# OPTIMASI *RADIAL BASIS FUNCTION NEURAL NETWORK* MENGGUNAKAN *HYBRID PARTICLE SWARM OPTIMIZATION* DAN *GENETIC ALGORITHM* UNTUK PERAMALAN CURAH HUJAN

**SKRIPSI**

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Kentia Dea Hapsari
NIM: 115060801111084


INFORMATIKA / ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

# PENGESAHAN

OPTIMASI *RADIAL BASIS FUNCTION NEURAL NETWORK* MENGGUNAKAN *HYBRID PARTICLE SWARM OPTIMIZATION* DAN *GENETIC ALGORITHM* UNTUK PERAMALAN CURAH HUJAN

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Kentia Dea Hapsari
NIM: 115060801111084

Skripsi ini telah diuji dan dinyatakan lulus pada
15 Januari 2016
Telah diperiksa dan disetujui oleh:

| Dosen Pembimbing I | Dosen Pembimbing II |
|---|---|
| Imam Cholissodin, S.Si., M.Kom | Edy Santoso, S.Si., M.Kom |
| NIK: 850719 16 1 1 0422 | NIP: 19740414 200312 1 004 |

Mengetahui
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Marji, M.T
NIP: 19670801 199203 1 001

# PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 1 Januari 2016

Kentia Dea Hapsari

NIM: 115060801111084

# KATA PENGANTAR

Puji syukur kehadirat Allah SWT karena atas segala rahmat dan karunia-Nya, penulis dapat menyelesaikan penyusunan skripsi yang berjudul "Optimasi *Radial Basis Function Neural Network* menggunakan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk Peramalan Curah Hujan". Skripsi ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer di Program Studi Informatika/Ilmu Komputer Fakultas Ilmu Komputer Universitas Brawijaya Malang.

Penulis menyadari bahwa skripsi ini tidak akan dapat diselesaikan dengan baik tanpa keterlibatan dari berbagai pihak. Untuk itu, penulis menyampaikan ucapan terimakasih yang sebesar-besarnya kepada :

1. Imam Cholissodin, S.Si., M.Kom., selaku dosen pembimbing pertama yang telah memberikan bimbingan dan masukan dalam penulisan skripsi ini.
2. Edy Santoso, S.Si., M.Kom., selaku dosen pembimbing kedua yang telah memberikan bimbingan dan masukan dalam penulisan skripsi ini.
3. Drs. Marji, M.T dan Issa Arwani, S.Kom., M.Sc., selaku Ketua dan Sekretaris Program Studi Informatika/Ilmu Komputer Universitas Brawijaya.
4. Segenap Bapak dan Ibu dosen atas kesediaan membagi ilmu kepada penulis selama menempuh pendidikan di Program Studi Informatika/Ilmu Komputer Universitas Brawijaya.
5. Kedua orang tua, adik, serta keluarga besar penulis atas segala do'a, nasihat, dan perhatiannya dalam mendampingi dan memberikan dukungan moral kepada penulis.
6. Teman-teman penulis terutama Ira, Irma, Saras, Disya, dan Febria yang telah memberikan dorongan dan bantuan dalam berbagai hal selama menempuh pendidikan di Universitas Brawijaya.
7. Teman-teman seperjuangan skripsi, Maulian Eka dan Risky Hetari yang selalu menemani saat konsultasi skripsi dan memberikan dorongan serta bantuan dalam penulisan skripsi ini.
8. Teman-teman dari DOT Indonesia, terutama Mas Arya, Mas Didik, dan Mas Bekti yang selalu memberikan semangat dan bantuan dalam penulisan skripsi ini.
9. Future Classic, Roche Musique, Soulection, Avicii, Perfect Havoc Disco, Majestic Casual, HW&W, KAASI, ODESZA, Galimatias, Snakehips, Bondax, Julio Bashmore dan masih banyak lainnya yang selalu menemani penulis mengerjakan tugas selama menempuh pendidikan di Program Studi Informatika/Ilmu Komputer Universitas Brawijaya.

10. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Tidak ada gading yang tak retak, begitupula juga dengan laporan yang penulis susun. Maka dari itu, penulis sangat mengharapkan saran dan kritik dari pembaca sehingga di kesempatan berikutnya penulis dapat menyusun laporan yang lebih baik lagi. Terakhir semoga skripsi ini dapat memberikan manfaat bagi pembaca terutama mahasiswa Fakultas Ilmu Komputer Universitas Brawijaya

Malang, 1 Januari 2016

Penulis

kentia.dea@gmail.com

# ABSTRAK

Peramalan curah hujan yang akurat sangat dibutuhkan oleh Kementerian Pertanian dalam membuat kalender tanam yang dapat digunakan petani menentukan awal musim tanam. BMKG sebagai badan resmi yang bertugas melakukan observasi dan mengolah data di bidang meteorologi, klimatologi, dan geofisika di Indonesia menggunakan beberapa metode dalam meramal curah hujan yaitu ANFIS, transformasi *wavelet*, dan ARIMA. Keakurasian hasil peramalan dari metode-metode ini diakui BMKG masih kurang baik sehingga menyebabkan keakurasian dari kalender tanam hanya mencapai 50% untuk seluruh wilayah Indonesia. Alasannya adalah karena dinamika pola atmosfer (seperti suhu muka laut dan siklon tropis) di Indonesia yang tidak menentu serta terdapat kelemahan pada masing-masing metode yang digunakan oleh BMKG. Metode lain yang populer digunakan untuk peramalan curah hujan adalah *Back Propagation* (BP) dan *Radial Basis Function* (RBF) yang termasuk dalam *Neural Network* (NN). RBF memiliki struktur yang lebih sederhana, serta kemampuan pendekatan non linier dan kecepatan konvergensi yang lebih baik dari BP. Sayangnya, teknik dalam mencari parameter yang sesuai dalam metode ini sangat kompleks.

Pada penelitian ini sistem peramalan dibuat menggunakan *Radial Basis Function Neural Network* (RBF-NN) dengan *Hybrid Particle Swarm Optimization dan Genetic Algorithm* (HPSOGA) untuk menentukan parameter optimal dari RBF (jumlah dari *node* tersembunyi, titik pusat vektor masukan, radius, dan bobot dari keluaran). Hasil penelitian menunjukkan bahwa model peramalan yang dibuat dapat menghasilkan nilai MAPE sebesar 37.325% atau akurasi sebesar 62.675%.

Kata kunci: peramalan curah hujan, *Radial Basis Function Neural Network*, Algoritma Genetika, *Particle Swarm Optimization*

# ABSTRACT

Accurate rainfall forecast is needed by the Ministry of Agriculture for creating planting calendar that can be used by farmers to determine the beginning of cropping season. BMKG as the official agency in charge of observation and data processing in the field of meteorology, climatology, and geophysics in Indonesia uses several methods to predict rainfall such as ANFIS, wavelet transformation, and ARIMA. The accuracy of forecasting results with these methods according to BMKG is still not good, causing the accuracy of the planting calendar only reached 50% for the entire Indonesia region. The reasons of the low accuracy are because of the dynamics of atmospheric patterns (such as sea-surface temperatures and tropical cyclones) in Indonesia is erratic, and there are drawbacks to each method used by BMKG. Another popular methods used for rainfall forecasting are Back Propagation (BP) and Radial Basis Function (RBF) which is part of Neural Network (NN). RBF has simpler structure, better ability of non-linear approach and convergence speed than BP. Unfortunately, the technique of finding appropriate parameters in this method is very complex.

In this research, the rainfall forecast system is built using Radial Basis Function Neural Network (RBF-NN) with Hybrid Particle Swarm Optimization and Genetic Algorithm (HPSOGA) to determine the optimal parameters of RBF (number of hidden nodes, the centers of input vectors, radius, and weights of the output). The results show that the conducted forecasting model can reach MAPE value of 37.325% or accuracy of 62.675%.

Keyword: rainfall forecast, Radial Basis Function Neural Network, Genetic Algorithm, Particle Swarm Optimization

# DAFTAR ISI

PENGESAHAN ..... ii
PERNYATAAN ORISINALITAS ..... iii
KATA PENGANTAR ..... iv
ABSTRAK ..... vi
ABSTRACT ..... vii
DAFTAR ISI ..... viii
DAFTAR TABEL ..... xii
DAFTAR GAMBAR ..... xiv
DAFTAR LAMPIRAN ..... xvi
BAB 1 PENDAHULUAN ..... 1
1.1 Latar belakang ..... 1
1.2 Rumusan masalah ..... 2
1.3 Tujuan ..... 3
1.4 Manfaat ..... 3
1.5 Batasan masalah ..... 3
1.6 Sistematika pembahasan ..... 4
BAB 2 KAJIAN PUSTAKA DAN DASAR TEORI ..... 5
2.1 Kajian pustaka ..... 5
2.2 Dasar teori ..... 8
2.2.1 Curah hujan ..... 8
2.2.2 Kalender tanam ..... 8
2.2.3 *Radial basis function* (RBF) ..... 9
2.2.4 *Particle swarm optimization* (PSO) ..... 11
2.2.5 *Genetic algorithm* (GA) ..... 12
2.2.6 *Hybrid particle swarm optimization* dan *genetic algorithm* (HPSOGA) untuk optimasi RBF ..... 15
2.2.7 Normalisasi dan denormalisasi data ..... 18
2.2.8 Evaluasi hasil peramalan ..... 19
BAB 3 METODOLOGI DAN PERANCANGAN ..... 22
3.1 Metode penelitian ..... 22

3.1.1 Studi literatur ..... 22
3.1.2 Wawancara ..... 23
3.1.3 Analisis kebutuhan sistem ..... 23
3.1.4 Perancangan sistem ..... 23
3.1.5 Implementasi sistem ..... 24
3.1.6 Pengujian ..... 24
3.1.7 Kesimpulan dan saran ..... 24
3.2 Alir perancangan sistem ..... 24
3.2.1 Normalisasi data ..... 25
3.2.2 Tabulasi data *input* jaringan RBF ..... 26
3.2.3 Tabulasi data *output* jaringan RBF ..... 27
3.2.4 Optimasi RBF dengan HPSOGA ..... 28
3.2.5 Peramalan curah hujan dengan RBF ..... 43
3.2.6 Evaluasi hasil peramalan curah hujan dengan RBF-HPSOGA ..... 45
3.3 Perhitungan Manual Proses Peramalan Curah Hujan dengan RBF yang Dioptimasi Menggunakan HPSOGA ..... 48
3.3.1 Normalisasi data ..... 48
3.3.2 Tabulasi data *input* dan *output* ..... 49
3.3.3 Optimasi RBF dengan HPSOGA ..... 50
3.3.4 Peramalan curah hujan dengan RBF ..... 63
3.3.5 Evaluasi Hasil Peramalan Curah Hujan dengan RBF-HPSOGA ..... 65
3.4 Perancangan antarmuka ..... 68
3.4.1 Halaman *input* parameter ..... 68
3.4.2 Halaman normalisasi dan tabulasi ..... 69
3.4.3 Halaman optimasi RBF dengan HPSOGA ..... 70
3.4.4 Halaman peramalan dengan RBF ..... 71
3.4.5 Halaman evaluasi ..... 72
3.5 Perancangan uji coba dan evaluasi ..... 73
3.5.1 Pengujian parameter ..... 73
3.5.2 Pengujian data ..... 77
BAB 4 IMPLEMENTASI ..... 79
4.1 Lingkungan implementasi ..... 79

4.1.1 Lingkungan perangkat keras ..... 79
4.1.2 Lingkungan perangkat lunak ..... 79
4.2 Implementasi program ..... 80
4.2.1 Fungsi normalisasi data ..... 80
4.2.2 Fungsi tabulasi data *input* jaringan RBF ..... 80
4.2.3 Fungsi tabulasi data *output* jaringan RBF ..... 81
4.2.4 Fungsi optimasi RBF dengan HPSOGA ..... 82
4.2.5 Fungsi peramalan curah hujan dengan RBF ..... 98
4.2.6 Fungsi evaluasi hasil peramalan curah hujan dengan RBF-HPSOGA ..... 99
4.3 Implementasi antarmuka ..... 101
4.3.1 Tampilan halaman *input* parameter ..... 102
4.3.2 Tampilan halaman normalisasi dan tabulasi ..... 102
4.3.3 Tampilan halaman optimasi RBF dengan HPSOGA ..... 103
4.3.4 Tampilan halaman peramalan dengan RBF ..... 104
4.3.5 Tampilan halaman evaluasi ..... 104
BAB 5 PENGUJIAN DAN ANALISIS ..... 106
5.1 Pengujian ..... 106
5.2 Hasil dan analisis pengujian parameter ..... 106
5.2.1 Hasil dan analisis pengujian jumlah *node* masukan yang optimal ..... 106
5.2.2 Hasil dan analisis pengujian jumlah *node* tersembunyi yang optimal ..... 108
5.2.3 Hasil dan analisis pengujian jumlah populasi yang optimal ..... 109
5.2.4 Hasil dan analisis pengujian jumlah iterasi yang optimal ..... 111
5.2.5 Hasil dan analisis pengujian kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*) ..... 113
5.2.6 Hasil dan analisis pengujian nilai *delta* ($\delta$) tukar silang ..... 114
5.2.7 Hasil dan analisis pengujian kombinasi *learning rate* ($\gamma 1$ dan $\gamma 2$) ..... 116
5.3 Hasil dan analisis pengujian data ..... 118
5.3.1 Hasil dan analisis pengujian peramalan data uji menggunakan data latih dengan parameter optimal ..... 118

5.3.2 Hasil dan analisis pengujian jumlah data latih yang optimal .... 120
5.4 Analisis global keseluruhan pengujian ..... 121
BAB 6 KESIMPULAN DAN SARAN ..... 124
6.1 Kesimpulan ..... 124
6.2 Saran ..... 125
DAFTAR PUSTAKA ..... 126
LAMPIRAN A DATA HUJAN DASARIAN TAHUN 2009-2014 STASIUN KLIMATOLOGI KARANGPLOSO MALANG ..... 131
LAMPIRAN B VISUALISASI HASIL PERAMALAN CURAH HUJAN PENGUJIAN JUMLAH DATA LATIH OPTIMAL ..... 132
LAMPIRAN C VISUALISASI HASIL PERAMALAN CURAH HUJAN PADA TAHUN LAIN DENGAN SOLUSI TERBAIK ..... 137

# DAFTAR TABEL

Tabel 2.1 Kajian pustaka ..... 5
Tabel 2.2 Diagram skematik dari penulisan kromosom ..... 16
Tabel 3.1 Data curah hujan dasarian tahun 2009-2014 daerah Karangploso Kabupaten Malang ..... 48
Tabel 3.2 Data curah hujan dasarian tahun 2009-2014 daerah Karangploso Kabupaten Malang ternormalisasi ..... 48
Tabel 3.3 Data latih jaringan RBF pada perhitungan manual ..... 50
Tabel 3.4 Representasi kromosom pengkodean biner pada populasi awal ..... 51
Tabel 3.5 Representasi kromosom pengkodean nilai real pada populasi awal ..... 51
Tabel 3.6 Perhitungan fungsi *radial basis* dan nilai keluaran jaringan RBF untuk solusi $P_1$ ..... 54
Tabel 3.7 Perhitungan nilai *error* dan *fitness* untuk seluruh solusi pada populasi awal ..... 54
Tabel 3.8 Perangkingan kromosom pada populasi awal ..... 55
Tabel 3.9 Pengaturan kromosom untuk optimasi parameter jaringan RBF ..... 55
Tabel 3.10 Hasil tukar silang pertama pengkodean biner ..... 56
Tabel 3.11 Pembangkitan nilai acak (*c*) ..... 56
Tabel 3.12 Hasil tukar silang pertama pengkodean nilai real ..... 56
Tabel 3.13 Hasil tukar silang kedua pengkodean biner ..... 57
Tabel 3.14 Hasil tukar silang kedua pengkodean nilai real ..... 57
Tabel 3.15 Hasil mutasi pengkodean biner ..... 57
Tabel 3.16 Pembangkitan nilai acak (c) ..... 58
Tabel 3.17 Hasil mutasi pengkodean nilai real ..... 58
Tabel 3.18 Kecepatan awal partikel pengkodean biner ..... 59
Tabel 3.19 Kecepatan awal partikel pengkodean nilai real ..... 59
Tabel 3.20 Posisi awal partikel pengkodean biner ..... 59
Tabel 3.21 Posisi awal partikel pengkodean nilai real ..... 59
Tabel 3.22 Kecepatan baru partikel pengkodean biner ..... 60
Tabel 3.23 Kecepatan baru partikel pengkodean nilai real ..... 60
Tabel 3.24 Nilai acak untuk perbaruan posisi partikel pengkodean biner ..... 61
Tabel 3.25 Posisi baru partikel pengkodean biner ..... 62

Tabel 3.26 Posisi baru partikel pengkodean nilai real ..... 62
Tabel 3.27 Hasil proses seleksi ..... 63
Tabel 3.28 Individu terbaik pengkodean biner hasil proses seleksi ..... 63
Tabel 3.29 Individu terbaik pengkodean real hasil proses seleksi ..... 63
Tabel 3.30 Data uji jaringan RBF pada perhitungan manual ..... 64
Tabel 3.31 Perhitungan fungsi *radial basis* dan nilai keluaran jaringan RBF untuk individu terbaik ..... 65
Tabel 3.32 Data curah hujan uji aktual dan prediksi hasil proses denormalisasi . 66
Tabel 3.33 Rancangan uji coba jumlah *node* masukan ..... 74
Tabel 3.34 Rancangan uji coba jumlah *node* tersembunyi ..... 74
Tabel 3.35 Rancangan uji coba jumlah populasi ..... 75
Tabel 3.36 Rancangan uji coba jumlah iterasi ..... 75
Tabel 3.37 Rancangan uji coba kombinasi *pc* dan *pm* ..... 76
Tabel 3.38 Rancangan uji coba nilai *delta (δ)* tukar silang ..... 76
Tabel 3.39 Rancangan uji coba kombinasi $\gamma_1$ dan $\gamma_2$ ..... 77
Tabel 3.40 Rancangan uji coba peramalan data uji menggunakan data latih ..... 78
Tabel 3.41 Rancangan uji coba jumlah data latih ..... 78
Tabel 5.1 Hasil pengujian jumlah *node* masukan ..... 107
Tabel 5.2 Hasil pengujian jumlah *node* tersembunyi ..... 108
Tabel 5.3 Hasil pengujian jumlah populasi ..... 110
Tabel 5.4 Hasil pengujian jumlah iterasi ..... 112
Tabel 5.5 Hasil pengujian kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*) ..... 113
Tabel 5.6 Hasil pengujian nilai *delta* (*δ*) tukar silang ..... 115
Tabel 5.7 Hasil pengujian kombinasi *learning rate* (*γ1* dan *γ2*) ..... 116
Tabel 5.8 Hasil pengujian peramalan data uji menggunakan data latih ..... 118
Tabel 5.9 Hasil pengujian jumlah data latih ..... 120

