblemParams.SearchSpaceSize =
ProblemParams.VarMax - ProblemParams.VarMin;

%% Algorithmic Parameter Setting
AlgorithmParams.NumOfCountries = 50;
% Number of initial countries.
AlgorithmParams.NumOfInitialImperialists = 5;
% Number of Initial Imperialists.
AlgorithmParams.NumOfAllColonies =
AlgorithmParams.NumOfCountries -
AlgorithmParams.NumOfInitialImperialists;
AlgorithmParams.NumOfDecades = 50;
AlgorithmParams.RevolutionRate = 0.3;
% Revolution is the process in which the socio-
```

```
political characteristics of a country change
suddenly.
AlgorithmParams.AssimilationCoefficient = 2;
% In the original paper assimilation coefficient
is shown by "beta".
AlgorithmParams.AssimilationAngleCoefficient =
0.5;  % In the original paper assimilation angle
coefficient is shown by "gama".
AlgorithmParams.Zeta = 0.02;
% Total Cost of Empire = Cost of Imperialist +
Zeta * mean(Cost of All Colonies);
AlgorithmParams.DampRatio = 0.99;
AlgorithmParams.StopIfJustOneEmpire = false;
% Use "true" to stop the algorithm when just one
empire is remaining. Use "false" to continue the
algorithm.
AlgorithmParams.UnitingThreshold = 0.02;
% The percent of Search Space Size, which
enables the uniting process of two Empires.

%% Display Setting
DisplayParams.PlotEmpires = false;    % "true"
to plot. "false" to cancel ploting.
if DisplayParams.PlotEmpires
    DisplayParams.EmpiresFigureHandle =
figure('Name','Plot of
Empires','NumberTitle','off');
    DisplayParams.EmpiresAxisHandle = axes;
end

DisplayParams.PlotCost = true;    % "true" to
plot. "false"
if DisplayParams.PlotCost
    DisplayParams.CostFigureHandle =
figure('Name','Plot of Minimum and Mean
Costs','NumberTitle','off');
    DisplayParams.CostAxisHandle = axes;
end
```

```
ColorMatrix = [1   0   0  ; 0 1   0    ; 0   0 1
; 1   1   0  ; 1   0 1    ; 0 1   1    ; 1 1 1
;
    0.5 0.5 0.5; 0 0.5 0.5  ; 0.5 0 0.5  ; 0.5
0.5 0  ; 0.5 0 0    ; 0 0.5 0    ; 0 0 0.5     ;
    1   0.5 1  ; 0.1*[1 1 1]; 0.2*[1 1 1];
0.3*[1 1 1]; 0.4*[1 1 1]; 0.5*[1 1 1]; 0.6*[1 1
1]];
DisplayParams.ColorMatrix = [ColorMatrix ;
sqrt(ColorMatrix)];

DisplayParams.AxisMargin.Min =
ProblemParams.VarMin;
DisplayParams.AxisMargin.Max =
ProblemParams.VarMax;

%% Creation of Initial Empires
InitialCountries =
GenerateNewCountry(AlgorithmParams.NumOfCountrie
s , ProblemParams);
% evaluasi untuk mendapatkan country awal
x=InitialCountries;
for jj = 1:size(x,1)
    Kp=x(jj,1);
    Ki=x(jj,2);
    Kd=x(jj,3);

    sim('bismillahplant_10')
    t=Amp.time;
    y=Amp.signals.values;
    y1 = y(:,1);
    h = t(2)-t(1);
    e = abs(y1-setpoint).*t;
    ITAE = (h/2)*sum([e(1);2*e(2:end-
1);e(end)]);

    z(jj,1) = ITAE;

end
```

```
InitialCost=z;

[InitialCost,SortInd] = sort(InitialCost);
% Sort the cost in assending order. The best
countries will be in higher places
InitialCountries = InitialCountries(SortInd,:);
% Sort the population with respect to their
cost.

Empires =
CreateInitialEmpires(InitialCountries,InitialCos
t,AlgorithmParams, ProblemParams);

%% Main Loop
MinimumCost =
repmat(nan,AlgorithmParams.NumOfDecades,1);
MeanCost =
repmat(nan,AlgorithmParams.NumOfDecades,1);

if DisplayParams.PlotCost
    axes(DisplayParams.CostAxisHandle);
    if
any(findall(0)==DisplayParams.CostFigureHandle)

h_MinCostPlot=plot(MinimumCost,'r','LineWidth',1
.5,'YDataSource','MinimumCost');
        hold on;

h_MeanCostPlot=plot(MeanCost,'k:','LineWidth',1.
5,'YDataSource','MeanCost');
        hold off;
        pause(0.05);
    end
end

for Decade = 1:AlgorithmParams.NumOfDecades
    AlgorithmParams.RevolutionRate =
AlgorithmParams.DampRatio *
AlgorithmParams.RevolutionRate;
```

```
    %     Remained =
AlgorithmParams.NumOfDecades - Decade
    for ii = 1:numel(Empires)
        %% Assimilation;  Movement of Colonies
Toward Imperialists (Assimilation Policy)
        Empires(ii) =
AssimilateColonies(Empires(ii),AlgorithmParams,P
roblemParams);

        %% Revolution;  A Sudden Change in the
Socio-Political Characteristics
        Empires(ii) =
RevolveColonies(Empires(ii),AlgorithmParams,Prob
lemParams);

        %% New Cost Evaluation
        x=Empires(ii).ColoniesPosition;
     for jj = 1:size(x,1)
            Kp=x(jj,1);
            Ki=x(jj,2);
            Kd=x(jj,3);


            sim('bismillahplant_10');
            t=Amp.time;
            y=Amp.signals.values;
            y1 = y(:,1);
            h = t(2)-t(1);
            e = abs(y1-setpoint).*t;
            ITAE = (h/2)*sum([e(1);2*e(2:end-
1);e(end)]);

        z(jj,1) = ITAE;

    end
    Empires(ii).ColoniesCost=z(size(x,1),1);
    %% Empire Possession
    Empires(ii) = PossesEmpire(Empires(ii));
```

```
    %% Computation of Total Cost for Empires
    Empires(ii).TotalCost =
Empires(ii).ImperialistCost +
AlgorithmParams.Zeta *
mean(Empires(ii).ColoniesCost);

end

%% Uniting Similiar Empires
Empires =
UniteSimilarEmpires(Empires,AlgorithmParams,Prob
lemParams);

%% Imperialistic Competition
Empires = ImperialisticCompetition(Empires);

if numel(Empires) == 1 &&
AlgorithmParams.StopIfJustOneEmpire
    break
end

%% Displaying the Results
DisplayEmpires(Empires,AlgorithmParams,ProblemPa
rams,DisplayParams);

ImerialistCosts = [Empires.ImperialistCost];
MinimumCost(Decade) = min(ImerialistCosts)
MeanCost(Decade) = mean(ImerialistCosts);

if DisplayParams.PlotCost
    refreshdata(h_MinCostPlot);
    refreshdata(h_MeanCostPlot);
    title('mean dan minimimum cost')
    xlabel('   iterasi')
    ylabel('   cost')
    legend('minimum cost','mean cost')
    drawnow;
    pause(0.01);
```

```
end
%     fprintf(1,'Iteration:%f\n,  Minimum Cost:
%d',Decade, MinimumCost(Decade));
disp(['Iteration: ',num2str(Decade)])
disp(['Minimum Cost:
',num2str(MinimumCost(Decade))])

end % End of Algorithm
BestCost = MinimumCost(end)
BestIndex = find(ImerialistCosts ==
min(ImerialistCosts)); BestIndex = BestIndex(1);
BestSolution =
Empires(BestIndex).ImperialistPosition
toc
```

CreateInitialEmpires.m