# DAFTAR GAMBAR


Gambar 2.1 Arsitektur jaringan syaraf tiruan RBF ..... 10
Gambar 2.2 Contoh metode tukar silang *one-point* ..... 14
Gambar 2.3 Contoh metode mutasi *bit-wise* ..... 14
Gambar 3.1 Metode penelitian ..... 22
Gambar 3.2 Diagram alir sistem ..... 25
Gambar 3.3 Diagram alir normalisasi data ..... 26
Gambar 3.4 Diagram alir tabulasi data *input* jaringan RBF ..... 27
Gambar 3.5 Diagram alir tabulasi data *output* jaringan RBF ..... 27
Gambar 3.6 Diagram alir optimasi RBF dengan HPSOGA ..... 29
Gambar 3.7 Diagram alir representasi kromosom ..... 30
Gambar 3.8 Diagram alir perhitungan *fitness* ..... 32
Gambar 3.9 Diagram alir perangkingan ..... 33
Gambar 3.10 Diagram alir tukar silang ..... 34
Gambar 3.11 Diagram alir mutasi ..... 37
Gambar 3.12 Diagram alir perbarui partikel *personal best* (*Pbest*) dan g*lobal best* (*Pgest*) ..... 39
Gambar 3.13 Diagram alir perbarui kecepatan dan posisi partikel ..... 40
Gambar 3.14 Diagram alir seleksi ..... 43
Gambar 3.15 Diagram alir peramalan curah hujan dengan RBF ..... 44
Gambar 3.16 Diagram alir denormalisasi data ..... 46
Gambar 3.17 Diagram alir evaluasi ..... 47
Gambar 3.18 Arsitektur jaringan syaraf tiruan RBF pada perhitungan manual ..... 49
Gambar 3.19 Arsitektur jaringan syaraf tiruan untuk solusi $P_1$ ..... 52
Gambar 3.20 Arsitektur jaringan syaraf tiruan RBF untuk individu terbaik ..... 64
Gambar 3.21 Data curah hujan uji aktual dan prediksi hasil proses denormalisasi ..... 66
Gambar 3.22 Rancangan antarmuka halaman i*nput* parameter ..... 69
Gambar 3.23 Rancangan antarmuka halaman normalisasi dan tabulasi ..... 70
Gambar 3.24 Rancangan antarmuka halaman optimasi RBF dengan HPSOGA ..... 71
Gambar 3.25 Rancangan antarmuka halaman peramalan dengan RBF ..... 72

Gambar 3.26 Rancangan antarmuka halaman evaluasi ..... 73
Gambar 4.1 Antarmuka halaman *input* parameter ..... 102
Gambar 4.2 Antarmuka halaman normalisasi dan tabulasi ..... 103
Gambar 4.3 Antarmuka halaman optimasi RBF dengan HPSOGA ..... 103
Gambar 4.4 Antarmuka halaman peramalan dengan RBF ..... 104
Gambar 4.5 Antarmuka halaman evaluasi ..... 105
Gambar 5.1 Grafik pengujian jumlah *node* masukan ..... 107
Gambar 5.2 Grafik pengujian jumlah *node* tersembunyi ..... 109
Gambar 5.3 Grafik pengujian jumlah populasi ..... 111
Gambar 5.4 Grafik pengujian jumlah iterasi ..... 112
Gambar 5.5 Grafik pengujian kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*) ..... 114
Gambar 5.6 Grafik pengujian nilai *delta* ($\delta$) tukar silang ..... 116
Gambar 5.7 Grafik pengujian kombinasi *learning rate* ($\gamma 1$ dan $\gamma 2$) ..... 117
Gambar 5.8 Hasil peramalan curah hujan bulan Januari - Desember tahun 2011 daerah Karangploso Kabupaten Malang ..... 119
Gambar 5.9 Hasil peramalan curah hujan bulan Januari - Desember tahun 2012 daerah Karangploso Kabupaten Malang ..... 119
Gambar 5.10 Grafik pengujian jumlah data latih ..... 121
Gambar 5.11 Arsitektur jaringan RBF solusi terbaik untuk peramalan curah hujan daerah Karangploso Malang tahun 2013 ..... 122
Gambar 5.12 Hasil peramalan curah hujan daerah Karangploso Malang tahun 2013 ..... 123

# DAFTAR LAMPIRAN

LAMPIRAN A DATA HUJAN DASARIAN TAHUN 2009-2014 STASIUN KLIMATOLOGI KARANGPLOSO MALANG ..... 131

LAMPIRAN B VISUALISASI HASIL PERAMALAN CURAH HUJAN PENGUJIAN JUMLAH DATA LATIH OPTIMAL ..... 132

LAMPIRAN C VISUALISASI HASIL PERAMALAN CURAH HUJAN PADA TAHUN LAIN DENGAN SOLUSI TERBAIK ..... 137

# BAB 1 PENDAHULUAN

## 1.1 Latar belakang

Kabupaten Malang adalah salah satu daerah di Propinsi Jawa Timur yang memiliki tingkat produksi di sektor pertanian dan perkebunan yang cukup tinggi (BPS Jatim, 2014). Namun sayangnya sektor pertanian dan perkebunan rentan mengalami gagal panen apabila memasuki musim penghujan dengan curah hujan yang tinggi (di atas 300 mm per bulan) serta apabila memasuki musim kemarau dengan curah hujan yang rendah (di bawah 100 mm per bulan) (BMKG Staklim Karangploso Malang, 2015). Selama ini upaya yang dilakukan para petani untuk mengatasi hal tersebut hanyalah berupa upaya reaktif seperti melakukan panen secara dini. Upaya ini cukup efektif dalam mengurangi besarnya kerugian, tetapi sebaiknya yang dilakukan adalah upaya proaktif agar gagal panen tidak lagi terjadi (Roqib, 2015).

Salah satu contoh upaya proaktif yang dapat dilakukan adalah membuat kalender tanam yang dapat digunakan petani dalam menentukan awal musim tanam terbaik, seperti yang telah dilakukan oleh Badan Penelitian dan Pengembangan Pertanian (Balitbangtan) Kementerian Pertanian setiap dua kali dalam setahun. Dalam hal ini, Balitbangtan menggunakan data peramalan curah hujan dasarian dari Badan Meteorologi Klimatologi dan Geofisika (BMKG) untuk menentukan awal masuk dan berakhirnya musim penghujan atau kemarau (Ekasari, 2015). Sayangnya BMKG dalam operasinya sering memberikan ramalan yang kurang akurat (Utomo, 2014) sehingga menyebabkan keakurasian dari kalender tanam Balitbangtan ini baru mencapai 50% untuk seluruh wilayah Indonesia (Dianingtyas, 2014).

Beberapa metode peramalan curah hujan yang sering digunakan oleh BMKG adalah *Adaptive Neuro-Fuzzy Inference Systems* (ANFIS) (Ingragustari, 2005a), transformasi *wavelet* (Ingragustari, 2005b), dan *Autoregressive Integrated Moving Average* (ARIMA) (Nuryadi, 2005). Ketiga metode ini digunakan pula pada Stasiun Klimatologi Kelas II Karangploso Malang. Untuk dapat menetapkan angka perkiraan curah hujan di sepuluh hari atau bulan berikutnya, hasil peramalan dari ketiga metode tersebut digabungkan kemudian pihak pakar dari Stasiun Klimatologi akan melihat dari aspek dinamika pola atmosfer pada saat itu. Keakurasian dari metode peramalan ini diakui BMKG masih kurang baik yaitu sekitar 70%. Alasan utamanya adalah karena dinamika pola atmosfer di Indonesia yang tidak menentu, contohnya seperti suhu muka laut dan siklon tropis.

Selain itu terdapat beberapa kekurangan dari metode-metode yang digunakan BMKG dalam melakukan peramalan. Metode ARIMA tidak bisa menangani masalah seperti multi variabel dan heteroskedastisitas (ketidaksamaan varian dari *error* di sekitar garis regresi) (S. Yu, Wang, dan Wei, 2015). Metode ANFIS

membutuhkan data pelatihan yang besar dan penentuan bentuk fungsi keanggotaan serta lokasi untuk tiap variabel *fuzzy* yang tepat untuk hasil yang akurat (Giovanis, 2012). Sedangkan metode transformasi *wavelet* kurang dalam hal pemilihan arah yang baik (hanya tersedia arah horizontal, vertikal, dan diagonal) dan *shift variance* (kondisi dimana sinyal yang bergeser mengakibatkan hasil koefisien *wavelet* yang berbeda secara signifikan apabila dibandingkan dengan aslinya) (Vijay dan Mathurakani, 2014).

Selain metode yang sering digunakan BMKG, metode lain yang populer digunakan untuk peramalan curah hujan adalah *Back Propagation* (BP) dan *Radial Basis Function* (RBF) yang termasuk dalam *Neural Network* (NN). RBF memiliki struktur yang lebih sederhana, serta kemampuan pendekatan non linier dan kecepatan konvergensi yang lebih baik dari BP. Sayangnya, teknik dalam mencari parameter yang sesuai dalam metode ini sangat kompleks (Wu, Long, dan Liu, 2015).

Untuk mengatasi kekurangan RBF, diusulkan metode *Hybrid Particle Swarm Optimization dan Genetic Algorithm* (HPSOGA). HPSOGA sendiri telah diaplikasikan untuk optimasi pada berbagai bidang, seperti prediksi permintaan energi (S. Yu, Wei, dan Wang, 2012) dan penyesuaian kurva manufaktur (Gálvez dan Iglesias, 2013). Penelitian yang menggabungkan metode RBF dengan HPSOGA telah dilakukan sebelumnya untuk memprediksi curah hujan di kota Liuzhou Cina, dimana HPSOGA digunakan untuk mendapatkan parameter RBF yang optimal. Penggabungan metode ini memberikan performa yang lebih baik dari metode RBF-NN dan RBF-GA dari segi nilai *Average Absolute Relative Error* (AARE), *Root Mean Squared Error* (RMSE), dan *Correlation Coefficient* (CC) baik pada proses pelatihan maupun pengujian (Wu, Long, dan Liu, 2015). Maka dari itu, pada penelitian ini diusulkan metode *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* (HPSOGA) untuk optimasi RBF dalam meramalkan curah hujan di Kabupaten Malang dengan harapan dapat memberikan hasil peramalan curah hujan yang lebih akurat.

## 1.2 Rumusan masalah

Berdasarkan latar belakang masalah di atas, dapat dibuat rumusan masalah sebagai berikut:

1. Bagaimana mengimplementasikan metode *Radial Basis Function Neural Network* menggunakan optimasi *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan?
2. Bagaimana nilai evaluasi dari hasil implementasi metode *Radial Basis Function Neural Network* menggunakan optimasi *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan?

## 1.3 Tujuan

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan metode *Radial Basis Function Neural Network* menggunakan optimasi *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan.
2. Mengetahui nilai evaluasi dari hasil implementasi metode *Radial Basis Function Neural Network* menggunakan optimasi *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan.

## 1.4 Manfaat

Manfaat dari penelitian ini adalah:

1. Bagi pihak BMKG untuk dapat memberikan peramalan curah hujan dasarian yang lebih akurat dari sebelumnya.
2. Bagi Badan Penelitian dan Pengembangan Pertanian (Balitbangtan) Kementerian Pertanian untuk dapat membuat kalender pola tanam dengan menggunakan data ramalan curah hujan yang lebih akurat.
3. Bagi Dinas Pertanian dan Perkebunan Kabupaten Malang untuk dapat menggunakan kalender pola tanam sebagai data dasar untuk memberikan penyuluhan dan penjelasan yang tepat pada sektor pertanian dan perkebunan Kabupaten Malang.
4. Bagi penduduk Kabupaten Malang khususnya yang bekerja pada sektor pertanian dan perkebunan untuk dapat mengoptimalkan hasil produksi dengan menyesuaikan penanaman sesuai kalender pola tanam yang telah dibuat.

## 1.5 Batasan masalah

Penelitian ini dibatasi oleh hal-hal sebagai berikut:

1. Data curah hujan yang digunakan pada penelitian ini mencakup wilayah tertentu saja, yaitu wilayah Karangploso Kabupaten Malang.
2. Periode waktu dari data curah hujan yang digunakan pada penelitian ini adalah 6 tahun dari tahun 2009-2014.
3. Peramalan curah hujan yang dilakukan pada penelitian ini adalah dalam rentang waktu dasarian (10 hari). Sehingga dalam satu bulan dibagi menjadi tiga dasarian, yaitu:
    - Dasarian 1: tanggal 1 sampai dengan 10
    - Dasarian 2: tanggal 11 sampai dengan 20
    - Dasarian 3: tanggal 21 sampai dengan akhir bulan

## 1.6 Sistematika pembahasan

Sistematika pembahasan penelitian ditunjukkan untuk memberikan gambaran dan uraian dari penyusun tugas akhir yang meliputi beberapa bab sebagai berikut:

**BAB 1 : Pendahuluan**

Pada bab ini akan dijelaskan mengenai latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika pembahasan.

**BAB 2 : Kajian Pustaka dan Dasar Teori**

Pada bab ini diuraikan tentang teori atau konsep dari literatur ilmiah yang mendasari pembuatan sistem peramalan curah hujan menggunakan *Radial Basis Function Neural Network* yang dioptimasi dengan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm*.

**BAB 3 : Metodologi dan Perancangan**

Pada bab ini diuraikan tentang tahapan dalam pembuatan serta perancangan perangkat lunak sistem peramalan curah hujan menggunakan *Radial Basis Function Neural Network* yang dioptimasi dengan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm*.

**BAB 4 : Implementasi**

Pada bab ini dibahas implementasi dari sistem peramalan curah hujan menggunakan *Radial Basis Function Neural Network* yang dioptimasi dengan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* sesuai dengan perancangan sistem yang telah dibuat sebelumnya.

**BAB 5 : Pengujian dan Analisis**

Pada bab ini diuraikan tentang hasil pengujian dan analisis terhadap sistem peramalan curah hujan yang telah dibangun.

**BAB 6 : Kesimpulan dan Saran**

Pada bab ini dimuat kesimpulan yang diperoleh dari pembuatan dan pengujian perangkat serta saran untuk pengembangan penelitian terkait lebih lanjut.

# BAB 2 KAJIAN PUSTAKA DAN DASAR TEORI

Bab ini meliputi dua bagian pembahasan yaitu kajian pustaka dan dasar teori. Pada bagian kajian pustaka akan dibahas perbedaan antara penelitian yang telah ada dengan penelitian yang diusulkan oleh penulis. Sedangkan pada bagian dasar teori akan dibahas teori-teori yang digunakan pada penelitian yang diusulkan.

## 2.1 Kajian pustaka

Kajian pustaka disini menjelaskan tentang perbandingan antara penelitian yang diusulkan oleh penulis dengan penelitian sebelumnya yang relevan. Perbandingan tersebut dapat dilihat pada Tabel 2.1.

**Tabel 2.1 Kajian pustaka**

| No. | Judul | Objek dan *Input* Penelitian | Metode dan Hasil |
|---|---|---|---|
| 1. | *Rainfall Prediction in Kemayoran Jakarta Using Hybrid Genetic Algorithm* (GA) *and Partially Connected Feedforward Neural Network* (PCFNN) (Nurcahyo, Nhita, dan Adiwijaya, 2014) | Objek: curah hujan<br>*Input* penelitian:<br>• Suhu udara<br>• Kecepatan angin<br>• Intensitas radiasi matahari<br>• Tekanan udara<br>• Kelembapan udara<br>• Curah hujan hari sebelumnya | Metode: *Genetic Algorithm* (GA) – *Partially Connected Feedforward Neural Network* (PCFNN).<br>Nilai evaluasi:<br>*Mean Absolute Percentage Error* (MAPE). |
| 2. | *A New Hybrid Evolutionary Based RBF Networks Method for Forecasting Time Series: A Case Study of Forecasting Emergency Supply Demand Time Series* (Mohammadi, Ghomi, dan Zeinali, 2014) | Objek: permintaan persediaan bantuan makanan untuk darurat bencana<br>*Input* penelitian:<br>• Volume air minum<br>• Jumlah makanan kaleng<br>• Jumlah roti | Metode: *Adaptive Particle Swarm Optimization* (APSO) – *Genetic Algorithm* (GA) – *Radial Basis Function* (RBF).<br>Nilai evaluasi:<br>*Mean Absolute Percentage Error* (MAPE). |
| 3. | *A Hybrid Self-Adaptive Particle Swarm Optimization-Genetic Radial Basis Function Model for Annual Electricity Demand Prediction* (S. Yu, Wang, dan Wei, 2015) | Objek: permintaan beban listrik tahunan<br>*Input* penelitian:<br>• Produk Domestik Bruto (PDB)<br>• Populasi penduduk<br>• Andil produksi industri tersier pada PDB (dalam persen)<br>• Temperatur rata-rata tahunan<br>• Intensitas energi industri | Metode: *Particle Swarm Optimization* (PSO) – *Genetic Algorithm* (GA) – *Radial Basis Function* (RBF).<br>Nilai evaluasi:<br>*Mean Absolute Percentage Error* (MAPE). |

**Tabel 2.1 Kajian pustaka (lanjutan)**

| No. | Judul | Objek dan *Input* Penelitian | Metode dan Hasil |
|---|---|---|---|
| 4. | *Evolving RBF Neural Networks for Rainfall Prediction Using Hybrid Particle Swarm Optimization and Genetic Algorithm* (Wu, Long, dan Liu, 2015) | Objek: curah hujan<br>*Input* penelitian:<br>Curah hujan bulanan | Metode: *Particle Swarm Optimization* (PSO) – *Genetic Algorithm* (GA) – *Radial Basis Function* (RBF).<br>Nilai evaluasi:<br>*Average Absolute Relative Error* (AARE), *Root Mean Square Error* (RMSE), dan *Correlation Coefficient* (CC). |
| 5. | Optimasi *Radial Basis Function Neural Network* menggunakan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk Peramalan Curah Hujan | Objek: curah hujan<br>*Input* penelitian:<br>Curah hujan dasarian | Metode: *Particle Swarm Optimization* (PSO) – *Genetic Algorithm* (GA) – *Radial Basis Function* (RBF).<br>Nilai evaluasi:<br>*Mean Absolute Error* (MAE), *Root Mean Square Error* (RMSE), dan *Correlation Coefficient* (CC). |

Sumber: (Nurcahyo, Nhita, dan Adiwijaya, 2014), (Mohammadi, Ghomi, dan Zeinali, 2014), (S. Yu, Wang, dan Wei, 2015), dan (Wu, Long, dan Liu, 2015)

Pada penelitian yang berjudul "*Rainfall Prediction in Kemayoran Jakarta Using Hybrid Genetic Algorithm* (GA) *and Partially Connected Feedforward Neural Network* (PCFNN)", objek yang diprediksi adalah curah hujan di Kemayoran Jakarta. Data masukan yang digunakan adalah suhu udara, kecepatan angin, intensitas radiasi matahari, tekanan udara, kelembapan udara, dan curah hujan pada hari sebelumnya. Penelitian ini menerapkan metode GA untuk mengoptimalkan koneksi dan bobot pada metode PCFNN. Dalam evaluasinya penelitian ini menggunakan *Mean Absolute Percentage Error* (MAPE) dan menghasilkan nilai 35.53% untuk *missing value* data curah hujan yang diganti dengan 0 dan 18.84% untuk *missing value* data curah hujan yang diganti dengan nilai rata-rata. Hasil ini lebih baik dari penelitian sebelumnya yang menerapkan metode *Evolving Neural Network* (ENN) dengan nilai MAPE sebesar 38.82%.

Pada penelitian yang berjudul "*A New Hybrid Evolutionary Based RBF Networks Method for Forecasting Time Series: A Case Study of Forecasting Emergency Supply Demand Time Series*", objek yang diprediksi adalah permintaan persediaan bantuan makanan untuk darurat bencana gempa bumi di Azerbaijan Timur Iran. Data masukan yang digunakan adalah permintaan volume air minum, jumlah makanan kaleng, dan jumlah roti. Penelitian ini menerapkan metode *Adaptive Particle Swarm Optimization* (APSO) dan GA untuk mengoptimalkan variabel masukan, jumlah neuron dalam lapisan tersembunyi, dan parameter pada metode

*Radial Basis Function* (RBF) yang meliputi titik pusat, lebar, dan bobot. Dalam evaluasinya penelitian ini menggunakan MAPE dan menghasilkan nilai 3.27% untuk model roti, 3.97% untuk model air minum, dan 3.13% untuk model makanan kaleng. Hasil ini lebih baik apabila dibandingkan dengan penelitian serupa yang menerapkan metode RBF dan *Particle Swarm Optimization* (PSO) saja.

Pada penelitian yang berjudul "*A Hybrid Self-Adaptive Particle Swarm Optimization-Genetic-Radial Basis Function Model for Annual Electricity Demand Prediction*", objek yang diprediksi adalah permintaan beban listrik tahunan di Wuhan Cina. Data masukan yang digunakan adalah tingkat produk Domestik Bruto (PDB), populasi penduduk, andil produksi industri tersier pada PDB (dalam persen), temperatur rata-rata tahunan, dan intensitas energi industry di Kota Wuhan. Penelitian ini menerapkan metode PSO dan GA untuk mengoptimalkan struktur jaringan dan parameter pada metode *Radial Basis Function* (RBF) yang meliputi titik pusat, lebar, dan bobot. Dalam evaluasinya penelitian ini menggunakan MAPE dan menghasilkan nilai 2.89% untuk dataset pelatihan dan 1.31% untuk dataset pengujian. Hasil ini lebih baik apabila dibandingkan dengan penelitian serupa dalam prediksi kebutuhan energi.

Pada penelitian yang berjudul "*Evolving RBF Neural Networks for Rainfall Prediction Using Hybrid Particle Swarm Optimization and Genetic Algorithm*", objek yang diprediksi adalah curah hujan bulanan di Liuzhou Cina. Data masukan yang digunakan adalah curah hujan bulanan di Kota Liuzhou. Penelitian ini menerapkan metode PSO dan GA untuk mengoptimalkan jumlah *node* tersembunyi dan parameter pada metode *Radial Basis Function* (RBF) yang meliputi titik pusat, radius, dan bobot. Dalam evaluasinya penelitian ini menggunakan *Average Absolute Relative Error* (AARE) dengan nilai 0.6, *Root Mean Square Error* (RMSE) dengan nilai 67.73, dan *Correlation Coefficient* (CC) dengan nilai 0.93. Hasil ini lebih baik apabila dibandingkan dengan penelitian serupa yang menerapkan metode RBF dan metode RBF - GA.

Sedangkan pada penelitian yang diusulkan penulis berjudul "Optimasi *Radial Basis Function Neural Network* menggunakan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk Peramalan Curah Hujan", objek yang diprediksi adalah curah hujan dasarian di Kabupaten Malang khususnya daerah Karangploso. Data masukan yang digunakan adalah curah hujan dasarian di Karangploso. Penelitian ini juga menerapkan metode PSO dan GA untuk mengoptimalkan jumlah *node* tersembunyi dan parameter pada metode *Radial Basis Function* (RBF) yang meliputi titik pusat, radius, dan bobot. Dalam evaluasinya penelitian ini akan menggunakan nilai MAE, RMSE, dan CC. Evaluasi menggunakan nilai AARE tidak digunakan karena terdapat data curah hujan dasarian aktual yang bernilai 0 (tidak terjadi hujan sama sekali pada dasarian tersebut) dan akan menghasilkan nilai evaluasi AARE yang tidak terbatas.

## 2.2 Dasar teori

Pada bab ini dijelaskan tentang dasar teori dari objek dan metode-metode yang digunakan pada penelitian ini, yang meliputi: curah hujan, kalender tanam, metode *Radial Basis Function* (RBF), metode *Particle Swarm Optimization* (PSO), metode *Genetic Algorithm* (GA), metode *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* (HPSOGA) untuk optimasi RBF, normalisasi data, tabulasi data, dan nilai evaluasi.

### 2.2.1 Curah hujan

Curah hujan adalah ketinggian air hujan yang terkumpul pada tempat yang datar, tidak mengalir, tidak menguap, dan tidak meresap. Satuan dari curah hujan adalah milimeter (mm). Curah hujan satu milimeter berarti dalam satu luasan satu meter persegi pada tempat yang datar tertampung air setinggi satu milimeter atau sebanyak satu liter (Badan Meteorologi Klimatologi dan Geofisika, 2015). Curah hujan dapat diukur dalam periode jangka waktu yang bermacam-macam. Curah hujan jangka pendek (per jam dan per hari) diukur oleh Stasiun Meteorologi sedangkan jangka panjang (per 10 harian dan per bulan) diukur oleh Stasiun Klimatologi.

Jumlah curah hujan dalam suatu tempat dapat dijadikan penentu awal musim baik musim kemarau maupun penghujan. Yang dimaksud dengan awal musim kemarau adalah awal terjadinya periode satu dasarian (10 harian) yang memiliki jumlah curah hujan kurang dari 50 milimeter dan diikuti oleh beberapa dasarian berikutnya dengan jumlah curah hujan yang juga kurang dari 50 milimeter. Sedangkan awal musim hujan adalah awal terjadinya periode satu dasarian yang memiliki jumlah curah hujan sama atau lebih dari 50 milimeter dan diikuti oleh beberapa dasarian berikutnya dengan jumlah curah hujan yang juga sama atau lebih dari 50 milimeter (Badan Meteorologi Klimatologi dan Geofisika, 2015).

### 2.2.2 Kalender tanam

Kalender Tanam Terpadu adalah sistem informasi dari Badan Penelitian dan Pengembangan Pertanian Kementerian Pertanian yang dibuat untuk mendukung Program Peningkatan Produksi Beras Nasional (P2BN). Sistem informasi ini dapat digunakan sebagai panduan kalender tanam baik untuk petani, penyuluh, petugas dinas pertanian, ataupun kelompok tani pada skala nasional karena informasinya yang sangat mendetail sampai dengan level kecamatan di Indonesia. Kalender tanam terpadu ini dapat diakses pada halaman *website* http://katam.litbang.pertanian.go.id/.

Informasi yang dapat diperoleh dari kalender tanam terpadu ini meliputi informasi spasial dan tabular tentang prediksi musim, awal tanam, pola tanam, luas tanam potensial, wilayah rawan banjir dan kekeringan, potensi serangan organisme pengganggu tanaman, rekomendasi varietas dan kebutuhan padi dan

palawija, rekomendasi dosis dan kebutuhan pupuk serta rekomendasi alat dan sarana pertanian berdasarkan prediksi variabilitas dan perubahan iklim (Badan Penelitian dan Pengembangan Pertanian Kementerian Pertanian, 2015).

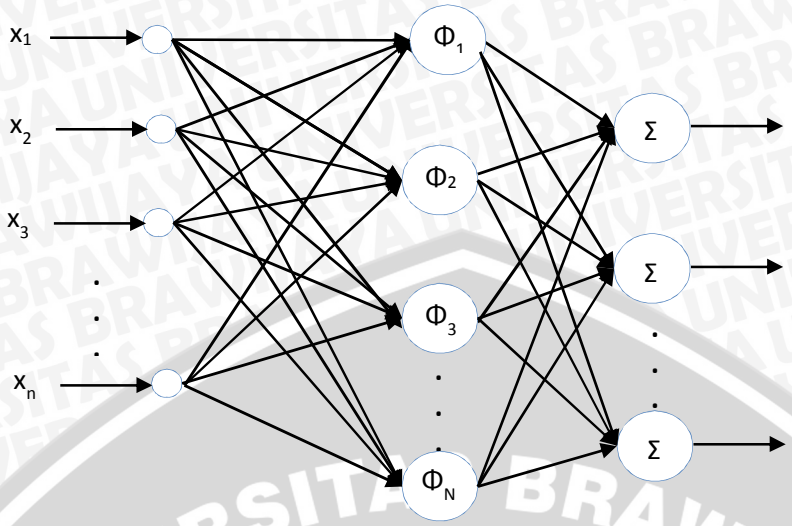
Dalam satu periode terdapat dua kalender tanam yang diterbitkan sesuai dengan jumlah musim yang terjadi Indonesia yaitu musim hujan (MH) dan musim kemarau (MK). Periode MH dimulai dari bulan Oktober sampai Maret tahun berikutnya, sedangkan MK dimulai dari bulan April sampai dengan bulan September (Badan Penelitian dan Pengembangan Pertanian Kementerian Pertanian, 2015). Untuk menentukan awal waktu tanam dalam dasarian pada setiap musim di setiap daerah, kalender tanam menggunakan data prakiraan iklim berupa curah hujan dalam dasarian dari Badan Meteorologi Klimatologi dan Geofisika pusat (Ekasari, 2015).

Keberadaan sistem informasi kalender tanam terpadu ini sudah dipastikan akan dapat membantu para petani di Indonesia dalam melakukan kegiatan budidaya tanaman pangan. Namun sayangnya keakurasian dari kalender tanam Balitbangtan ini baru mencapai 50% untuk seluruh wilayah Indonesia (Dianingtyas, 2014) diakibatkan BMKG dalam operasinya sering memberikan ramalan yang kurang akurat (Utomo, 2014). Oleh karena itu diperlukan suatu metode yang dapat memperbaiki keakurasian peramalan curah hujan dari BMKG yang akan dilakukan pada penelitian ini.

### 2.2.3 *Radial basis function* (RBF)

*Radial Basis Function* (RBF) merupakan algoritma jaringan syaraf tiruan yang diperkenalkan oleh Broomhead dan Lowe pada tahun 1988. Algoritma ini diilhami dari banyaknya jumlah unit respon lokal pada otak manusia (Winkler, Lawrence, dan Niranjan, 2005). Dalam satu dekade terakhir, RBF telah berhasil diaplikasikan untuk menyelesaikan banyak persoalan khususnya peramalan atau prediksi di berbagai bidang, contohnya peramalan tenaga angin (Sideratos dan Hatziargyriou, 2007), prediksi emisi medan magnet (Diao et al. 2015), dan estimasi karakteristik elektris dari modul surya (Bonanno et al., 2012). Alasan pemilihan algoritma ini adalah karena RBF memiliki kecepatan pelatihan yang lebih tinggi, struktur yang lebih sederhana, dan kemampuan pendekatan non linier yang lebih baik daripada algoritma jaringan syaraf tiruan yang lain (contohnya *Back Propagation*) (S. Yu, Wang, dan Wei, 2015).

Jaringan RBF terdiri dari tiga lapisan: lapisan masukan, lapisan tersembunyi, dan lapisan keluaran. Lapisan masukan berfungsi untuk mengumpulkan informasi yang masuk ke jaringan dan memformulasikan vektor masukan. Lapisan tersembunyi, yang terdiri dari satu atau lebih *node* tersembunyi, berfungsi untuk melakukan transformasi non-linier pada vektor masukan dengan fungsi *radial basis*. Lapisan keluaran berfungsi untuk mengirimkan respon jaringan syaraf tiruan ke lingkungan luar (Afantitis et al., 2005). Arsitektur dasar dari jaringan RBF dengan 3 lapisan ditampilkan pada Gambar 2.1.




**Gambar 2.1 Arsitektur jaringan syaraf tiruan RBF**

Fungsi yang biasa digunakan dalam lapisan tersembunyi jaringan RBF adalah fungsi *Gaussian* yang telah dinormalisasi. Persamaan dari fungsi *Gaussian* ini dijelaskan pada Persamaan (2.1) (Wu, Long, dan Liu, 2015).

$$\Phi_i(x, c_i) = \exp\left(-\frac{\|x - c_i\|_2}{{r_i}^2}\right) \quad (2.1)$$

Keterangan:

| | | |
|---|---|---|
| X | : | vektor masukan, $x \in R^{nx1}$ |
| N | : | jumlah *node* tersembunyi |
| $c_i$ | : | titik pusat vektor masukan ke-i, $c_i \in R^{nx1}$ dan i = 1, 2, …, N |
| $\|\|.\|\|_2$ | : | *Euclidean norm* |
| $r_i$ | : | radius dari *node* ke-i |

dimana Euclidean (L2) norm adalah persamaan yang biasa digunakan dalam kalkulus untuk perhitungan jarak yang dijelaskan pada Persamaan (2.2).

$$\|v\|_2 = \left(\sum_{i=1}^{n} {v_i}^2\right)^{1/2} \quad (2.2)$$

Kemudian untuk nilai keluaran dari jaringan $(y_t)$ dapat dihitung sebagai kombinasi linier dari fungsi r*adial basis* seperti pada Persamaan (2.3) (Wu, Long, dan Liu, 2015).

$$y_t = \sum_{i=1}^{N} w_{ti} \cdot \phi_i(x, c_i), \quad t = 1, 2, \dots, m \quad (2.3)$$

Keterangan:

$\phi_i(x, c_i)$ : fungsi *radial basis* untuk *node* tersembunyi ke-i
$w_{ti}$ : bobot yang menghubungkan antara *node* tersembunyi ke-i dengan *node* keluaran t

Dibalik banyaknya keuntungan dari pemakaian algoritma ini, RBF ternyata memiliki masalah utama yaitu bagaimana cara menetapkan nilai awal terbaik pada empat parameter berikut: jumlah dari *node* tersembunyi, titik pusat vektor masukan, radius, dan bobot dari keluaran. Jumlah dari *node* tersembunyi apabila tidak mencukupi akan menyebabkan hasil perkiraan menjadi sensitif terhadap *over-fitting* (banyak *error* yang acak) atau performa yang buruk. Jumlah parameter lain (titik pusat, radius, dan bobot) yang kurang akan menyebabkan *over-fitting*, sedangkan terlalu banyak parameter akan memberikan hasil yang lebih buruk (Wu, Long, dan Liu, 2015).

Oleh karena itu, untuk mendapatkan performa RBF yang baik maka parameter jaringan perlu dioptimalkan dengan mempertimbangkan koordinasinya. Penelitian untuk mengoptimalkan parameter jaringan RBF telah banyak dilakukan, diantaranya adalah dengan menggunakan algoritma *Particle Swarm Optimization* (Sermpinis et al., 2013), algoritma genetika (Burdsall dan Giraud-Carrier, 1998), atau dengan gabungan keduanya yang biasa disebut *Hybrid Particle Swarm Optimization* & *Genetic Algorithm* (Wu, Long, dan Liu, 2015).

### 2.2.4 *Particle swarm optimization* (PSO)

Algoritma *Particle Swarm Optimization* (PSO) merupakan algoritma metaheuristik yang digunakan untuk mencari solusi optimal atau mendekati optimal dari persoalan optimasi dengan mensimulasikan perilaku sosial binatang, contohnya kumpulan ikan yang berenang atau kawanan burung yang terbang ke arah yang sama secara terkoordinasi. Algoritma ini diusulkan oleh Kennedy dan Eberhart pada tahun 1995 dan menjadi salah satu algoritma optimasi yang sering digunakan karena konsepnya yang simpel, mudah untuk diimplementasikan, dan dapat dengan cepat mencari solusi yang layak (Askarzadeh, 2014). PSO telah banyak diterapkan untuk masalah optimasi pada berbagai bidang, contohnya seperti optimasi performa panas dari alat pemanas udara bertenaga surya (Siddhartha, Sharma, dan Varun, 2012), optimasi daya reaktif dan kontrol tegangan volt untuk penilaian keamanan tegangan volt (Yoshida et al., 2000), dan optimasi portofolio investasi keuangan (Cura, 2008).

Dalam algoritma PSO, setiap solusi yang layak dari suatu masalah (partikel) diinisialisasi secara acak dalam ruang pencarian. Setiap partikel kemudian akan bergerak dalam ruang pencarian untuk mendapatkan posisi yang lebih baik dari sebelumnya dengan cara memperbarui posisi dan kecepatannya. Posisi terbaik yang ditemukan oleh partikel itu sendiri serta posisi terbaik yang ditemukan oleh kelompok kemudian akan disimpan ke dalam memori (Askarzadeh, 2014).