```
function Empires =
CreateInitialEmpires(InitialCountries,InitialCos
t,AlgorithmParams, ProblemParams)

AllImperialistsPosition =
InitialCountries(1:AlgorithmParams.NumOfInitialI
mperialists,:);
AllImperialistsCost =
InitialCost(1:AlgorithmParams.NumOfInitialImperi
alists,:);

AllColoniesPosition =
InitialCountries(AlgorithmParams.NumOfInitialImp
erialists+1:end,:);
AllColoniesCost =
InitialCost(AlgorithmParams.NumOfInitialImperial
ists+1:end,:);

if max(AllImperialistsCost)>0
    AllImperialistsPower = 1.3 *
max(AllImperialistsCost) - AllImperialistsCost;
else
```

```
    AllImperialistsPower = 0.7 *
max(AllImperialistsCost) - AllImperialistsCost;
end

AllImperialistNumOfColonies =
round(AllImperialistsPower/sum(AllImperialistsPo
wer) * AlgorithmParams.NumOfAllColonies);
AllImperialistNumOfColonies(end) =
AlgorithmParams.NumOfAllColonies -
sum(AllImperialistNumOfColonies(1:end-1));
RandomIndex =
randperm(AlgorithmParams.NumOfAllColonies);

Empires(AlgorithmParams.NumOfInitialImperialists
).ImperialistPosition = 0;

for ii =
1:AlgorithmParams.NumOfInitialImperialists
    Empires(ii).ImperialistPosition =
AllImperialistsPosition(ii,:);
    Empires(ii).ImperialistCost =
AllImperialistsCost(ii,:);
    R =
RandomIndex(1:AllImperialistNumOfColonies(ii));
RandomIndex(AllImperialistNumOfColonies(ii)+1:en
d);
    Empires(ii).ColoniesPosition =
AllColoniesPosition(R,:);
    Empires(ii).ColoniesCost =
AllColoniesCost(R,:);
    Empires(ii).TotalCost =
Empires(ii).ImperialistCost +
AlgorithmParams.Zeta *
mean(Empires(ii).ColoniesCost);
end

for ii = 1:numel(Empires)
    if numel(Empires(ii).ColoniesPosition) == 0
```

```
        Empires(ii).ColoniesPosition =
GenerateNewCountry(1,ProblemParams);
%
        Empires.ColoniesCost =
feval(ProblemParams.FunctionName,Empires.Colonie
sPosition);
    end
end
```

GenerateNewCountry.m

```
function NewCountry =
GenerateNewCountry(NumOfCountries,ProblemParams)

    VarMinMatrix =
repmat(ProblemParams.VarMin,NumOfCountries,1);
    VarMaxMatrix =
repmat(ProblemParams.VarMax,NumOfCountries,1);
    NewCountry = (VarMaxMatrix - VarMinMatrix)
.* rand(size(VarMinMatrix)) + VarMinMatrix;

end
```

DisplayEmpires.m

```
function
DisplayEmpires(Empires,AlgorithmParams,ProblemPa
rams,DisplayParams)

    if ~DisplayParams.PlotEmpires
        return;
    end


    if
~any(findall(0)==DisplayParams.EmpiresFigureHand
le)
        return;
```

```
    end


    if ProblemParams.NPar == 2
        figure(1)
        for ii = 1:numel(Empires)

plot(DisplayParams.EmpiresAxisHandle,Empires(ii)
.ImperialistPosition(1),Empires(ii).ImperialistP
osition(2),'p',...
                'MarkerEdgeColor','k',...

'MarkerFaceColor',DisplayParams.ColorMatrix(ii,:
),...

'MarkerSize',70*numel(Empires(ii).ColoniesCost)/
AlgorithmParams.NumOfAllColonies + 33);
            hold on
    end

    if  ProblemParams.NPar == 3
        figure(1)
        for ii = 1:numel(Empires)

plot3(DisplayParams.EmpiresAxisHandle,Empires(ii
).ImperialistPosition(1),Empires(ii).Imperialist
Position(2),Empires(ii).ImperialistPosition(3),'
p',...
                'MarkerEdgeColor','k',...

'MarkerFaceColor',DisplayParams.ColorMatrix(ii,:
),...

'MarkerSize',70*numel(Empires(ii).ColoniesCost)/
AlgorithmParams.NumOfAllColonies + 13);
            hold on


plot3(DisplayParams.EmpiresAxisHandle,Empires(ii
).ColoniesPosition(:,1),Empires(ii).ColoniesPosi
```

```
tion(:,2),Empires(ii).ColoniesPosition(:,3),'ok'
,...
                    'MarkerEdgeColor','k',...

'MarkerFaceColor',DisplayParams.ColorMatrix(ii,:
),...
                    'MarkerSize',8);
            end

            xlim([DisplayParams.AxisMargin.Min(1)
DisplayParams.AxisMargin.Max(1)]);
            ylim([DisplayParams.AxisMargin.Min(2)
DisplayParams.AxisMargin.Max(2)]);
            zlim([DisplayParams.AxisMargin.Min(3)
DisplayParams.AxisMargin.Max(3)]);
            hold off
        end

        pause(0.05);
    end
```

AssimilateColonies.m

```
function TheEmpire =
AssimilateColonies(TheEmpire,AlgorithmParams,Pro
blemParams)

% for i = 1:numel(Imperialists)
%     Imperialists{i}.Number_of_Colonies_matrix
= [Imperialists{i}.Number_of_Colonies_matrix
Imperialists{i}.Number_of_Colonies];
%
%     Imperialists_cost_matrix (i) =
Imperialists{i}.cost_just_by_itself;
%
%     Imperialists_position_matrix(i,:) =
Imperialists{i}.position;
```

```
NumOfColonies =
size(TheEmpire.ColoniesPosition,1);

Vector =
repmat(TheEmpire.ImperialistPosition,NumOfColoni
es,1)-TheEmpire.ColoniesPosition;
d=Vector;
beta=AlgorithmParams.AssimilationCoefficient;
TheEmpire.ColoniesPosition =
TheEmpire.ColoniesPosition + 2 * beta *
rand(size(d)) .* d;
% TheEmpire.ColoniesPosition =
TheEmpire.ColoniesPosition + beta *
rand(size(d)) .* d;
% TheEmpire.ColoniesPosition =
TheEmpire.ColoniesPosition + 2 *
AlgorithmParams.AssimilationCoefficient *
rand(size(Vector)) .* Vector;

MinVarMatrix =
repmat(ProblemParams.VarMin,NumOfColonies,1);
MaxVarMatrix =
repmat(ProblemParams.VarMax,NumOfColonies,1);

TheEmpire.ColoniesPosition=max(TheEmpire.Colonie
sPosition,MinVarMatrix);
TheEmpire.ColoniesPosition=min(TheEmpire.Colonie
sPosition,MaxVarMatrix);
```

RevolveColonies.m

```
function TheEmpire =
RevolveColonies(TheEmpire,AlgorithmParams,Proble
mParams)
    NumOfRevolvingColonies =
round(AlgorithmParams.RevolutionRate *
numel(TheEmpire.ColoniesCost));
    RevolvedPosition =
GenerateNewCountry(NumOfRevolvingColonies ,
ProblemParams);
```

```
    R = randperm(numel(TheEmpire.ColoniesCost));
    R = R(1:NumOfRevolvingColonies);
    TheEmpire.ColoniesPosition(R,:) =
RevolvedPosition;
end
```

PossesEmpire.m

```
function TheEmpire = PossesEmpire(TheEmpire)

    ColoniesCost = TheEmpire.ColoniesCost;

    [MinColoniesCost
BestColonyInd]=min(ColoniesCost);
    if MinColoniesCost <
TheEmpire.ImperialistCost

        OldImperialistPosition =
TheEmpire.ImperialistPosition;
        OldImperialistCost =
TheEmpire.ImperialistCost;