Untuk memperbarui kecepatan dan posisi partikel pada masing-masing iterasi (t) dapat dihitung dengan Persamaan (2.4) dan (2.5) (Askarzadeh, 2014).

$$v_j(t+1) = \omega \cdot v_j(t) + c_1 \cdot r_1\left(pbest_j(t) - x_j(t)\right) + c_2 \cdot r_2\left(gbest(t) - x_j(t)\right) \quad (2.4)$$

$$x_j(t+1) = v_j(t+1) + x_j(t), \quad t = 1, 2, \dots, t_{max} \;\&\; j = 1, 2, \dots, N_p \quad (2.5)$$

Dimana $x_j$ merupakan posisi partikel, $N_p$ merupakan jumlah partikel, $v_j$ merupakan kecepatan partikel, r1 dan r2 merupakan angka acak antara 0 sampai 1 yang dibangkitkan secara independen untuk tiap partikel di tiap tahap perbaruan, $c_1$ dan $c_2$ merupakan *learning factor*, $t_{max}$ merupakan waktu iterasi maksimum, dan $\omega$ merupakan *inertia weight*.

Apabila posisi partikel berbentuk biner, maka Persamaan (2.5) dapat ditulis menjadi Persamaan (2.6) (Wu, Long, dan Liu, 2015).

$$x_j(t+1) = \begin{cases} 0, & rand \geq \frac{1}{1+\exp\left(-v_j(t+1)\right)} \\ 1, & rand < \frac{1}{1+\exp\left(-v_j(t+1)\right)} \end{cases} \quad (2.6)$$

Langkah-langkah dari algoritma PSO adalah sebagai berikut (Askarzadeh, 2014):

1. Bangkitkan populasi secara acak dalam ruang pencarian.
2. Inisialisasi kecepatan awal masing-masing partikel secara acak.
3. Hitung nilai fungsi objektif untuk setiap partikel.
4. Pilih posisi awal setiap partikel sebagai *pbest* dan partikel terbaik di antara populasi sebagai *gbest*.
5. Hitung kecepatan baru untuk tiap partikel berdasarkan Persamaan (2.4).
6. Pindah partikel ke posisi baru berdasarkan Persamaan (2.5) atau (2.6).
7. Jika partikel melebihi kisaran diperbolehkan diganti dengan yang posisi sebelumnya.
8. Hitung nilai fungsi objektif baru untuk setiap partikel.
9. Perbarui nilai *pbest* dan *gbest*.
10. Periksa kriteria berhenti dari algoritma. Jika memenuhi maka algoritma dihentikan dan *gbest* dipilih sebagai solusi optimal. Jika tidak, ulangi langkah 5 sampai 8.

### 2.2.5 *Genetic algorithm* (GA)

Algoritma genetika (*Genetic Algorithms, GAs*) merupakan algoritma yang diusulkan oleh John Holland dan koleganya pada tahun 1970-an di *University of Michigan*. GA telah berhasil diaplikasikan untuk mencari solusi yang diterima pada jangkauan domain masalah yang luas, seperti pada masalah optimasi efisiensi energi dan kenyamanan *thermal* pada desain bangunan (W. Yu et al., 2015), optimasi pemindahan guru sukarelawan di Taiwan (C.-H. Chen et al. 2015), dan optimasi desain sistem distribusi air (Bi, Dandy, dan Maier, 2014). Salah satu alasan

mengapa GA banyak digunakan adalah karena GA memiliki kemampuan untuk mencari solusi optimal dalam waktu yang pantas. GA juga mampu untuk memecahkan masalah yang kompleks, sulit, dan membutuhkan perhitungan komputasi yang tinggi (Sukstrienwong, 2013).

GA merupakan model komputasional yang terinspirasi oleh proses evolusi. Dalam proses evolusi, individu yang mampu menyesuaikan diri lebih baik dengan lingkungannya akan dapat bertahan dan mewariskan kromosom mereka untuk keturunannya, sementara itu yang tidak mampu akan menjadi punah. Kromosom individu ini akan direpresentasikan oleh *string* biner pada GA. Tiap bit pada setiap *string* disebut gen, dan nilai yang diwakili oleh gen disebut alel. Terdapat pula operator genetik dasar pada GA yang meliputi rekombinasi, mutasi, dan seleksi (Billings dan Zheng, 1995).

Langkah umum dalam mendapatkan solusi menggunakan Algoritma Genetika adalah sebagai berikut (Chande dan Sinha, 2008):

1. Inisialisasi

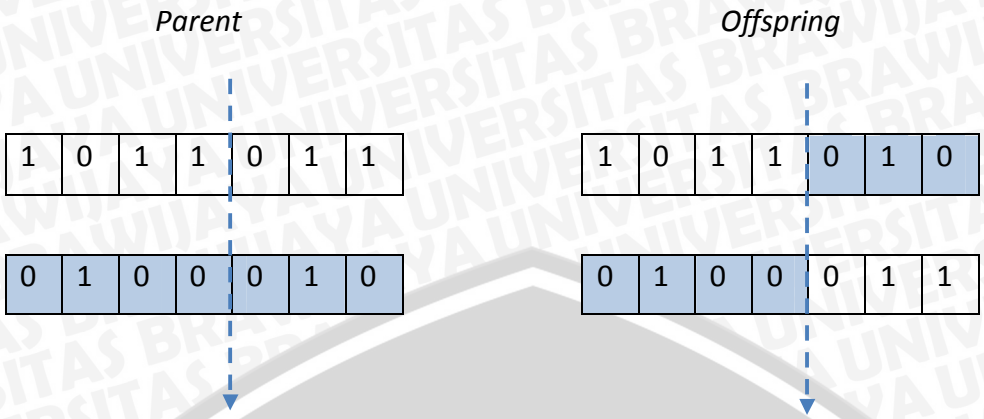
Populasi awal dibangkitkan dari pemilihan kemungkinan solusi, keputusan, atau hipotesis secara acak. Susunannya seperti kromosom dan bertindak sebagai representasi sebuah individu. Populasi awal idealnya memiliki individu yang beragam. Hal ini penting karena tiap individu melakukan pembelajaran dari masing-masing dari mereka. Rendahnya keberagaman dalam suatu populasi akan menyebabkan didapatkannya solusi di bawah optimal.

2. Reproduksi

Pada langkah ini, kandidat solusi dikombinasi untuk menghasilkan keturunan pada tiap iterasi algoritma yang dinamakan dengan generasi. Dari generasi induk dan anaknya, mereka yang paling baiklah yang bertahan menjadi kandidat solusi pada generasi selanjutnya. Keturunan biasanya diproduksi dengan operator genetik spesifik, seperti mutasi dan tukar silang.

a. Tukar silang

Tukar silang adalah secara acak memilih satu atau lebih pasangan individu sebagai induk dan secara acak pula menukar segmen gen induknya yang hasilnya akan menjadi keturunan pada generasi selanjutnya. Salah satu metode tukar silang dasar yang dapat dilakukan adalah metode *one-point* pada kromosom pengkodingan biner. Sepasang kromosom induk yang terpilih (*parent*) akan dipotong pada satu titik secara acak kemudian bagian gen yang terpotong tersebut akan saling ditukar untuk membentuk sepasang kromosom keturunan yang baru (*offspring*) (Lee, 2006). Pada Gambar 2.2 adalah contoh tukar silang metode *one-point* dengan titik yang dipilih adalah titik ke-4.




**Gambar 2.2 Contoh metode tukar silang *one-point***

b. Mutasi

Mutasi adalah cara paling dasar untuk mengubah kandidat solusi untuk generasi selanjutnya. Caranya adalah dengan secara acak mengganti salah satu atau lebih gen milik induk yang hasilnya akan menjadi keturunan. Salah satu metode mutasi dasar yang dapat dilakukan adalah metode *bit-wise* pada kromosom pengkodingan biner. Metode *bit-wise* mengubah gen dari 1 menjadi 0, dan sebaliknya (Lee, 2006). Pada Gambar 2.3 adalah contoh mutasi metode *bit-wise* dengan gen yang diubah adalah gen ke-5.


**Gambar 2.3 Contoh metode mutasi *bit-wise***

c. Evaluasi

Nilai *fitness* diberikan pada tiap solusi (kromosom) yang diukur dari seberapa dekat kromosom tersebut dapat memecahkan masalah yang bersangkutan. Fungsi *fitness* sendiri adalah ukuran dari tujuan yang ingin dicapai (nilai maksimum atau minimum). Fungsi *fitness* dapat dioptimalkan dengan menggunakan proses genetik dan pengevaluasian pada tiap solusi untuk menentukan apakah solusi tersebut akan berkontribusi secara baik pada generasi selanjutnya atau tidak.

d. Seleksi

Setelah melakukan evaluasi *fitness* pada tiap kromosom maka dilakukan seleksi untuk memilih individu / kromosom mana yang akan hidup atau bereproduksi pada generasi selanjutnya. Proses seleksi sangat terpengaruh dari fungsi evaluasi / *fitness*, dimana kromosom yang memiliki nilai *fitness* lebih tinggi akan memiliki peluang lebih tinggi untuk dipilih dan menjadi induk generasi selanjutnya.

Metode seleksi sendiri ada bermacam-macam, tiap metode memiliki kelebihan dan kekurangan masing-masing dan dapat dipilih berdasarkan masalah dan populasi yang dimiliki. Salah satu metode yang dapat digunakan adalah *elitism,* dimana individu / kromosom terbaik dari populasi selalu dipastikan diturunkan ke generasi selanjutnya. *Elitism* dikatakan dapat meningkatkan performa dari GA karena mencegah hilangnya solusi terbaik saat itu (Mitchell, 1998).

Apabila generasi baru mengandung solusi yang menghasilkan keturunan yang dekat atau sama dengan jawaban yang diinginkan, maka kemungkinan masalah telah terselesaikan lebih besar. Apabila generasi baru masih menghasilkan jawaban yang jauh dari keinginan, maka dilakukan iterasi dari langkah-langkah di atas sampai menemukan jawaban yang diinginkan atau sampai mencapai iterasi maksimum.

### 2.2.6 *Hybrid particle swarm optimization* dan *genetic algorithm* (HPSOGA) untuk optimasi RBF

*Hybrid Particle Swarm Optimization* & *Genetic Algorithm* (HPSOGA) adalah bentuk penggabungan dari algoritma *Particle Swarm Optimization* dengan *Genetics Algorithm*. Alasan utama dilakukannya hibridisasi disini adalah untuk mengambil keuntungan dari kedua algoritma sekaligus mengatasi keterbatasan utama dari masing-masing algoritma (Gálvez dan Iglesias, 2013).

GA memiliki beberapa keterbatasan yaitu dapat terjadinya konvergensi prematur, lemahnya kemampuan pencarian lokal, hilangnya informasi mengenai individu yang tidak terpilih karena tidak adanya memori, dan rendahnya kecepatan konvergensi (Duan et al., 2013). Keterbatasan GA ini dapat ditangani oleh PSO yang memiliki memori dan mudah diimplementasikan (Wu, Long, dan Liu, 2015), sehingga PSO mampu mendapatkan solusi yang mendekati optimal dengan kecepatan konvergensi yang tinggi (Duan et al., 2013). Sayangnya, pada PSO konvergensi prematur juga dapat terjadi karena sebagian besar partikel bergerak melalui perbandingan dengan posisi diri sendiri, posisi sekitar, dan posisi global dari semua partikel sebagai pola tunggal. Keterbatasan ini dapat ditangani oleh GA, dengan menggunakan mekanisme evolusi yang berbeda seperti rekombinasi dan mutasi dapat meningkatkan keragaman hasil PSO sehingga mengurangi terjadinya konvergensi prematur (S. Yu, Wei, dan Wang, 2012).

Pada penelitian ini HPSOGA digunakan untuk menetapkan nilai awal terbaik pada empat parameter berikut: jumlah dari *node* tersembunyi, titik pusat vektor masukan, radius, dan bobot dari keluaran. *Node* tersembunyi akan dikodekan dalam biner, dimana 1 menunjukkan adanya koneksi dari *node* masukan sampai ke *node* keluaran serta 0 menunjukkan tidak adanya koneksi. Panjang dari gen yang dikodekan sebagai *string* biner dihitung dengan Persamaan (2.7).

Panjang *string* biner (G) = n x N (2.7)

Dimana n adalah jumlah *node* masukan jaringan RBF dan N adalah jumlah *node* tersembunyi jaringan RBF. Sedangkan untuk titik pusat vektor masukan, radius, dan bobot dari keluaran jaringan akan dikodekan dalam real dengan panjang yang dihitung dengan Persamaan (2.8).

Panjang *string* real (H) = (N x n) + N + (N x m) (2.8)

Dimana m adalah jumlah *node* keluaran dari jaringan RBF. Sehingga panjang keseluruhan dari kromosom dalam penelitian ini dapat dihitung dengan Persamaan (2.9).

Panjang kromosom (L) = panjang gen biner (G) + panjang gen real (H) (2.9)

Tiap *string* baik biner maupun real berhubungan dengan sebuah kromosom, dengan urutan penulisan seperti pada Tabel 2.2. Pertama-tama dituliskan *string* biner terlebih dahulu yang berisi koneksi *node*, kemudian *string* real yang berisi titik pusat *node* tersembunyi, radius *node* tersembunyi, dan bobot *node* tersembunyi ke *node* keluaran.

**Tabel 2.2 Diagram skematik dari penulisan kromosom**

| Koneksi *Node* ($k_{11}$, $k_{12}$, …) | Titik Pusat ($cn_{11}$, $cn_{21}$, …) | Radius ($r_1$, $r_2$, …) | Bobot ($\omega_{11}$, $\omega_{12}$, …) |
|---|---|---|---|
| 1, 0, 1, …, 1 | 0.2, …, 0.3 | 0.5, …, 0.2 | 0.7, … , 0.4 |

Langkah-langkah dari algoritma HPSOGA untuk optimasi RBF adalah sebagai berikut (Wu, Long, dan Liu, 2015):

1. Inisialisasi parameter-parameter dari metode RBF, PSO, dan GA yang akan digunakan dalam sistem.
2. Inisialisasi populasi dengan ukuran 4N individu secara acak.
3. Lakukan peramalan metode RBF dengan menggunakan parameter dari tiap individu pada populasi awal dengan Persamaan (2.1) dan (2.3).
4. Evaluasi dan perangkingan: hitung nilai *error* peramalan dengan menggunakan Persamaan (2.10) serta nilai *fitness* dari tiap 4N individu dengan Persamaan (2.11).

$$E_i = \frac{1}{m} \sum_{k=1}^{m} E_i(x) = \frac{1}{m} \sum_{k=1}^{m} \left| e_i(x_j) - y_j \right| \quad (2.10)$$

$$Fitness = \frac{1}{(1.0 + E_i)} \quad (2.11)$$

Dimana $e_i(x_j)$ adalah hasil keluaran jaringan untuk dataset pelatihan x ke-j dan $y_j$ adalah hasil keluaran peramalan yang sebenarnya untuk dataset pelatihan x ke-j. Setelah itu lakukan pengurutan dari individu dengan nilai *fitness* yang tertinggi ke terendah.

5. Metode GA: pada tiap iterasi, setelah nilai *fitness* dari semua individu pada populasi yang sama telah dihitung, tandai 2N individu yang memiliki nilai *fitness* tertinggi sebagai elite dan gunakan operator GA pada 2N individual ini.

   a. Tukar Silang

      - Gunakan operasi tukar silang dasar untuk kromosom yang dikodekan sebagai *string* biner (Pada penelitian ini akan digunakan metode *one-point*).
      - Gunakan operasi tukar silang dengan metode *extended intermediate crossover* seperti pada Persamaan (2.12) untuk kromosom yang dikodekan sebagai *string* real (Muñoz 2002).

$$z_i = x_i + \alpha_i(y_i - x_i) \quad (2.12)$$

      Dimana $x = (x_1, \dots, x_n)$ dan $y = (y_1, \dots, y_n)$ adalah sepasang *string* kromosom *parent*, $z = (z_1, \dots, z_n)$ adalah *string* kromosom *offspring*, dan $\alpha_i \in [-\delta, 1 + \delta]$ dengan $\delta > 0$.

   b. Mutasi

      - Gunakan operasi mutasi dasar untuk kromosom yang dikodekan sebagai *string* biner (pada penelitian ini akan digunakan metode *bitwise*).
      - Gunakan operasi mutasi seperti pada Persamaan (2.13) untuk kromosom yang dikodekan sebagai *string* real.

$$X_i^{t+1} = X_i^t + c_i \quad (2.13)$$

      Dimana $X_i^t$ adalah individu sebelum mutasi dan $X_i^{t+1}$adalah individu setelah mutasi, $c_i$ adalah nilai acak dalam interval $[\upsilon_{min} - \delta_1 - X_i^t, \upsilon_{max} + \delta_1 + X_i^t]$ sehingga individu mutasi masih berada pada kisaran pencarian.

6. Metode PSO: Perbarui $P_{best}$ dan $P_{gbest}$ berdasarkan hasil evaluasi *fitness* dan gunakan operator PSO (pembaruan kecepatan dan posisi) untuk memperbarui 2N individu menggunakan 2N individu terbaik sebagai kecepatan partikel dan 2N individu terburuk sebagai posisi partikel.

   a. Persamaan untuk mencari bobot inersia, kecepatan serta posisi baru pada kromosom yang dikodekan sebagai nilai real dapat ditulis sebagai Persamaan (2.14) sampai (2.16).

$$\varpi = [0.5 + (\text{rand} / 2.0)] \quad (2.14)$$

$$V_{id}^{New} = \varpi \,.\, V_{id}^{Old} + \gamma_1 \,.\, rand \,.\, (P_{best} - x_{id}^{Old}) + \gamma_2 \,.\, rand \,.\, (P_{gest} - x_{id}^{Old}) \quad (2.15)$$

$$X_{id}^{New} = X_{id}^{Old} + V_{id}^{New} \quad (2.16)$$

Dimana $\gamma_1 = \gamma_2$ = *learning rate* dengan nilai 2, $P_{best}$ adalah posisi terbaik dari individu pada lingkungannya, $P_{gest}$ adalah posisi terbaik global, i adalah iterasi, dan d adalah dimensi. Kecepatan baru partikel $V_{id}^{New}$ dibatasi pada rentang [-$V_{max}$, $V_{max}$] dimana $V_{max}$ ditetapkan sebesar 60% dari rentang maksimum tiap variabel pada tiap dimensi (H.-L. Chen et al., 2011).

b. Persamaan untuk mencari posisi baru pada kromosom yang dikodekan sebagai *string* biner dapat ditulis sebagai Persamaan (2.17).

$$X_{id}^{New} = \begin{cases} 0, & rand \geq \frac{1}{1+\exp(-V_{id}^{New})} \\ 1, & rand < \frac{1}{1+\exp(-V_{id}^{New})} \end{cases} \quad (2.17)$$

7. Bangkitkan populasi baru
8. Apabila individu terbaru dengan *fitness* baru tidak dapat memenuhi kondisi penghentian, kembali ke langkah 3, sebaliknya proses akan menghasilkan solusi final.
9. Untuk menghindari fase pelatihan yang berlebihan, amati kurva akurasi validasi, dan hentikan pelatihan apabila iterasinya memiliki akurasi validasi terbaik saat fase pelatihan.
10. Latih kembali dan bangun RBF-NN pada dataset pengujian yang lebih besar berdasarkan jumlah *node* tersembunyi dan nilai parameter RBF-NN pada iterasi yang dihentikan pada langkah sebelumnya.

### 2.2.7 Normalisasi dan denormalisasi data

Normalisasi data umumnya dilakukan sebelum melakukan pemrosesan data. Atribut dari dataset dinormalisasi dengan penskalaan sehingga nilainya berada dalam rentang yang cukup kecil, seperti dari 0 sampai 1 atau lainnya (Jain dan Bhandare, 2011). Terdapat beberapa persamaan untuk normalisasi data, salah satunya adalah normalisasi min-max yang dapat ditulis sebagai Persamaan (2.18) (Han, Kamber, dan Pei, 2011).

$$V'_i = \frac{V_i - min_A}{max_A - min_A} (new\_max_A - new\_min_A) + new\_min_A \quad (2.18)$$

Keterangan:

| | | |
|---|---|---|
| $V'_i$ | : | data hasil normalisasi |
| $V_i$ | : | data asli belum ternormalisasi |
| $min_A$ | : | nilai minimum dari atribut numerik A |
| $max_A$ | : | nilai maksimum dari atribut numerik A |
| $new\_min_A$ | : | *range* minimum dari nilai hasil normalisasi |
| $new\_max_A$ | : | *range* maksimum dari nilai hasil normalisasi |

Sedangkan untuk mengembalikan data *output* setelah pemrosesan data menjadi nilai yang sebenarnya dapat dilakukan denormalisasi dengan Persamaan (2.19) yang didapatkan dari proses berikut ini.

$$V'_i = \frac{V_i - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

$$V'_i - new\_min_A = \frac{V_i - min_A}{max_A - min_A}(new\_max_A - new\_min_A)$$

$$\frac{V'_i - new\_min_A}{new\_max_A - new\_min_A} = \frac{V_i - min_A}{max_A - min_A}$$

$$(V_i - min_A)(new\_max_A - new\_min_A) = (V'_i - new\_min_A)(max_A - min_A)$$

$$V_i - min_A = \frac{(V'_i - new\_min_A)(max_A - min_A)}{new\_max_A - new\_min_A}$$

$$V_i = \frac{(V'_i - new\_min_A)(max_A - min_A)}{new\_max_A - new\_min_A} + min_A \quad (2.19)$$

Keterangan:

| | | |
|---|---|---|
| $V'_i$ | : | data *output* |
| $V_i$ | : | data hasil denormalisasi |
| $min_A$ | : | nilai minimum dari atribut numerik A |
| $max_A$ | : | nilai maksimum dari atribut numerik A |
| $new\_min_A$ | : | *range* minimum dari nilai *output* |
| $new\_max_A$ | : | *range* maksimum dari nilai *output* |

### 2.2.8 Evaluasi hasil peramalan

Dalam melakukan peramalan diperlukan proses evaluasi untuk mengetahui performa model peramalan yang diterapkan. Pada penelitian ini digunakan tiga macam statistik standar untuk proses evaluasi, yaitu *Mean Absolute Error* (MAE), *Mean Absolute Percentage Error* (MAPE), *Root Mean Square Error* (RMSE), dan *Correlation Coefficient* (CC).

*Mean Absolute Error* (MAE) merupakan rata-rata dari perbedaan antara nilai peramalan dengan nilai aktual dalam bentuk absolut. *Range* nilai dari MAE berkisar dari 0 sampai tak hingga, karena signifikansi dari *error* bergantung pada nilai aktualnya (Berretti, Thampi, dan Srivastava, 2015). Semakin kecil nilai dari MAE maka semakin baik model peramalan yang diterapkan. Persamaan untuk menghitung nilai MAE ditunjukkan dalam Persamaan (2.20) (Morecroft, 2007):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Xm_i - Xa_i| \quad (2.20)$$

Keterangan:

| | | |
|---|---|---|
| n | : | jumlah data |
| $Xa_i$ | : | nilai yang sebenarnya untuk data ke-i |

$Xm_i$ : nilai hasil peramalan untuk data ke-i

*Root Mean Square Error* (RMSE) merupakan salah satu standar pengukuran *error* yang paling banyak digunakan dalam evaluasi hasil peramalan. RMSE didapat dari nilai rata-rata pangkat perbandingan antara nilai yang diprediksi dengan nilai yang sebenarnya. Semakin kecil nilai dari RMSE maka semakin akurat hasil peramalan yang dilakukan (Brooks dan Tsolacos, 2010).

Persamaan untuk menghitung nilai RMSE ditunjukkan dalam Persamaan (2.21) (Phillips dan Lopez, 2007):

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(Y_t^s - Y_t^a)^2} \quad (2.21)$$

Keterangan:
N : jumlah data
$Y_t^s$ : nilai yang sebenarnya untuk data ke-t
$Y_t^a$ : nilai hasil peramalan untuk data ke-t

*Correlation Coefficient* (CC) merupakan nilai yang mengukur derajat hubungan antara dua variabel dan nilainya berkisar antara -1.00 sampai 1.00. Semakin kuat hubungan antar variabel maka nilai CC semakin mendekati angka -1.00 atau +1.00. Sedangkan semakin lemah hubungan antar variabel maka nilai CC semakin mendekati angka 0.00. Nilai korelasi yang positif menunjukkan hubungan langsung antar dua variabel, dimana peningkatan pada satu variabel akan menyebabkan peningkatan pada variabel lain dan penurunan pada satu variabel akan menyebabkan penurunan pula pada lainnya. Nilai korelasi yang negatif mengindikasikan hubungan *invers* atau negatif antar dua variabel, dimana peningkatan pada satu variabel akan menyebabkan penurunan pada variabel lain dan sebaliknya (Jackson, 2011). Persamaan untuk menghitung nilai CC ditunjukkan dalam Persamaan (2.22) (Rubin, 2012):

$$CC = \frac{N\sum XY - \sum X \sum Y}{\sqrt{[N\sum X^2 - (\sum X)^2][N\sum Y^2 - (\sum Y)^2]}} \quad (2.22)$$

Keterangan:
N : jumlah data
ΣX : total nilai dari variabel pertama
ΣY : total nilai dari variabel kedua
ΣXY : total nilai dari hasil perkalian antara setiap nilai X dengan setiap nilai Y
$\Sigma X^2$ : total nilai dari setiap nilai X yang dikuadratkan
$\Sigma Y^2$ : total nilai dari setiap nilai Y yang dikuadratkan

*Mean Absolute Percentage Error* (MAPE) merupakan rata-rata nilai absolut kesalahan peramalan dalam bentuk persentase (Berretti, Thampi, dan Srivastava,

2015). Semakin kecil nilai dari MAPE maka semakin akurat model peramalan yang diterapkan. Persamaan untuk menghitung nilai MAPE pada penelitian ini ditunjukkan dalam Persamaan (2.23) dan (2.24):

$$MAPE = \frac{1}{N} \sum_{i=1}^{N} Error_i \tag{2.23}$$

$$Error_i = \begin{cases} 100, & \left(\frac{|(X_i+c)-(F_i+c)|}{(X_i+c)} \times 100\right) > 100 \\ \frac{|(X_i+c)-(F_i+c)|}{(X_i+c)} \times 100, & \left(\frac{|(X_i+c)-(F_i+c)|}{(X_i+c)} \times 100\right) < 100 \end{cases} \tag{2.24}$$

Keterangan:

N : jumlah data
$X_i$ : nilai yang sebenarnya untuk data ke-i
$F_i$ : nilai hasil peramalan untuk data ke-i
c : nilai konstanta

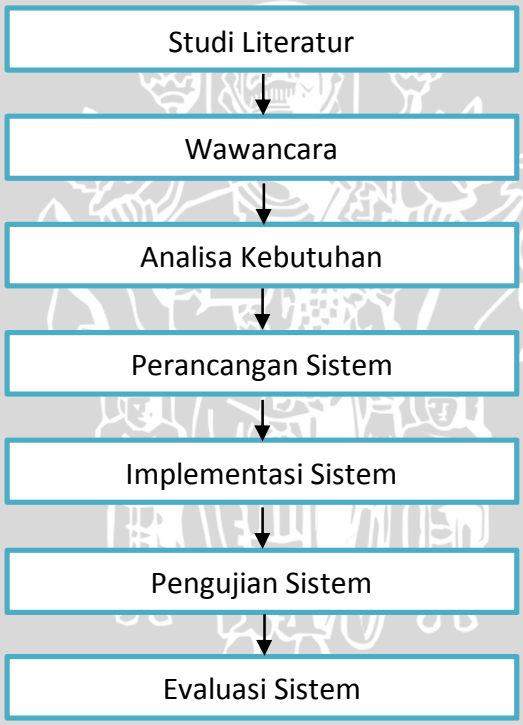
Apabila nilai yang sebenarnya identik dengan nilai hasil peramalan maka perhitungan *error* peramalan akan menghasilkan nilai 0% dan dapat dikatakan keakuratan peramalan adalah 100%. Sebaliknya apabila perhitungan *error* peramalan menghasilkan nilai 100% atau bahkan lebih dapat dikatakan bahwa keakuratan peramalan bernilai 0%. Oleh karena itu, pada penelitian ini apabila terdapat perhitungan *error* yang nilainya melebihi 100% maka nilai *error* akan ditarik ke batas 100% sehingga nilai MAPE hanya memiliki rentang 0 sampai 100% (Autry et al., 2013). Sedangkan penambahan nilai *c* atau konstanta pada rumus asli MAPE bertujuan untuk menghindari pembagian yang menghasilkan nilai tak terhingga atau tak tentu apabila terdapat data aktual dengan nilai 0. Nilai konstanta *c* dapat diisi dengan nilai yang sangat kecil seperti 0.0001.

# BAB 3 METODOLOGI DAN PERANCANGAN

Bab ini membahas tentang tahapan dalam pembuatan serta perancangan perangkat lunak sistem peramalan curah hujan menggunakan *Radial Basis Function Neural Network* (RBF) yang dioptimasi dengan *Hybrid Particle Swarm Optimization dan Genetic Algorithm* (HPSOGA). Tahapan penelitian meliputi analisis kebutuhan sistem, deskripsi umum sistem, deskripsi data yang digunakan, perancangan sistem, perhitungan manual proses peramalan curah hujan dengan RBF yang dioptimasi menggunakan HPSOGA, perancangan antarmuka, dan perancangan uji coba dan evaluasi.

## 3.1 Metode penelitian

Langkah-langkah dalam pembuatan sistem ini meliputi studi literatur, analisis dan perancangan sistem, pembuatan sistem, pengujian sistem, dan evaluasi hasil yang ditunjukkan pada Gambar 3.1.




**Gambar 3.1 Metode penelitian**

### 3.1.1 Studi literatur

Langkah pertama yang dilakukan dalam penelitian ini adalah mencari literatur yang berhubungan dengan optimasi *Radial Basis Function Neural Network* menggunakan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan. Tujuannya adalah untuk meningkatkan pengetahuan dan

pemahaman yang berkaitan dengan proses penelitian. Sumber literatur berasal dari jurnal, artikel, internet, ataupun buku terkait.

### 3.1.2 Wawancara

Wawancara dilakukan terhadap narasumber yang merupakan pakar klimatologi yaitu Ibu Dhenok Sulistyorini, SP selaku PMG Penyelia bagian Analisa dan Informasi dari Stasiun Klimatologi Kelas II Karangploso Malang. Tahapan ini bertujuan untuk memastikan kebenaran informasi yang telah didapat dari proses studi literatur dan menggali informasi lebih dalam mengenai curah hujan dan metode peramalannya untuk kepentingan penerapan sistem yang akan dibuat pada kehidupan nyata.

### 3.1.3 Analisis kebutuhan sistem

Analisis kebutuhan sistem merupakan tahapan yang untuk menganalisis semua kebutuhan yang diperlukan oleh sistem yang akan dibuat. Tahapan ini terdiri dari deskripsi umum sistem dan analisis kebutuhan data yang akan digunakan.

#### 3.1.3.1 Deskripsi umum sistem

Secara umum sistem yang akan dibangun dalam penelitian ini adalah sistem peramalan curah hujan yang mengimplementasikan algoritma *Radial Basis Function Neural Network* (RBF). Jumlah *node* tersembunyi dan parameter-parameter yang ada pada RBF seperti titik pusat vektor masukan, radius, dan bobot dari keluaran sebelumnya akan dioptimalkan menggunakan algoritma *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm*. Tujuan akhirnya adalah untuk mendapatkan hasil peramalan curah hujan dengan keakurasian yang paling maksimal.

#### 3.1.3.2 Deskripsi data

Pada penelitian ini, data yang digunakan sebagai objek penelitian adalah data curah hujan dasarian (10 harian) di Kabupaten Malang khususnya daerah Karangploso dari tahun 2009-2014. Data ini didapatkan dari Stasiun Klimatologi Kelas II Karangploso Malang.

### 3.1.4 Perancangan sistem

Perancangan sistem merupakan proses mendesain untuk mentransformasikan kebutuhan yang telah dianalisa sebelumnya menjadi sistem yang sebenarnya. Pada penelitian ini perancangan yang dilakukan meliputi perancangan alir sistem, perancangan antarmuka, dan perancangan uji serta evaluasi.

### 3.1.5 Implementasi sistem

Implementasi dalam penelitian ini dilakukan dengan menggunakan bahasa pemrograman C# serta *tools* pendukung lainnya. Masukan dari sistem adalah data data curah hujan dasarian, parameter-parameter GA, RBF, serta PSO. Sedangkan untuk keluaran dari sistem ini adalah hasil peramalan curah hujan dengan parameter RBF yang paling optimal serta nilai evaluasinya. Untuk tahapan-tahapan yang ada dalam proses implementasi antara lain:

1. Pembuatan antarmuka sistem
2. Normalisasi dan tabulasi data curah hujan untuk *preprocessing* data.
3. Perhitungan metode HPSOGA untuk mendapatkan parameter RBF yang paling optimal.
4. Perhitungan metode RBF dengan parameter yang paling optimal untuk peramalan curah hujan.
5. Keluaran berupa hasil peramalan curah hujan dan nilai evaluasi.

### 3.1.6 Pengujian

Pengujian pada penelitian ini dilakukan agar dapat menunjukkan bahwa sistem yang dibangun telah mampu bekerja sesuai dengan kebutuhan awal yang telah ditentukan. Pengujian sistem yang dilakukan dalam penelitian ini yaitu pengujian data latih dan data uji curah hujan serta pengujian parameter RBF, PSO, maupun GA.

### 3.1.7 Kesimpulan dan saran

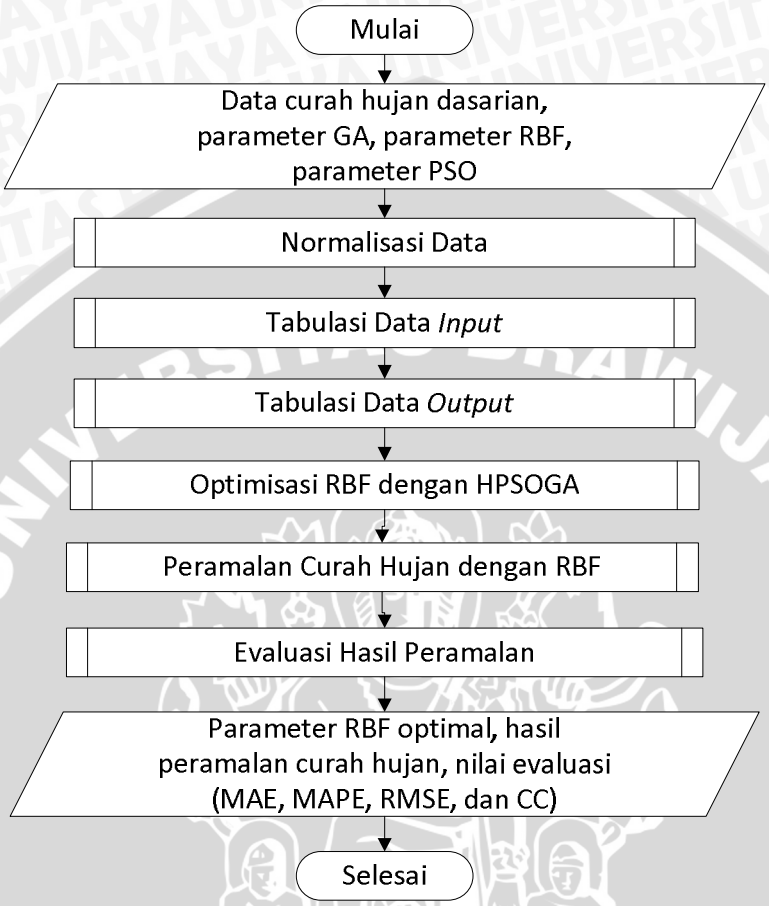
Pengambilan kesimpulan dilakukan setelah semua tahapan telah dilakukan. Kesimpulan diambil dari analisis hasil pengujian sistem yang telah dibuat untuk menjawab rumusan masalah yang telah ditetapkan sebelumnya. Kemudian terakhir adalah penerimaan saran yang dimaksudkan untuk memperbaiki kesalahan dari penulis dan membuka kesempatan pengembangan penelitian lebih lanjut dari pihak lain.