        TheEmpire.ImperialistPosition =
TheEmpire.ColoniesPosition(BestColonyInd,:);
        TheEmpire.ImperialistCost =
TheEmpire.ColoniesCost(BestColonyInd);


TheEmpire.ColoniesPosition(BestColonyInd,:) =
OldImperialistPosition;
        TheEmpire.ColoniesCost(BestColonyInd) =
OldImperialistCost;
    end

end
```

UniteSimilarEmpires.m

```
function
Empires=UniteSimilarEmpires(Empires,AlgorithmPar
ams,ProblemParams)

    ThereshholdDistance =
AlgorithmParams.UnitingThreshold *
norm(ProblemParams.SearchSpaceSize);
    NumOfEmpires = numel(Empires);

    for ii = 1:NumOfEmpires-1
        for jj = ii+1:NumOfEmpires
            DistanceVector =
Empires(ii).ImperialistPosition -
Empires(jj).ImperialistPosition;
            Distance = norm(DistanceVector);
            if Distance<=ThereshholdDistance
                if Empires(ii).ImperialistCost <
Empires(jj).ImperialistCost
                    BetterEmpireInd=ii;
                    WorseEmpireInd=jj;
                else
                    BetterEmpireInd=jj;
                    WorseEmpireInd=ii;
                end


Empires(BetterEmpireInd).ColoniesPosition =
[Empires(BetterEmpireInd).ColoniesPosition

Empires(WorseEmpireInd).ImperialistPosition

Empires(WorseEmpireInd).ColoniesPosition];


Empires(BetterEmpireInd).ColoniesCost =
[Empires(BetterEmpireInd).ColoniesCost

Empires(WorseEmpireInd).ImperialistCost

Empires(WorseEmpireInd).ColoniesCost];
```

```
                % Update TotalCost for new
United Empire

Empires(BetterEmpireInd).TotalCost =
Empires(BetterEmpireInd).ImperialistCost +
AlgorithmParams.Zeta *
mean(Empires(BetterEmpireInd).ColoniesCost);

                Empires =
Empires([1:WorseEmpireInd-1
WorseEmpireInd+1:end]);
                return;
            end

        end
    end

end
```

myrandint.m

```
function y = myrandint(MaxInt,m,n)
%function y = myrandint(MaxInt,m,n)
% This functions creates random numbers between
[1 MaxInt] (1 itself and MaxInt itself)    s
if nargin == 1
    y = ceil(rand*MaxInt);
elseif nargin == 3
    y = ceil(rand(m,n)*MaxInt);
else
    warning('Incorrect Number of Inputs');
end
```

ImperialisticCompetition.m

```
function
Empires=ImperialisticCompetition(Empires)
    if rand > .11
```

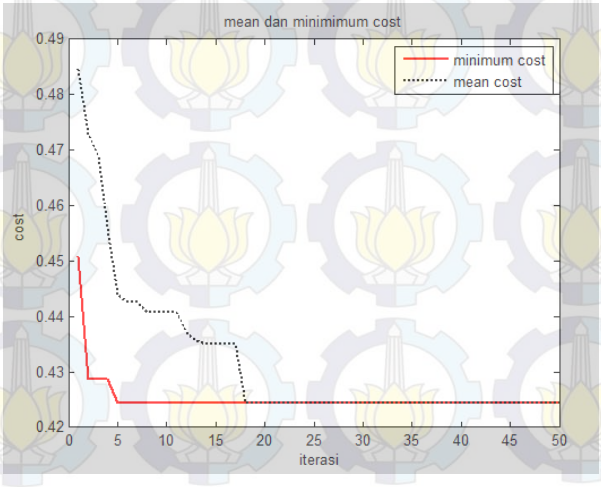
```
        return
    end
    if numel(Empires)<=1
        return;
    end

    TotalCosts = [Empires.TotalCost];
    [MaxTotalCost WeakestEmpireInd] =
max(TotalCosts);
    TotalPowers = MaxTotalCost - TotalCosts;
    PossessionProbability = TotalPowers /
sum(TotalPowers);

    SelectedEmpireInd =
SelectAnEmpire(PossessionProbability);

    nn =
numel(Empires(WeakestEmpireInd).ColoniesCost);
    jj = myrandint(nn,1,1);

    Empires(SelectedEmpireInd).ColoniesPosition
= [Empires(SelectedEmpireInd).ColoniesPosition

Empires(WeakestEmpireInd).ColoniesPosition(jj,:)
];

    Empires(SelectedEmpireInd).ColoniesCost =
[Empires(SelectedEmpireInd).ColoniesCost

Empires(WeakestEmpireInd).ColoniesCost(jj)];

    Empires(WeakestEmpireInd).ColoniesPosition =
Empires(WeakestEmpireInd).ColoniesPosition([1:jj
-1 jj+1:end],:);
    Empires(WeakestEmpireInd).ColoniesCost =
Empires(WeakestEmpireInd).ColoniesCost([1:jj-1
jj+1:end],:);
```

```
    %% Collapse of the the weakest colony-less
Empire
    nn =
numel(Empires(WeakestEmpireInd).ColoniesCost);
    if nn<=1

Empires(SelectedEmpireInd).ColoniesPosition =
[Empires(SelectedEmpireInd).ColoniesPosition

Empires(WeakestEmpireInd).ImperialistPosition];

        Empires(SelectedEmpireInd).ColoniesCost
= [Empires(SelectedEmpireInd).ColoniesCost

Empires(WeakestEmpireInd).ImperialistCost];