## 3.2 Alir perancangan sistem

Alir perancangan sistem untuk peramalan curah dengan algoritma *Radial Basis Function Neural Network* (RBF) yang dioptimasi menggunakan algoritma *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* (HPSOGA) ditunjukkan pada Gambar 3.2.

Pertama-tama sistem ini membutuhkan nilai masukan dari pengguna berupa data curah hujan dasarian yang akan dimodelkan menjadi vektor masukan pada jaringan syaraf tiruan RBF, parameter GA (jumlah populasi, jumlah iterasi maksimum, *crossover probability*, dan *mutation probability,* nilai *delta* tukar silang dan mutasi), parameter RBF (jumlah *node* masukan dan *node* tersembunyi dari jaringan), serta parameter PSO (*learning rate*). Data curah hujan kemudian akan

dinormalisasi dan ditabulasi terlebih dahulu sesuai dengan jumlah *node* masukan. Setelah itu sistem akan melakukan optimasi parameter-parameter RBF lainnya (jumlah *node* tersembunyi, titik pusat vektor masukan, radius, dan bobot dari keluaran) dengan HPSOGA.

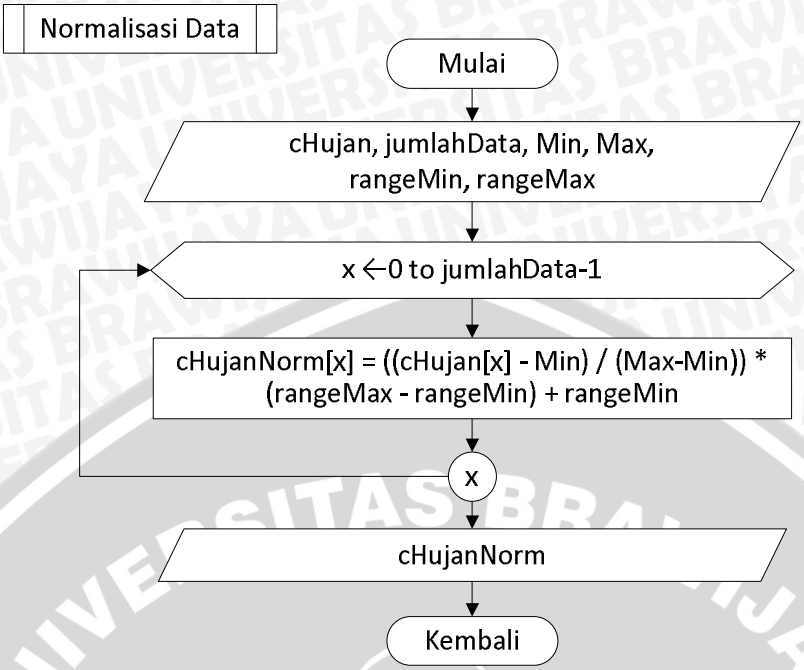



**Gambar 3.2 Diagram alir sistem**

Sesudah mendapatkan parameter RBF yang paling optimal, sistem akan melakukan peramalan curah hujan dengan RBF dengan parameter tersebut. Hasil peramalan akan dibandingkan dengan data curah hujan dasarian aktual untuk mendapatkan nilai kesalahan kemudian performa sistem akan dievaluasi dengan nilai evaluasi *Mean Absolute Error* (MAE), *Mean Absolute Percentage Error* (MAPE), *Root Mean Square Error* (RMSE), dan *Correlation Coefficient* (CC).

### 3.2.1 Normalisasi data

Pada bagian ini dijelaskan proses perhitungan normalisasi data curah hujan. Sebelum dilakukan pemrosesan data, data curah hujan dasarian dinormalisasi terlebih dahulu sehingga nilainya berada di rentang -10 dan 10. Proses normalisasi data ditunjukkan diagram alir pada Gambar 3.3.



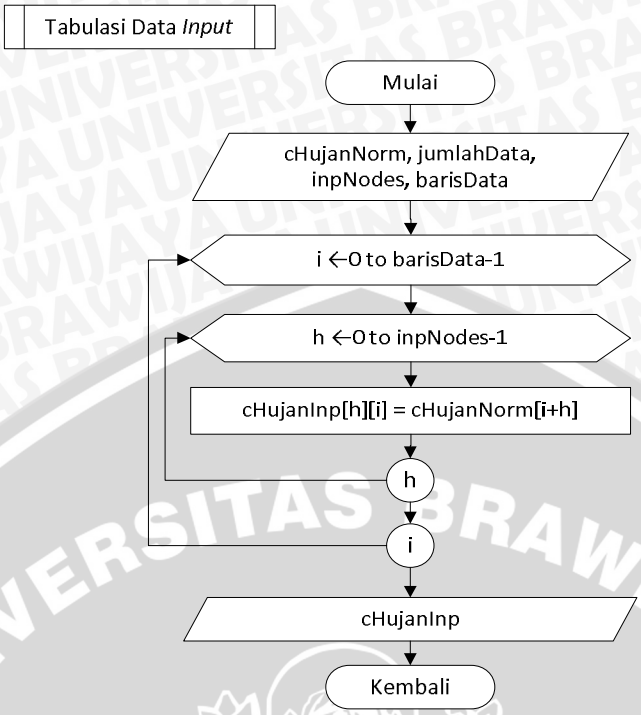

**Gambar 3.3 Diagram alir normalisasi data**

Penjelasan untuk Gambar 3.3 adalah sebagai berikut:

1. Memasukkan nilai curah hujan dasarian, jumlah data curah hujan, nilai curah hujan terendah dan tertinggi, serta *range* minimal dan maksimal dari data curah hujan ternormalisasi.
2. Menghitung normalisasi curah hujan dengan menggunakan Persamaan (2.18).
3. Melakukan proses perulangan sebanyak jumlah data curah hujan.
4. Menghasilkan keluaran berupa curah hujan hasil normalisasi untuk proses selanjutnya yaitu tabulasi data.

### 3.2.2 Tabulasi data *input* jaringan RBF

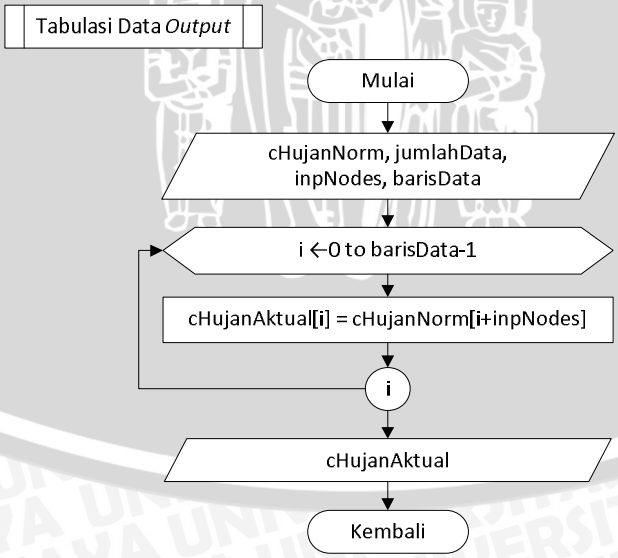
Setelah didapatkan data curah hujan ternormalisasi, langkah selanjutnya adalah mentabulasi data sebagai *node* masukan jaringan syaraf tiruan RBF untuk data latih. Data ditabulasi sesuai dengan pola berikut: apabila telah ditetapkan jumlah *node* masukan jaringan RBF sebesar n, maka untuk mendapatkan hasil prediksi peramalan curah hujan pada satu dasarian tertentu akan digunakan data masukan dari data curah hujan sebanyak n dasarian sebelumnya. Proses tabulasinya seperti ditunjukkan pada Gambar 3.4.




**Gambar 3.4 Diagram alir tabulasi data *input* jaringan RBF**

### 3.2.3 Tabulasi data *output* jaringan RBF

Setelah didapatkan *node* masukan jaringan syaraf tiruan RBF, langkah selanjutnya adalah mentabulasi data sebagai *node* keluaran jaringan syaraf tiruan RBF untuk data latih. Data ditabulasi sesuai dengan pola yang sama dengan sub bab sebelumnya. Proses tabulasinya seperti ditunjukkan pada Gambar 3.5.

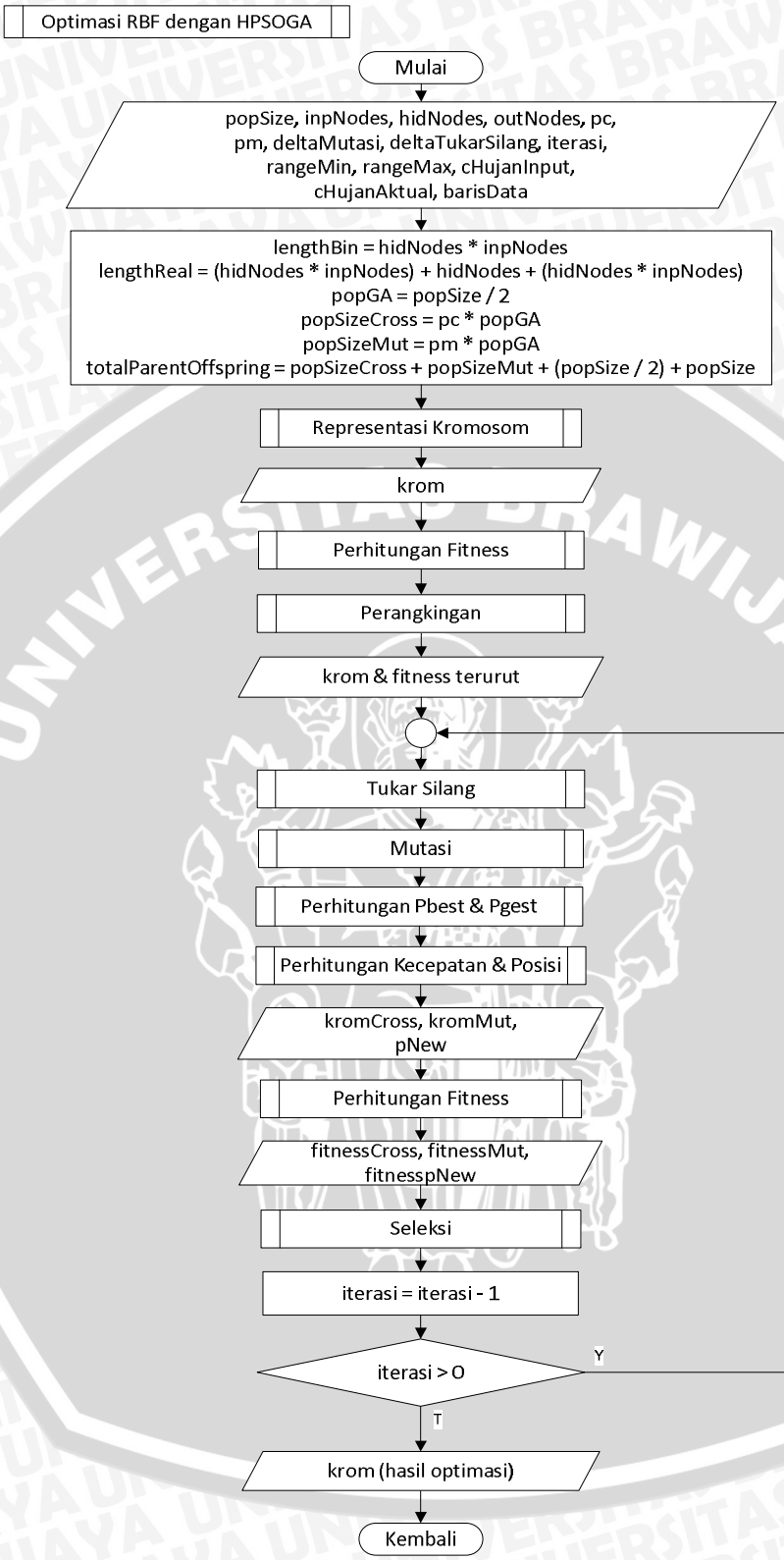



**Gambar 3.5 Diagram alir tabulasi data *output* jaringan RBF**

### 3.2.4 Optimasi RBF dengan HPSOGA

Pada proses pencarian parameter RBF paling optimal untuk peramalan curah hujan menggunakan HPSOGA terdapat beberapa tahapan yang harus dilakukan, seperti membangkitkan populasi awal, menghitung *fitness*, melakukan metode PSO dan GA untuk mendapatkan kromosom *offspring,* serta melakukan seleksi. Gambar 3.6 merupakan gambar diagram alir proses optimasi RBF dengan HPSOGA dengan penjelasan sebagai berikut :

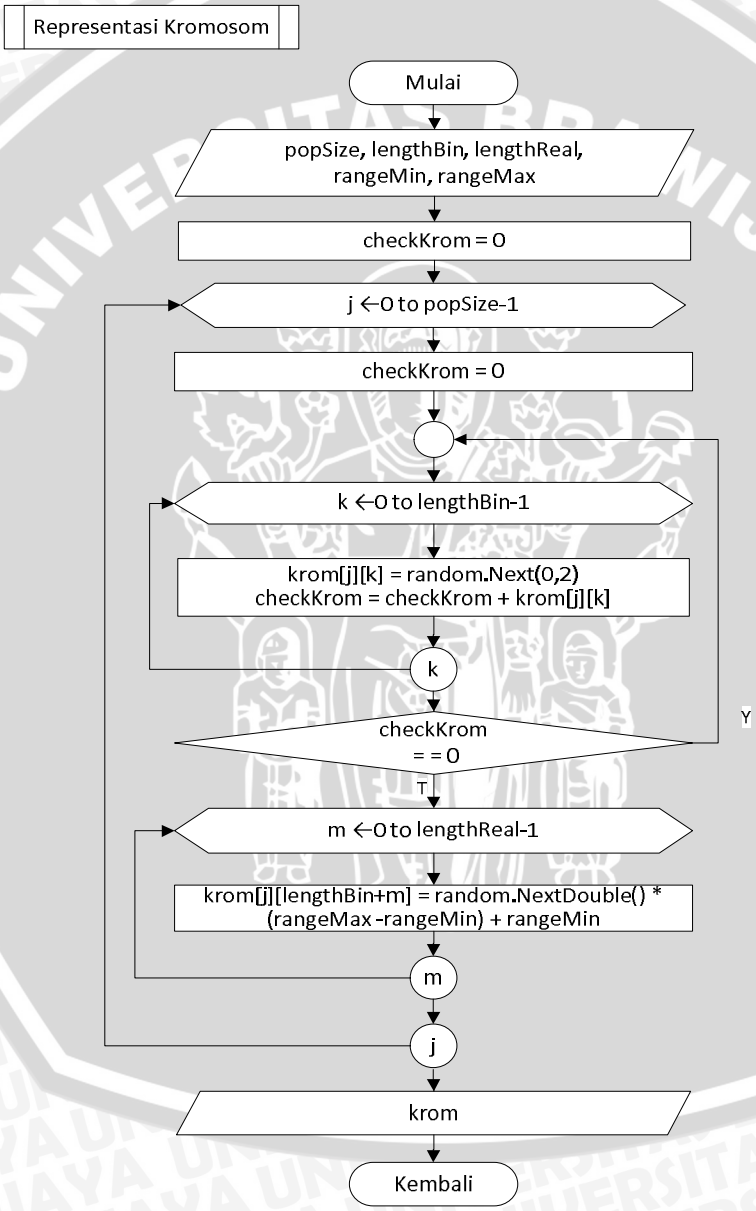
1. Memasukkan parameter-parameter yang akan digunakan dalam sistem, yaitu jumlah populasi, jumlah *node* masukan jaringan RBF, jumlah *node* tersembunyi jaringan RBF, jumlah *node* keluaran jaringan RBF, probabilitas tukar silang (*pc*), probabilitas mutasi (*pm*), nilai *delta* mutasi, nilai *delta* tukar silang, jumlah iterasi maksimal, rentang minimal dan maksimal nilai acak pembangkitan gen yang dikodekan dalam nilai real, data curah hujan masukan, data curah hujan keluaran yang sebenarnya, dan jumlah data masukan dan keluaran yang digunakan (baris data).
2. Menghitung panjang gen yang dikodekan dalam biner dan nilai real, jumlah populasi yang diproses dengan metode GA, jumlah populasi *offspring* hasil proses tukar silang, jumlah populasi *offspring* hasil proses mutasi, dan jumlah keseluruhan populasi *offspring.*
3. Membangkitkan kromosom awal yang berisi parameter-parameter metode RBF sebanyak jumlah populasi yang telah ditentukan.
4. Menghitung nilai *fitness* untuk masing-masing kromosom.
5. Melakukan perangkingan kromosom berdasarkan nilai *fitness.*
6. Melakukan proses tukar silang pada kromosom-kromosom yang memiliki nilai *fitness* tertinggi.
7. Melakukan proses mutasi pada kromosom-kromosom yang memiliki nilai *fitness* tertinggi.
8. Memperbarui nilai partikel *personal best* (*Pbest*) yang diambil dari kromosom-kromosom dengan nilai *fitness* terendah.
9. Memperbarui nilai kecepatan partikel yang dibentuk dari kromosom-kromosom dengan nilai *fitness* tertinggi dan memperbarui nilai posisi partikel yang dibentuk dari kromosom-kromosom dengan nilai *fitness* terendah.
10. Menghitung nilai *fitness* untuk kromosom *offspring* hasil metode GA dan PSO.
11. Menggabungkan semua *offspring* yang diperoleh dari metode GA dan PSO kemudian melakukan seleksi.
12. Melakukan pengecekan jumlah iterasi. Apabila jumlah iterasi belum mencapai jumlah iterasi maksimal maka kembali ke proses no. 6. Apabila sudah mencapai jumlah iterasi maksimal maka kromosom terbaik hasil seleksi terakhir adalah kromosom yang akan digunakan untuk peramalan curah hujan.




**Gambar 3.6 Diagram alir optimasi RBF dengan HPSOGA**

### 3.2.4.1 Representasi kromosom

Dalam penelitian ini parameter dari jaringan RBF yang ingin dioptimasi dikodekan dalam *string* biner dan real. *Node* tersembunyi akan dikodekan sebagai *string* biner dimana 1 menunjukkan adanya koneksi ke *node* masukan dan keluaran serta 0 menunjukkan tidak adanya koneksi. Kemudian titik pusat vektor masukan, radius, dan bobot dari keluaran akan dikodekan sebagai *string* real. Satu komponen jaringan RBF akan dianggap sebagai satu gen penyusun kromosom individu. Proses pembentukan representasi kromosom ditunjukkan diagram alir pada Gambar 3.7




**Gambar 3.7 Diagram alir representasi kromosom**

Untuk gen dalam kromosom yang dikodekan dalam biner proses pembentukannya adalah dengan membangkitkan nilai acak 0 atau 1. Sedangkan untuk yang dikodekan dalam nilai real dibentuk dengan membangkitkan nilai acak pada rentang -10 sampai 10. Panjang gen yang dikodekan dalam biner serta nilai real dapat dihitung dengan Persamaan (2.7) dan (2.8), sedangkan untuk panjang kromosom dapat dihitung dengan Persamaan (2.9). Penjelasan untuk Gambar 3.7 adalah berikut ini:

1. Memasukkan jumlah populasi, panjang gen yang dikodekan dalam biner dan nilai real, serta rentang minimal dan maksimal nilai acak pembangkitan gen yang dikodekan dalam nilai real.
2. Membangkitkan nilai antara 0 atau 1 sebanyak panjang gen yang dikodekan dalam biner.
3. Apabila hasil pengecekan proses pembangkitan *string* biner bernilai 0 (tidak ada koneksi dari *node* masukan ke *node* tersembunyi karena tidak memiliki nilai 1 sama sekali) maka ulangi langkah sebelumnya.
4. Membangkitkan nilai real pada rentang minimal dan maksimal yang telah dimasukkan sebanyak panjang gen yang dikodekan dalam real.
5. Melakukan proses perulangan sebanyak jumlah populasi kromosom.
6. Menghasilkan keluaran berupa kromosom untuk proses selanjutnya yaitu perhitungan *fitness*.

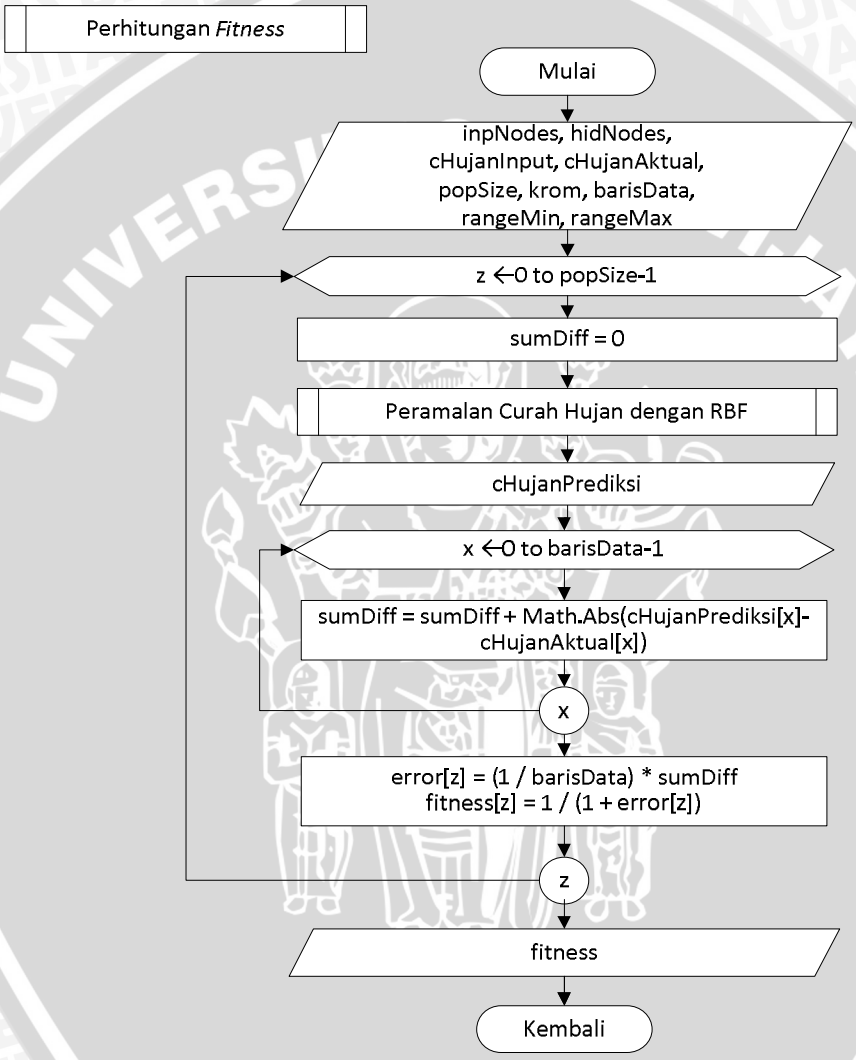
### 3.2.4.2 Perhitungan *fitness*

Dalam proses perhitungan nilai *fitness*, semua kromosom pertama-tama akan digunakan untuk melakukan peramalan curah hujan dengan metode RBF. Setelah mendapatkan nilai keluaran dari jaringan RBF, kemudian nilai *error* dari hasil peramalan tersebut dihitung dengan menggunakan Persamaan (2.10) dan nilai *fitness* dengan menggunakan Persamaan (2.11). Nilai *fitness* berbanding terbalik dengan nilai *error* dari hasil peramalan. Semakin kecil nilai *error* dari hasil peramalan maka semakin besar nilai *fitness* dari sebuah kromosom. Semakin besar nilai *fitness* dari sebuah kromosom maka semakin besar pula kemungkinan kromosom itu untuk terpilih sebagai solusi di iterasi berikutnya.

Berdasarkan diagram alir pada Gambar 3.8, proses perhitungan *fitness* dapat dijelaskan sebagai berikut :

1. Memasukkan semua kromosom, data curah hujan masukan, data curah hujan keluaran yang sebenarnya, jumlah *node* masukan dan tersembunyi, jumlah populasi, serta jumlah data masukan dan keluaran yang digunakan (baris data).
2. Melakukan peramalan curah hujan dengan metode RBF.
3. Menghasilkan keluaran berupa hasil peramalan curah hujan.
4. Menghitung perbedaan nilai antara hasil peramalan curah hujan dengan data curah hujan keluaran yang sebenarnya untuk keperluan perhitungan nilai *error.*

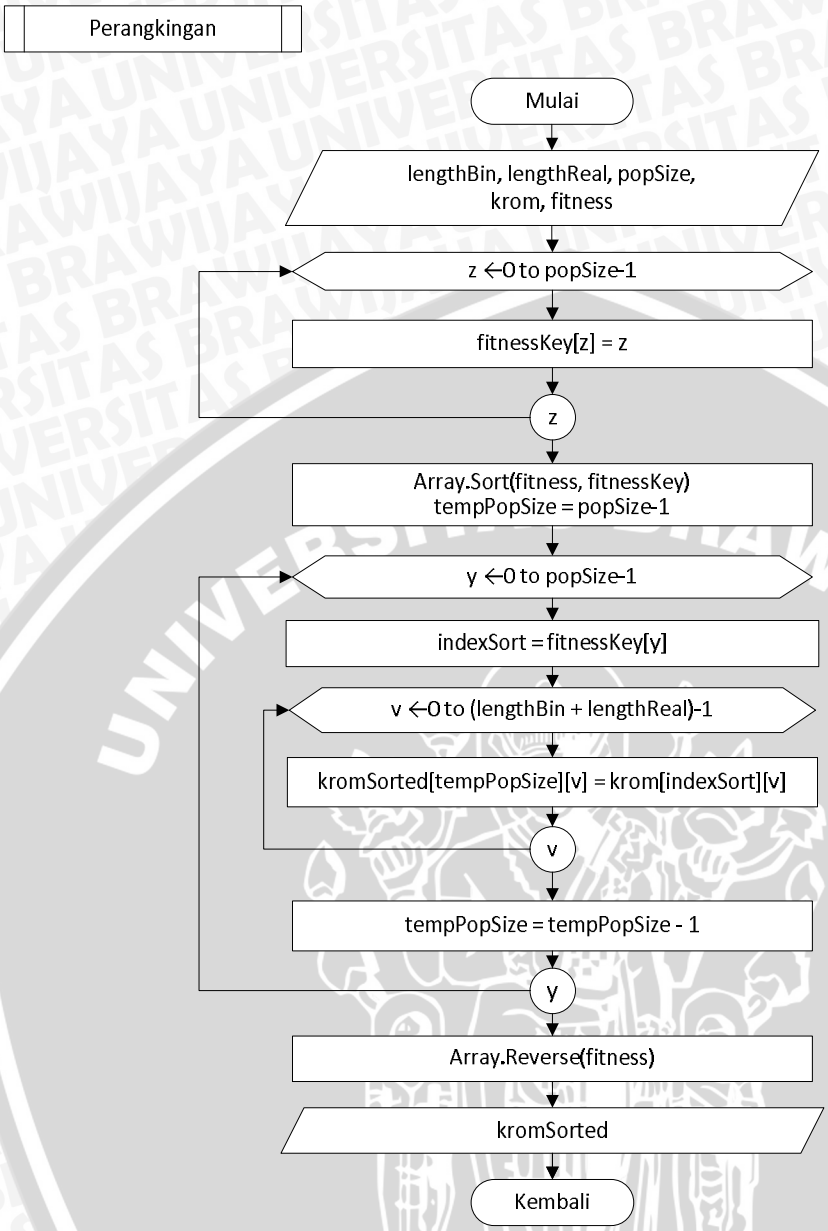
5. Melakukan proses perulangan sebanyak jumlah data masukan dan keluaran yang digunakan.
6. Menghitung nilai *error* dari hasil peramalan dengan menggunakan Persamaan (2.10).
7. Menghitung nilai *fitness* kromosom dengan menggunakan Persamaan (2.11).
8. Melakukan proses perulangan sebanyak jumlah populasi kromosom.
9. Menghasilkan keluaran berupa *fitness* kromosom untuk proses selanjutnya yaitu perangkingan.




**Gambar 3.8 Diagram alir perhitungan *fitness***

### 3.2.4.3 Perangkingan

Proses perangkingan kromosom dilakukan dengan cara mengurutkan kromosom berdasarkan nilai *fitness* yang terbesar hingga yang terkecil. Proses perangkingan ditunjukkan pada Gambar 3.9.




**Gambar 3.9 Diagram alir perangkingan**

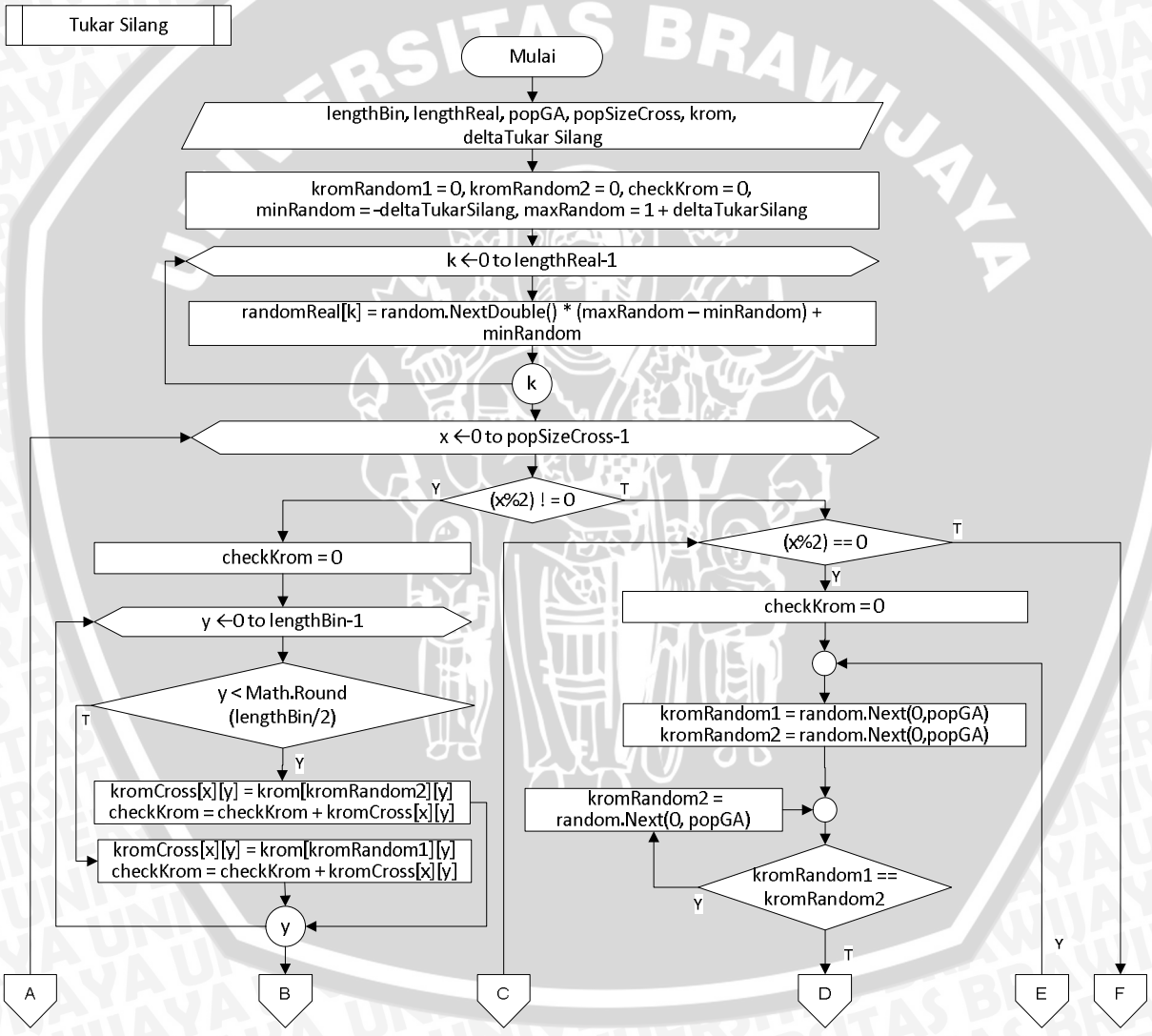
Berdasarkan Gambar 3.9 proses perangkingan kromosom dapat dijelaskan sebagai berikut:

1. Memasukkan semua kromosom, *fitness*, jumlah populasi, serta panjang gen yang dikodekan dalam biner dan nilai real.
2. Membangkitkan *array* untuk key dari *fitness*.
3. Mengurutkan nilai *fitness* kromosom dari terendah ke tertinggi beserta dengan *key*-nya (indeks *array fitness*-nya).

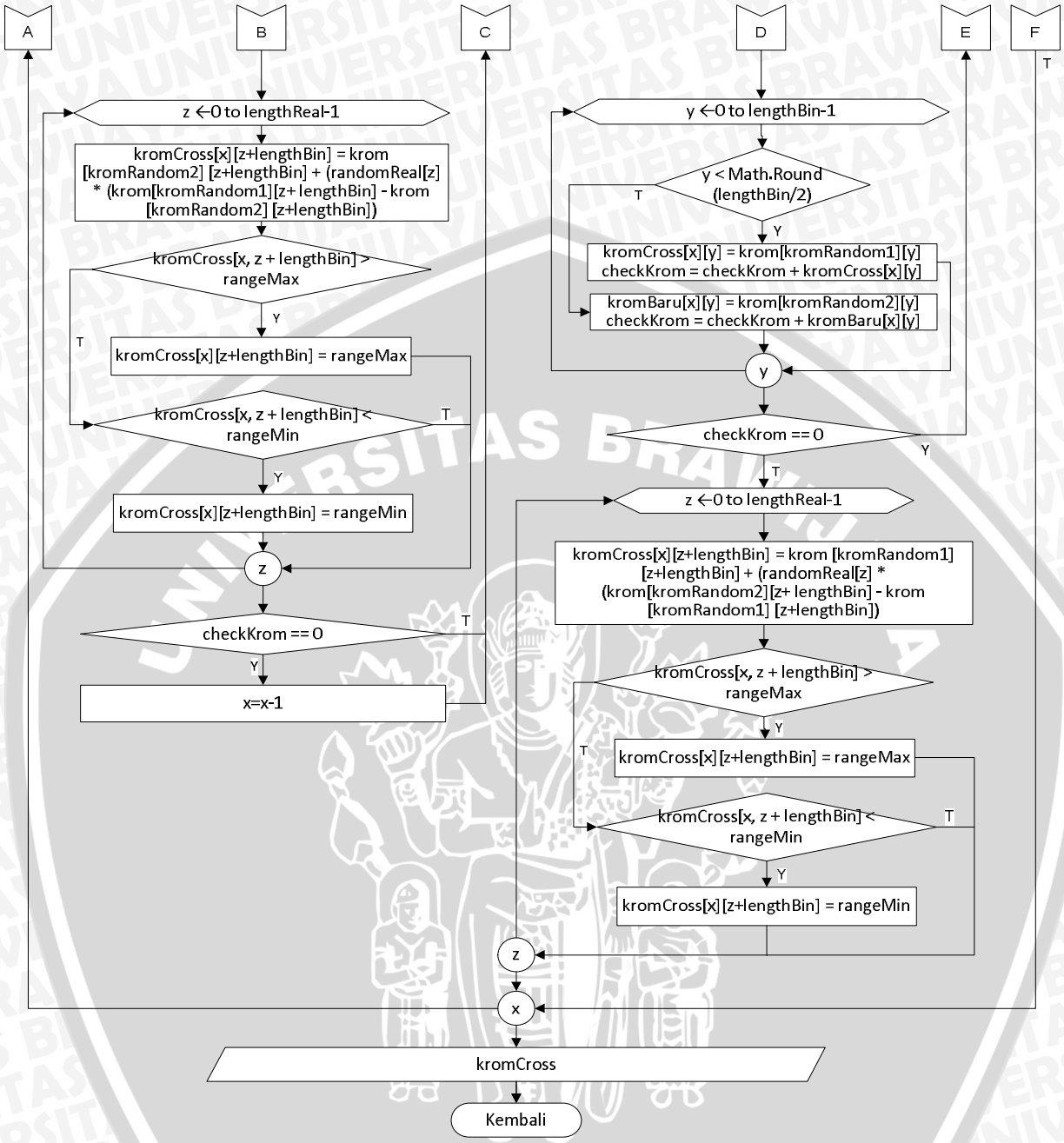
4. Menyalin krom dengan nilai *fitness* terendah ke tertinggi ke dalam kromSorted dimulai dari indeks yang paling besar.
5. Membalik urutan *array fitness* dari *fitness* terendah ke tertinggi menjadi tertinggi ke terendah.
6. Menghasilkan keluaran berupa kromosom yang telah diurutkan berdasarkan nilai *fitness* tertinggi ke terendah untuk proses selanjutnya.