        Empires=Empires([1:WeakestEmpireInd-1
WeakestEmpireInd+1:end]);
    end

end

function Index = SelectAnEmpire(Probability)
    R = rand(size(Probability));
    D = Probability - R;
    [MaxD Index] = max(D);
end
```

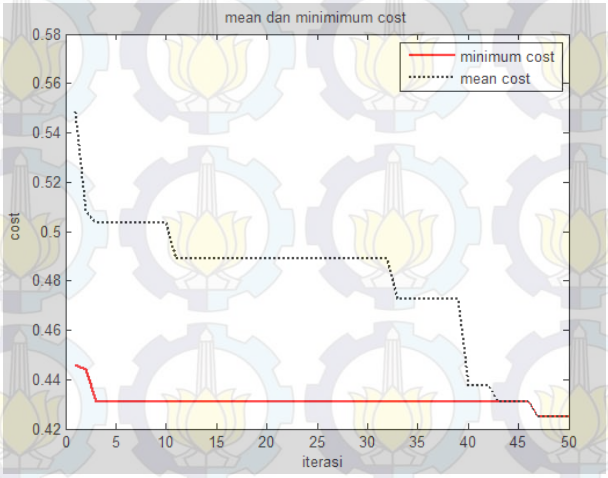
Program diperoleh freeware dari Gargari Atasphaz di *www.mathworks.com*, dan dilakukan modifikasi pada main programnya oleh penulis.

"Halaman ini memang dikosongkan"

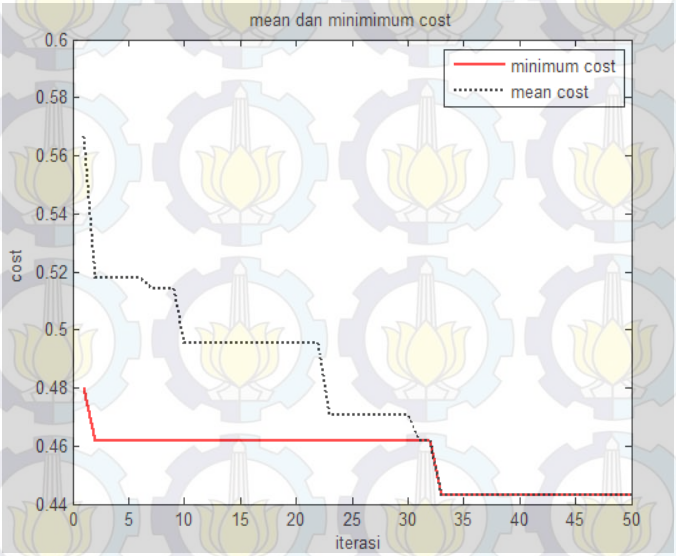
LAMPIRAN-3
KONVERGENSI OPTIMASI ICA



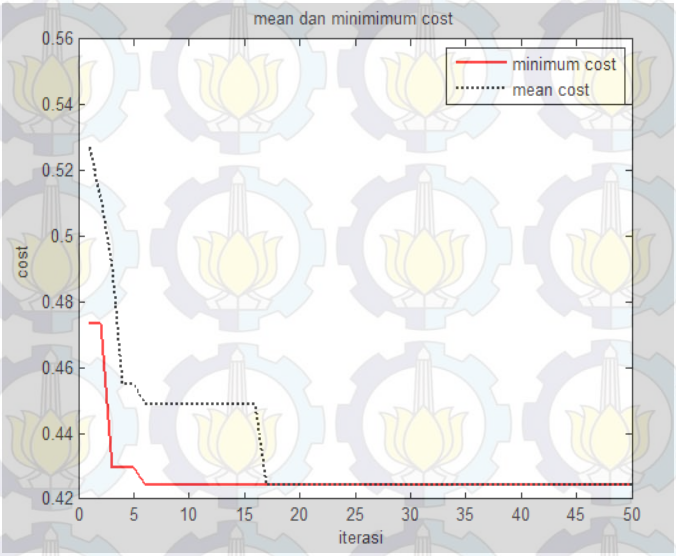
Konvergensi data ke-1 *setpoint* 10 pps



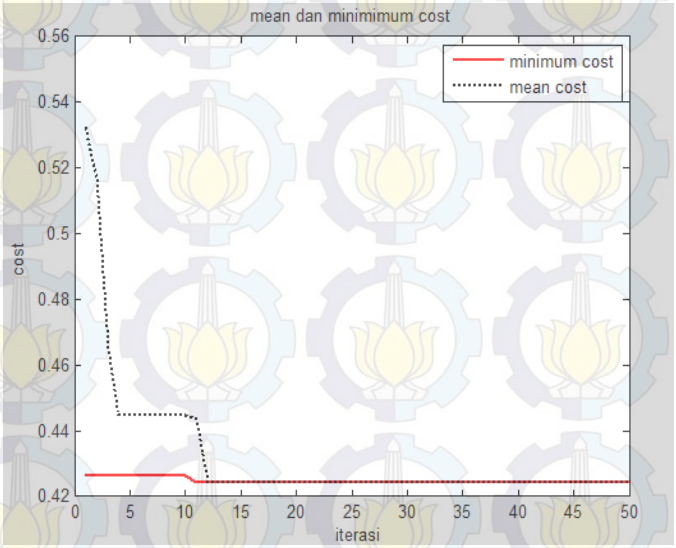
Konvergensi data ke-2 *setpoint* 10 pps



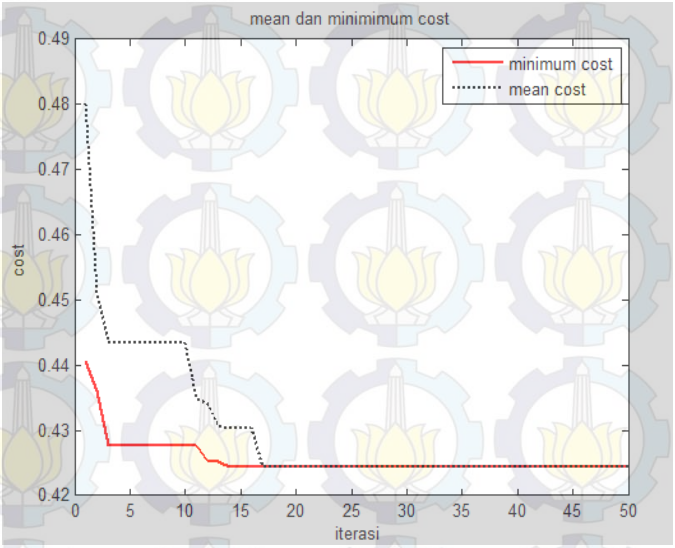
Konvergensi data ke-3 *setpoint* 10 pps



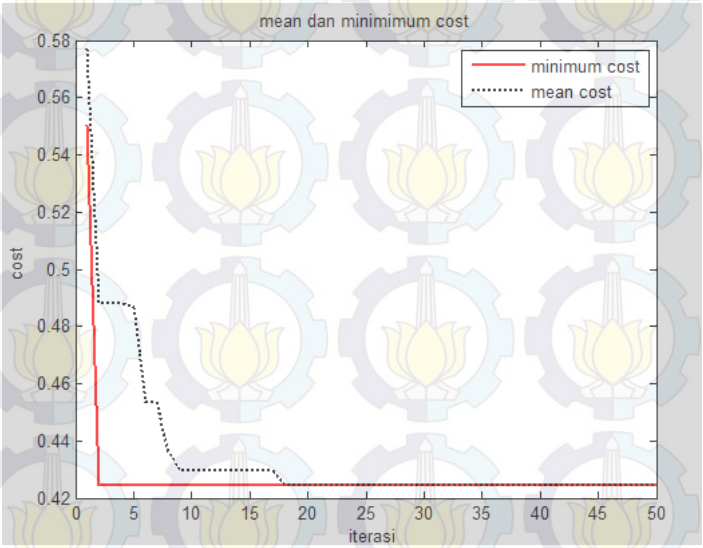
Konvergensi data ke-4 *setpoint* 10 pps



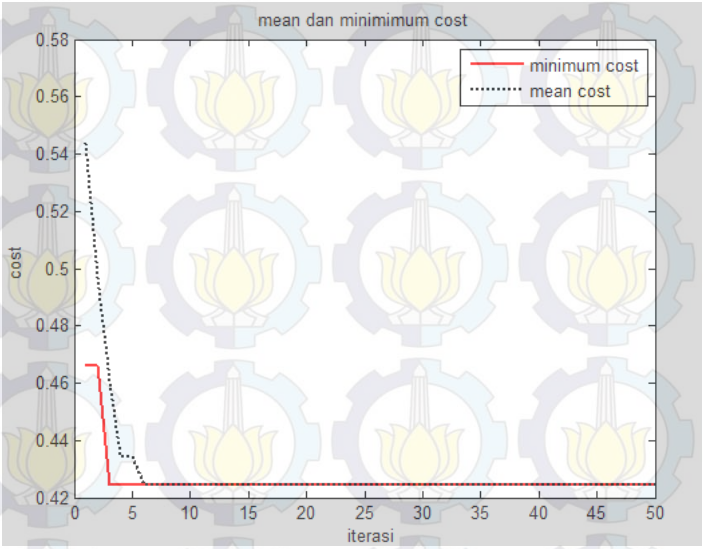
Konvergensi data ke-5 *setpoint* 10 pps



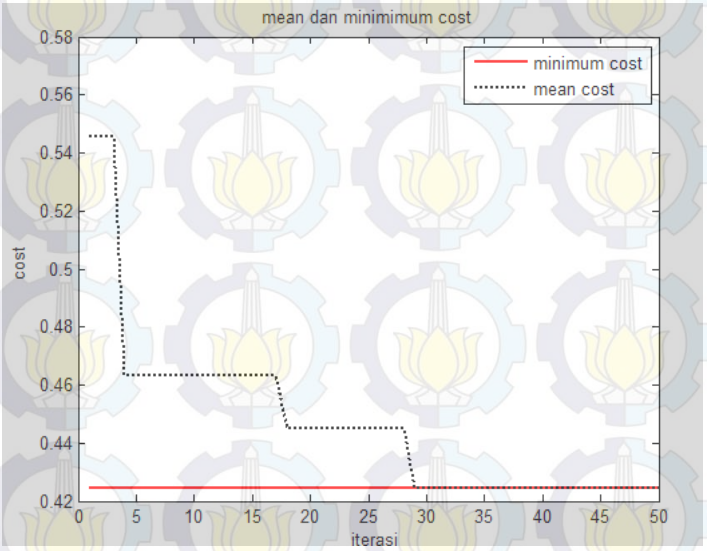
Konvergensi data ke-6 *setpoint* 10 pps



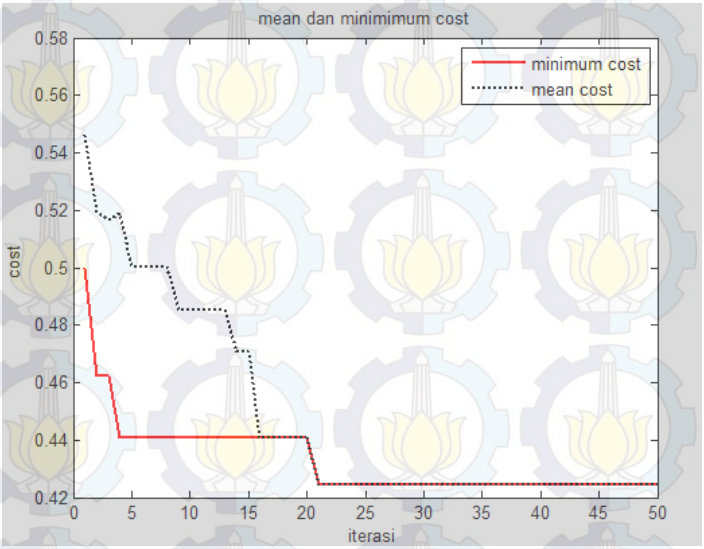
Konvergensi data ke-7 *setpoint* 10 pps



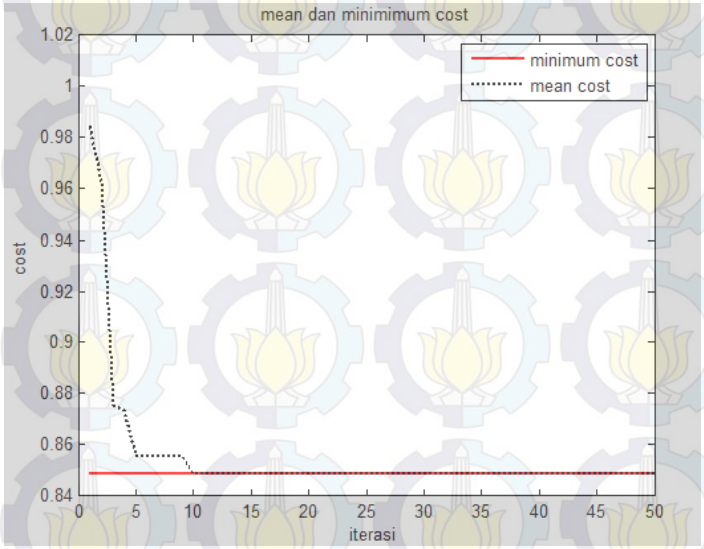
Konvergensi data ke-8 *setpoint* 10 pps



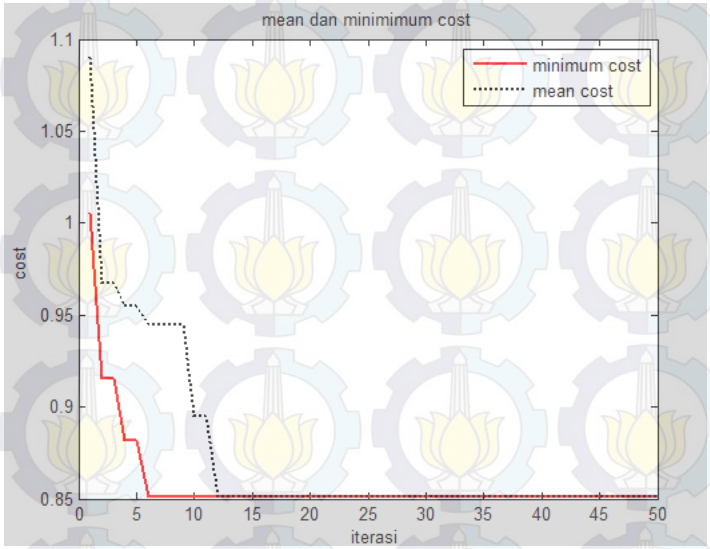
Konvergensi data ke-9 *setpoint* 10 pps



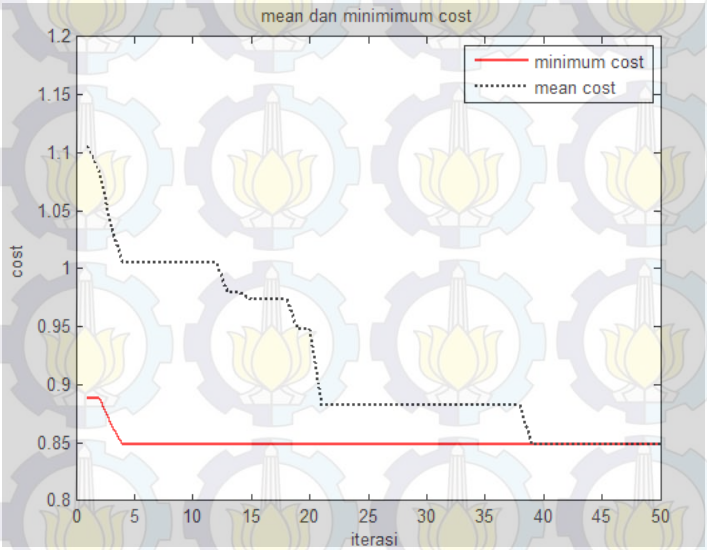
Konvergensi data ke-10 *setpoint* 10 pps



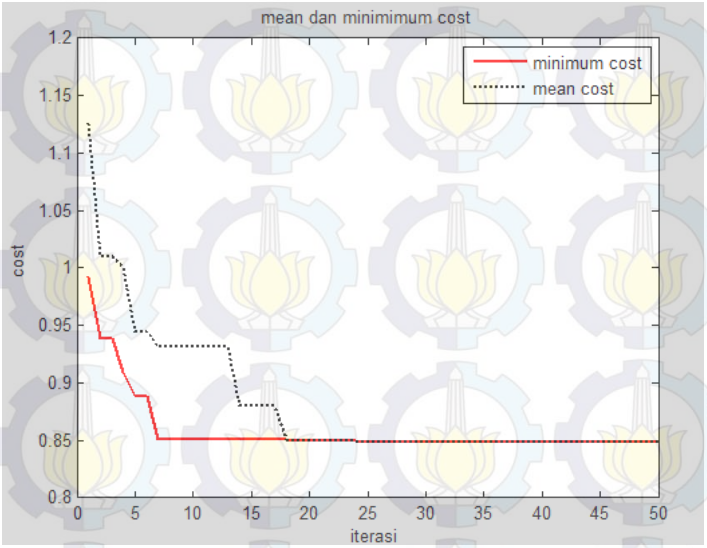
Konvergensi data ke-1 *setpoint* 20 pps



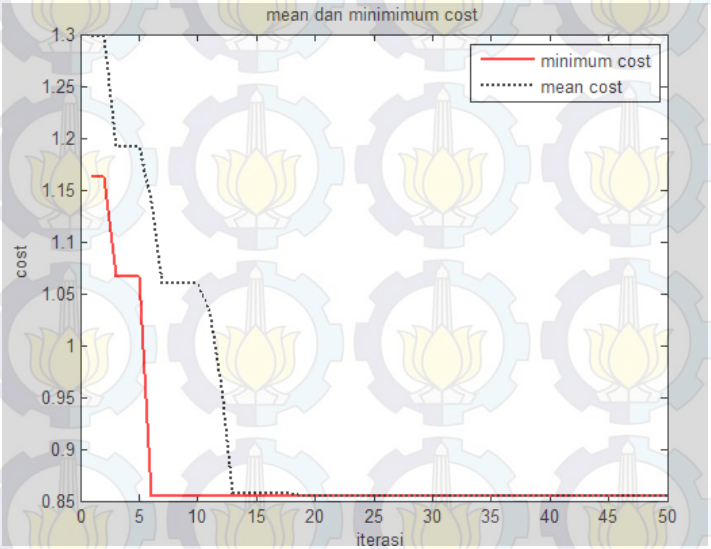
Konvergensi data ke-2 *setpoint* 20 pps



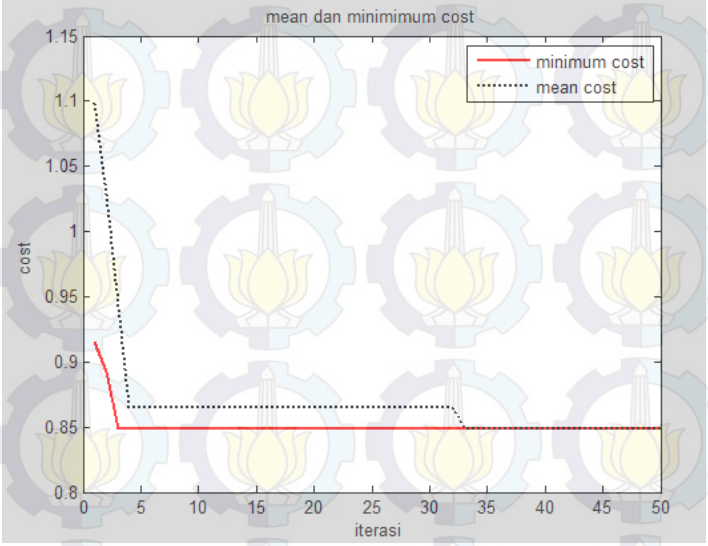
Konvergensi data ke-3 *setpoint* 20 pps



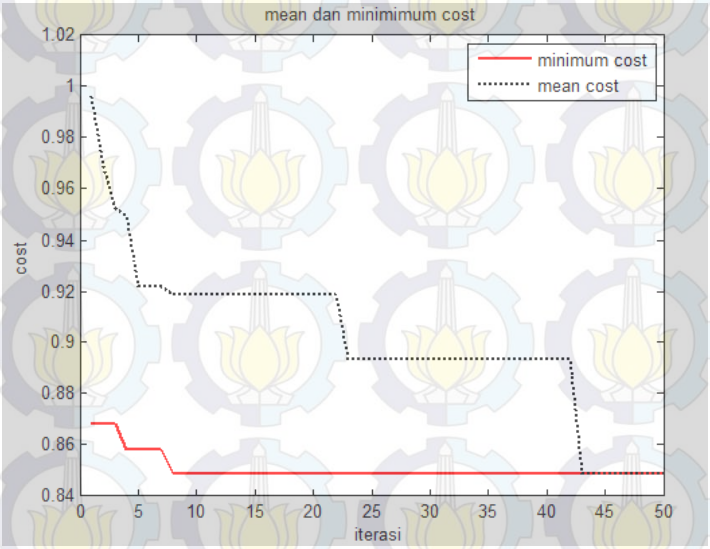
Konvergensi data ke-4 *setpoint* 20 pps



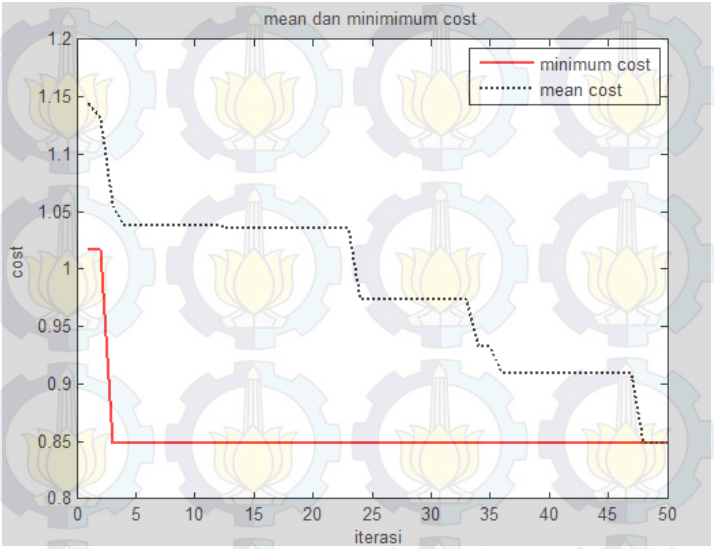
Konvergensi data ke-5 *setpoint* 20 pps



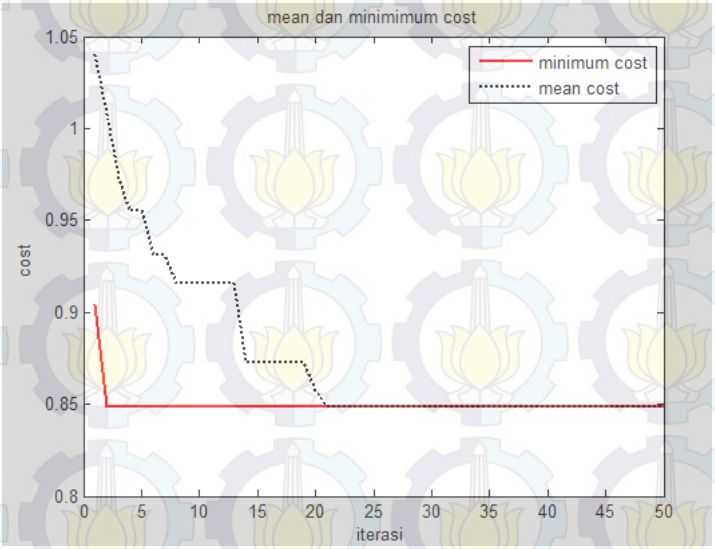
Konvergensi data ke-6 *setpoint* 20 pps



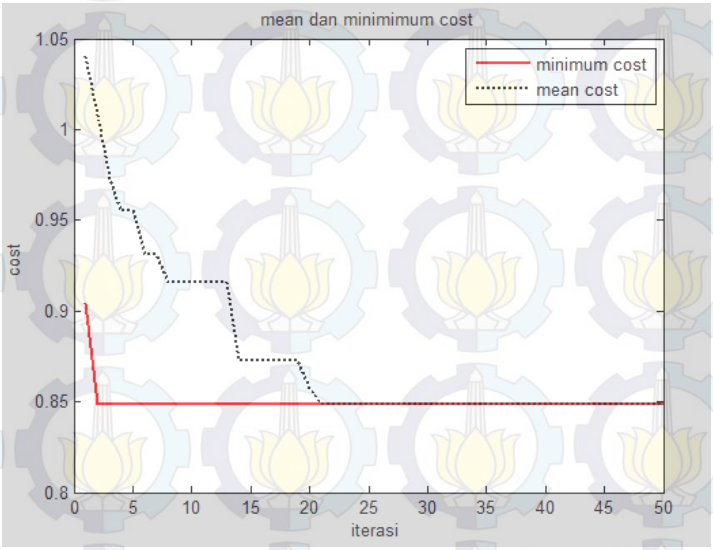
Konvergensi data ke-7 *setpoint* 20 pps



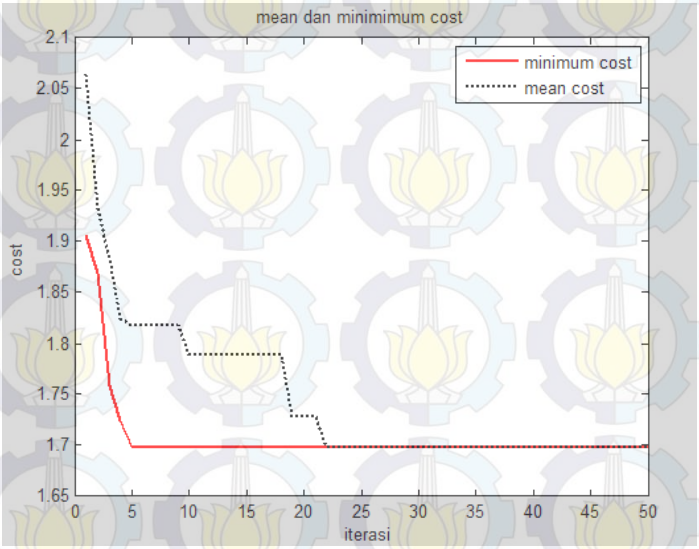
Konvergensi data ke-8 *setpoint* 20 pps



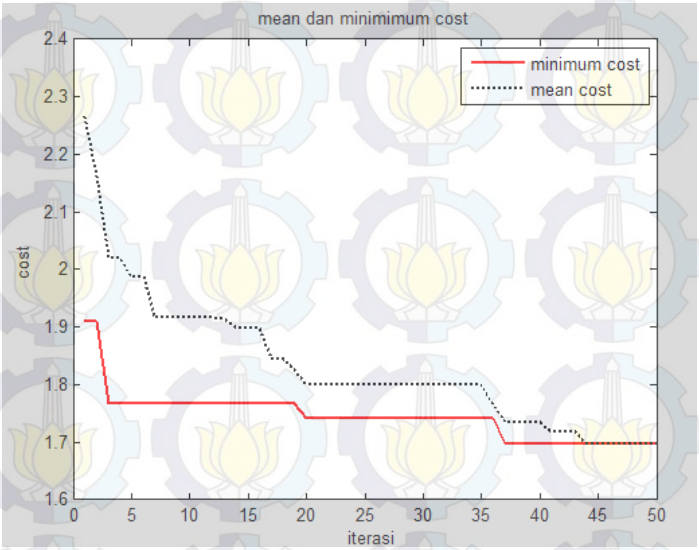
Konvergensi data ke-9 *setpoint* 20 pps



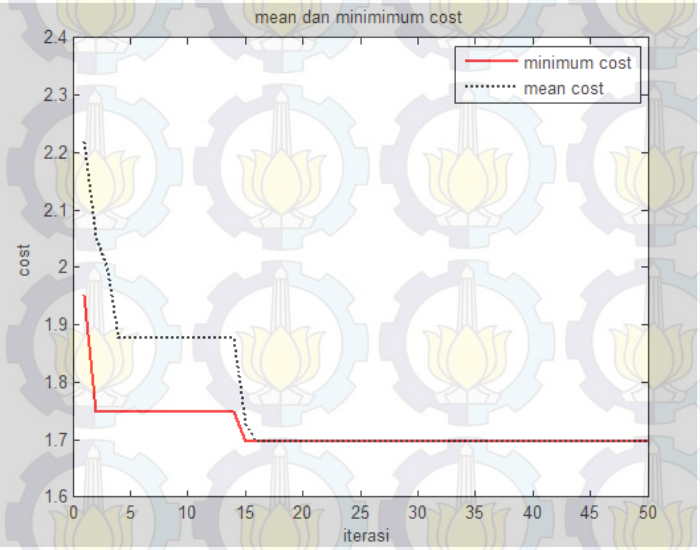
Konvergensi data ke-10 *setpoint* 20 pps



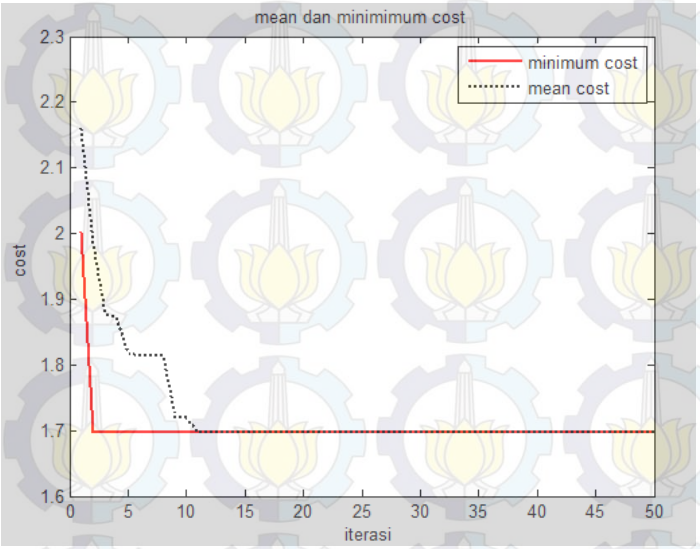
Konvergensi data ke-1 *setpoint* 40 pps



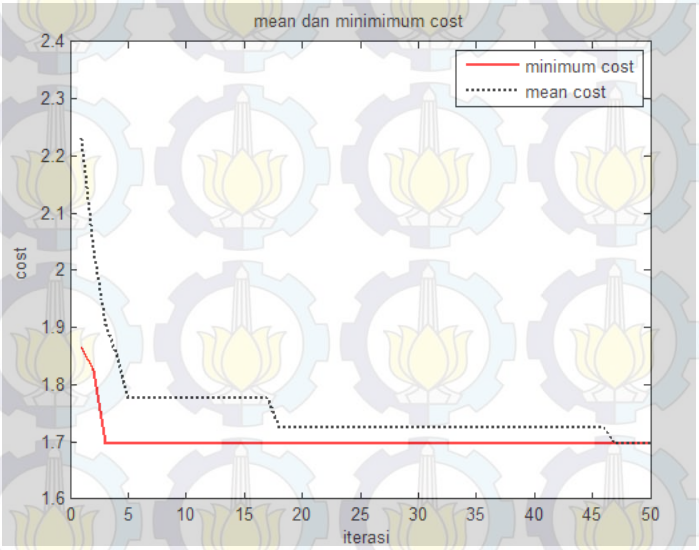
Konvergensi data ke-2 *setpoint* 40 pps



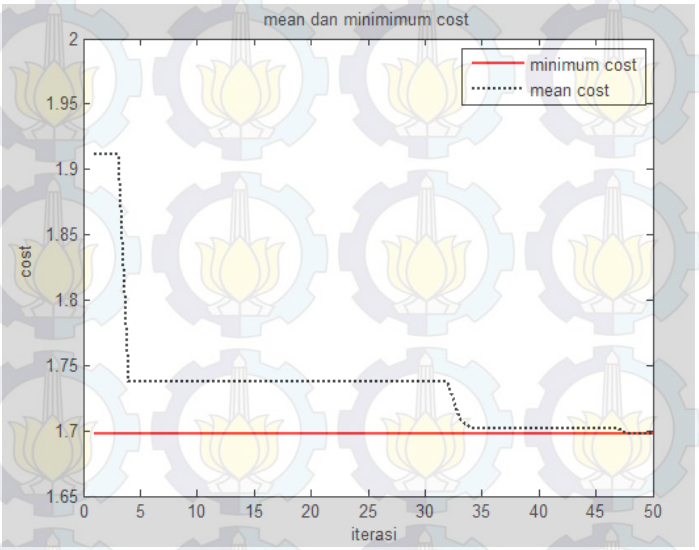
Konvergensi data ke-3 *setpoint* 40 pps



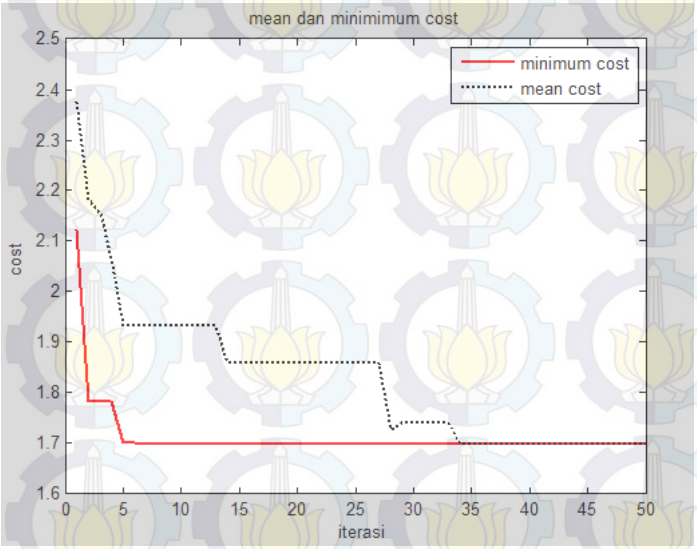
Konvergensi data ke-4 *setpoint* 40 pps



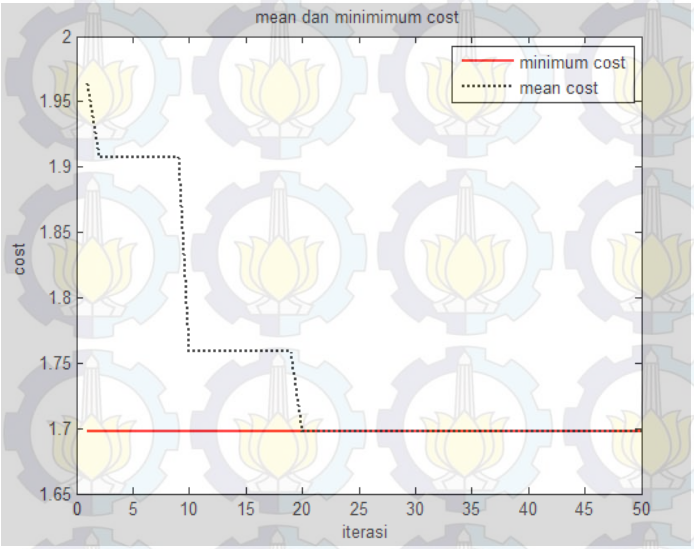
Konvergensi data ke-5 *setpoint* 40 pps



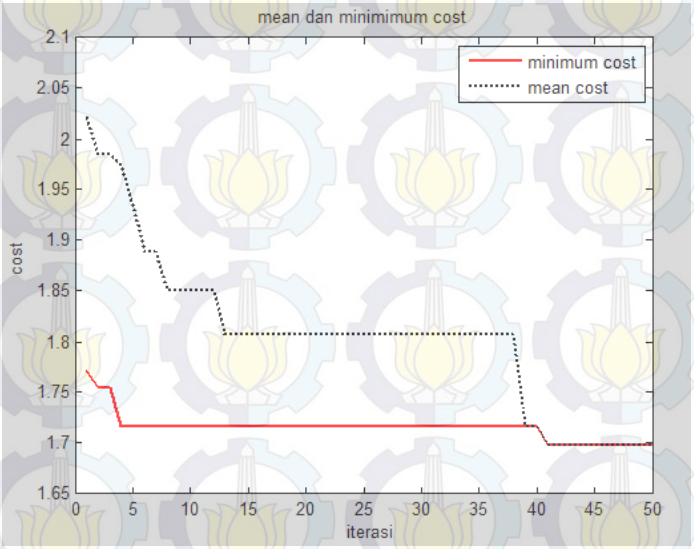
Konvergensi data ke-6 *setpoint* 40 pps



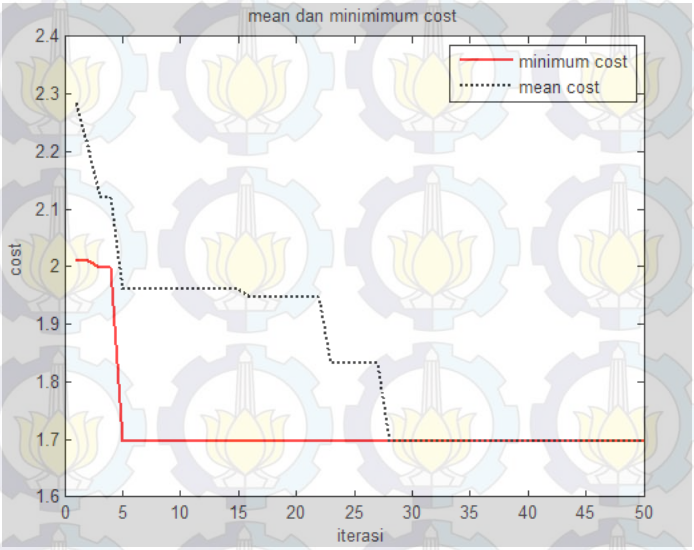
Konvergensi data ke-7 *setpoint* 40 pps



Konvergensi data ke-8 *setpoint* 40 pps



Konvergensi data ke-9 *setpoint* 40 pps



Konvergensi data ke-10 *setpoint* 40 pps

"Halaman ini memang dikosongkan"

BIODATA PENULIS

Akhmad Bakhrul Fauzi, lahir di Kabupaten Sidoarjo pada tanggal 06 Februati 1992, penulis adalah anak kedua dari pasangan Soleh Irsyad dan Nur Ulfiyah. Penulis sejak kecil di besarkan dan dididik oleh Bapak San Rois dan Ibu Mujiarti. Penulis telah menyelesaikan pendidikan formal dari SDN Keboharan, Krian, SMP Negeri 3 Krian, SMA Negeri 1 Krian, Kabupaten Sidoarjo, dan melanjutkan program pendidikan S1 di jurusan Teknik Fisika Institut Teknologi Sepuluh Nopember Surabaya melalui jalur beasiswa Bidik Misi. Pada saat kuliah penulis mengambil bidang minat studi Rekayasa Instrumentasi dan Kontrol. Selama masa kuliah penulis aktif dalam kegiatan organisasi diantaranya menjadi Staf Ahli Departement Sosial Masyarakat (SOSMAS) di Himpunan Mahasiswa Teknik Fisika, dan Menjadi Asisten Direktur bidang Pengembangan Sumber Daya Anggota (PSDA) Koperasi Mahasiswa (KOPMA) ITS. Selain itu penulis juga aktif sebagai Pemandu LKMM (Pemandu AMPLAS) di Fakultas Teknologi Industri.

Pengalaman yang berkesan selama masa studi adalah bersama-sama teman SOSMAS, mampu memberikan prestasi untuk SOSMAS HMTF sebagai COMDEV SOSMAS terbaik di ITS, dan mendapatkan kesempatan dalam mengikuti beberapa acara pertemuan pemuda-pemudi International, salah satunya di International Working Camp (IWOCA), Malang 2014, yang membahas mengenai *Sustainable Development in Agriculture* dan *Environmental Conservation*. Penulis memiliki hobi dalam kegiatan sosial, berorganisasi dan berpetualang. Penulis dapat dihubungi via email, bakhrul.fauzi@gmail.com.

"Halaman ini memang dikosongkan"