### 3.2.4.4 Tukar silang

Pada proses tukar silang, akan dihasilkan *offspring* sebanyak hasil perkalian antara jumlah populasi yang diproses algoritma genetika dengan probabilitas tukar silang (pc). Proses tukar silang ditunjukkan diagram alir pada Gambar 3.10.




**Gambar 3.10 Diagram alir tukar silang**




**Gambar 3.10 Diagram alir tukar silang (lanjutan)**

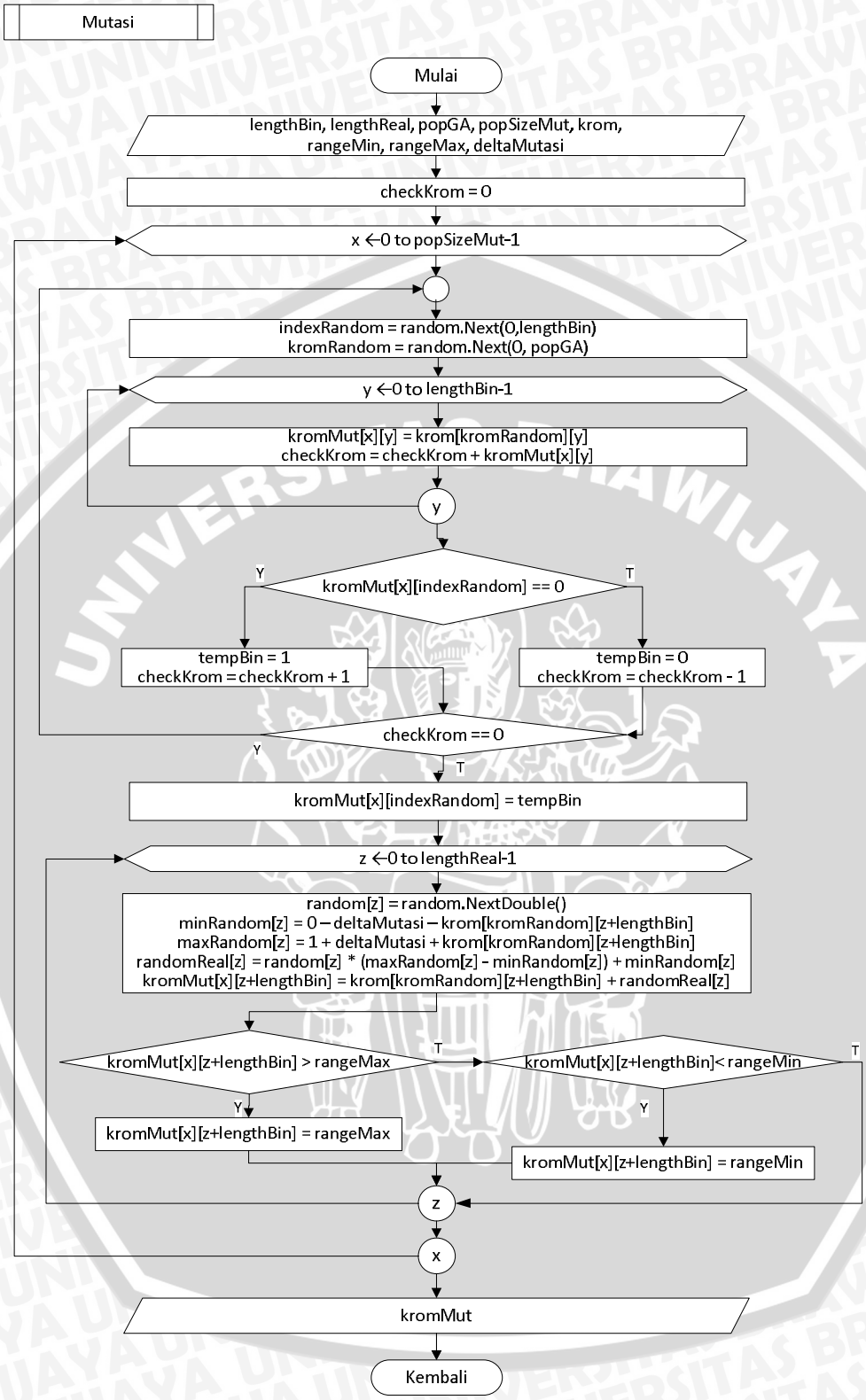
Proses tukar silang pada Gambar 3.10 dapat dijelaskan sebagai berikut:

1. Memasukkan semua kromosom, jumlah populasi metode GA, jumlah populasi hasil proses tukar silang, panjang gen yang dikodekan dalam biner dan nilai real, serta nilai *delta* tukar silang.
2. Membangkitkan nilai acak yang akan digunakan dalam proses tukar silang kromosom pengkodean nilai real dengan rentang [-*delta*, 1+*delta*].

3. Inisialisasi nilai x = 0 sampai dengan jumlah populasi tukar silang – 1.
4. Apabila x bernilai ganjil lanjutkan ke langkah no 5. Apabila tidak lanjutkan ke langkah no 8.
5. Melakukan proses tukar silang untuk gen yang dikodekan dalam biner dengan metode tukar silang *one-point*.
6. Melakukan proses tukar silang untuk gen yang dikodekan dalam nilai real dengan Persamaan (2.12).
7. Melakukan pengecekan hasil kromosom tukar silang. Apabila nilai kromosom *offspring* lebih dari rentang maksimal maka nilai kromosom *offspring* akan diisi dengan rentang maksimal tersebut. Sebaliknya nilai kromosom *offspring* kurang dari rentang minimal maka nilai kromosom *offspring* akan diisi dengan rentang minimal tersebut.
8. Apabila hasil pengecekan proses tukar silang pada *string* biner bernilai 0 maka kurangi x dengan 1 (hapus satu kromosom yang telah dibentuk sebelumnya) dan lakukan proses pemilihan *parent* kembali pada langkah no 8.
9. Apabila x bernilai genap (belum melakukan pemilihan *parent* untuk tukar silang) maka lanjutkan ke langkah no 9. Apabila tidak lanjutkan ke langkah no 13.
10. Memilih 2 *parent* secara acak dari populasi metode GA. Apabila *parent* yang dipilih sama maka ulangi pemilihan *parent* kedua sampai *parent*-nya berbeda.
11. Melakukan proses tukar silang untuk gen yang dikodekan dalam biner dengan metode tukar silang one-point.
12. Apabila hasil pengecekan proses tukar silang pada *string* biner bernilai 0 (tidak ada koneksi dari *node* masukan ke *node* tersembunyi karena tidak memiliki nilai 1 sama sekali) maka kembali ke langkah no 9.
13. Melakukan proses tukar silang untuk gen yang dikodekan dalam nilai real dengan Persamaan (2.12).
14. Melakukan pengecekan hasil kromosom tukar silang. Apabila nilai kromosom *offspring* lebih dari rentang maksimal maka nilai kromosom *offspring* akan diisi dengan rentang maksimal tersebut. Sebaliknya nilai kromosom *offspring* kurang dari rentang minimal maka nilai kromosom *offspring* akan diisi dengan rentang minimal tersebut.
15. Melakukan proses perulangan sampai nilai x mencapai nilai maksimal.
16. Menghasilkan keluaran berupa kromosom hasil tukar silang untuk proses selanjutnya yaitu seleksi.

#### 3.2.4.5 Mutasi

Pada proses mutasi, seperti halnya pada proses tukar silang, akan dihasilkan *offspring* sebanyak hasil perkalian antara jumlah populasi yang diproses algoritma genetika dengan probabilitas mutasi (pm). Proses mutasi ditunjukkan oleh diagram alir pada Gambar 3.11.



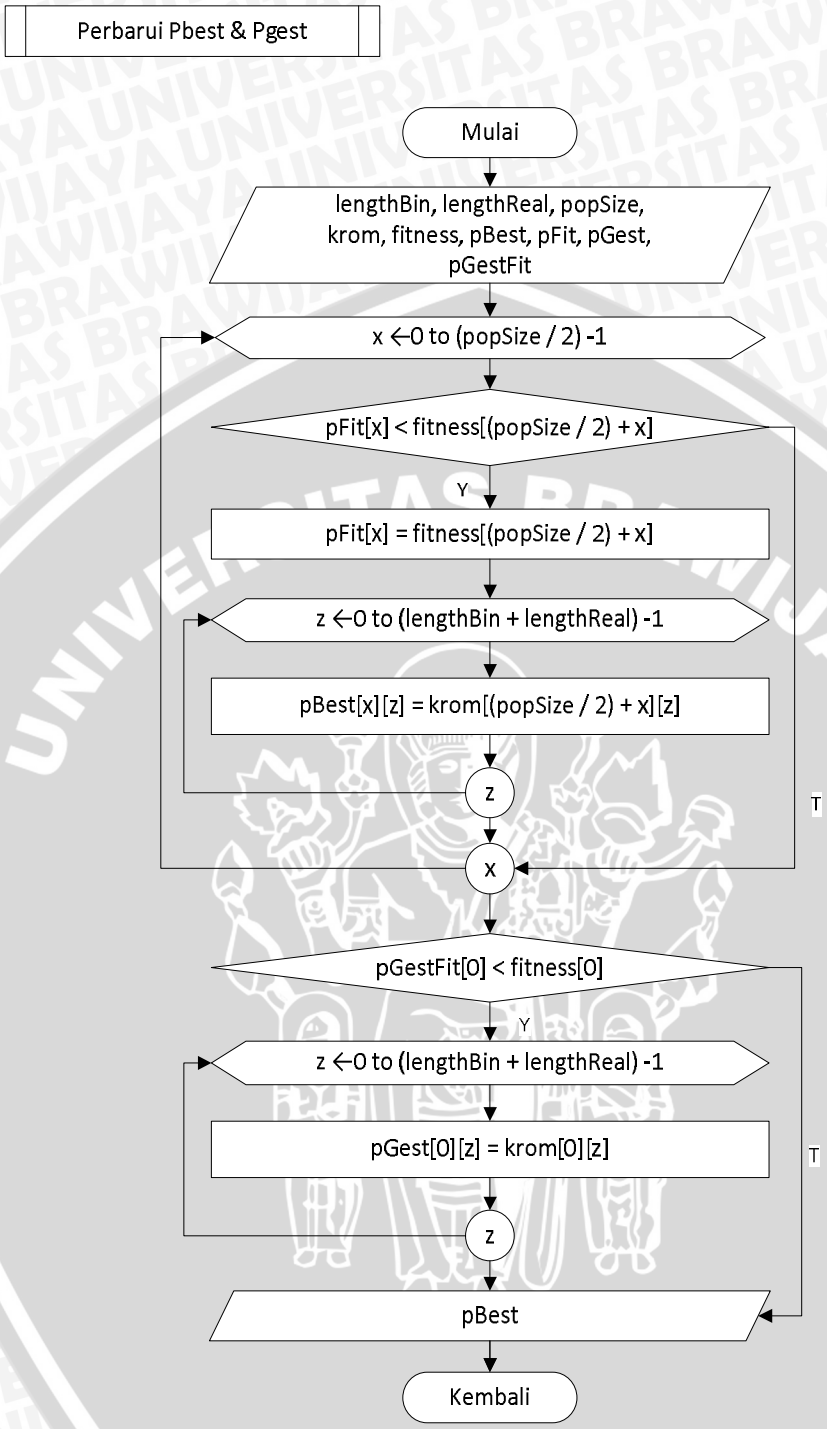

**Gambar 3.11 Diagram alir mutasi**

Gambar 3.11 dapat dijelaskan seperti berikut ini:

1. Memasukkan semua kromosom, jumlah populasi metode GA, jumlah populasi hasil proses mutasi, panjang gen yang dikodekan dalam biner dan nilai real, rentang minimal dan maksimal nilai acak pembangkitan gen yang dikodekan dalam nilai real, serta nilai *delta* mutasi.
2. Memilih 1 *parent* secara acak dari populasi metode GA serta 1 indeks secara acak dari gen yang dikodekan dalam biner.
3. Melakukan proses mutasi untuk gen yang dikodekan dalam biner dengan metode mutasi *bitwise* sesuai dengan indeks gen yang telah dipilih sebelumnya.
4. Apabila hasil pengecekan proses mutasi pada *string* biner bernilai 0 (tidak ada koneksi dari *node* masukan ke *node* tersembunyi karena tidak memiliki nilai 1 sama sekali) maka kembali ke langkah no 2.
5. Melakukan proses mutasi untuk gen yang dikodekan dalam nilai real dengan Persamaan (2.13).
6. Apabila nilai gen hasil mutasi yang dikodekan dalam nilai real lebih dari rentang maksimal maka hasil mutasi akan diisi dengan rentang maksimal tersebut.
7. Apabila nilai gen hasil mutasi yang dikodekan dalam nilai real kurang dari rentang minimal maka hasil mutasi akan diisi dengan rentang minimal tersebut.
8. Melakukan proses perulangan sebanyak jumlah populasi hasil proses mutasi.
9. Menghasilkan keluaran berupa kromosom hasil mutasi untuk proses selanjutnya yaitu seleksi.

#### 3.2.4.6 Perbarui partikel *personal best* (*Pbest*) dan global best (*Pgest*)

Langkah setelah melakukan pembentukan *offspring* dengan metode GA adalah memperbarui partikel *personal best* dan *global best* untuk keperluan pembentukan *offspring* dengan metode PSO. Nilai partikel *personal best* (*Pbest*) akan diambil dari kromosom dengan nilai *fitness* 2N terendah yang terurut (pada proses selanjutnya akan menjadi posisi partikel). Sedangkan partikel *global best* (*Pgest*) nantinya akan selalu diambil dari kromosom dengan indeks 0 karena memiliki *fitness* tertinggi dibandingkan partikel yang lainnya. Proses perbaruan partikel *personal best* dan *global best* ini ditunjukkan oleh diagram alir pada Gambar 3.12.

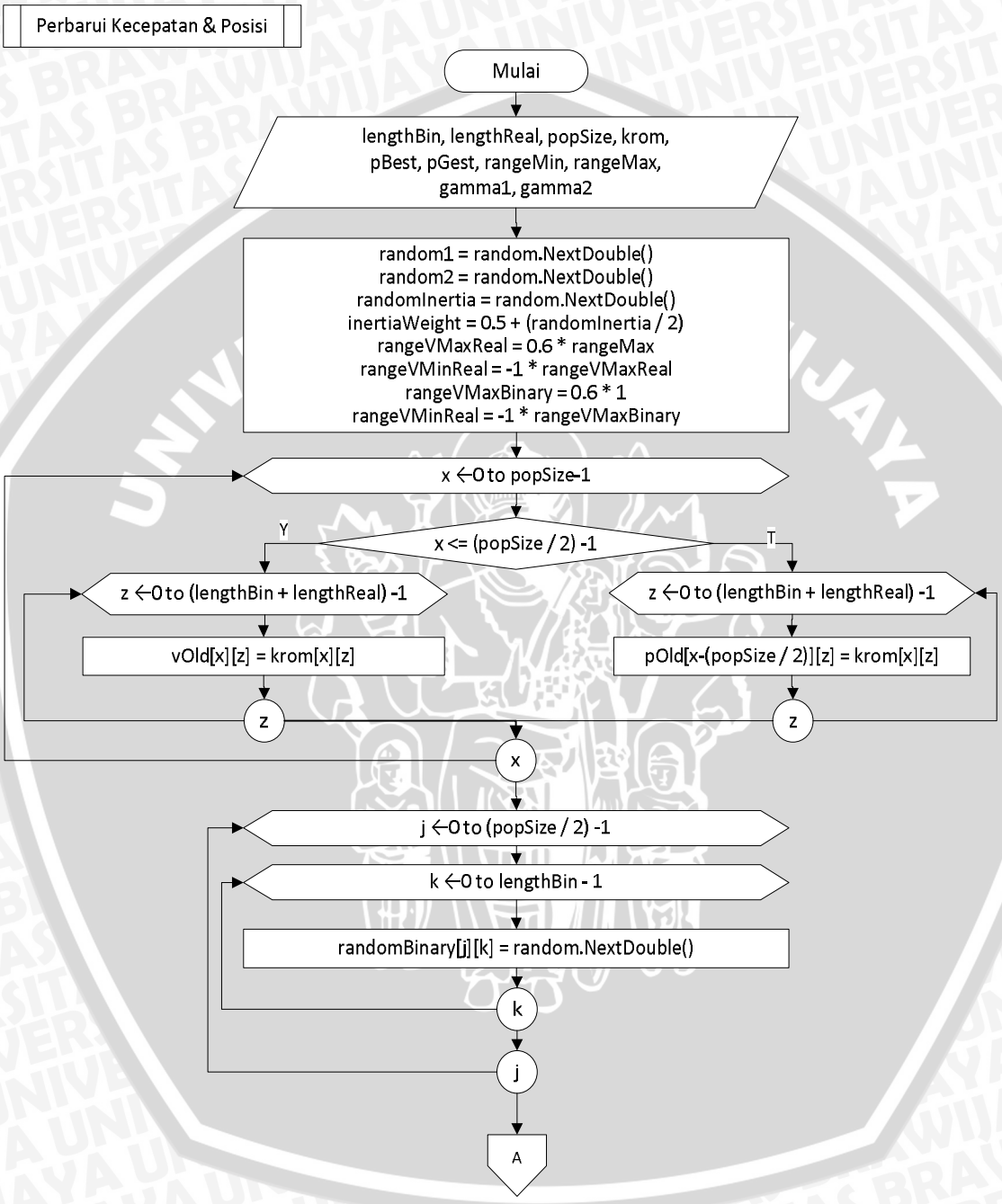



**Gambar 3.12 Diagram alir perbarui partikel *personal best* (*Pbest*) dan g*lobal best* (*Pgest*)**

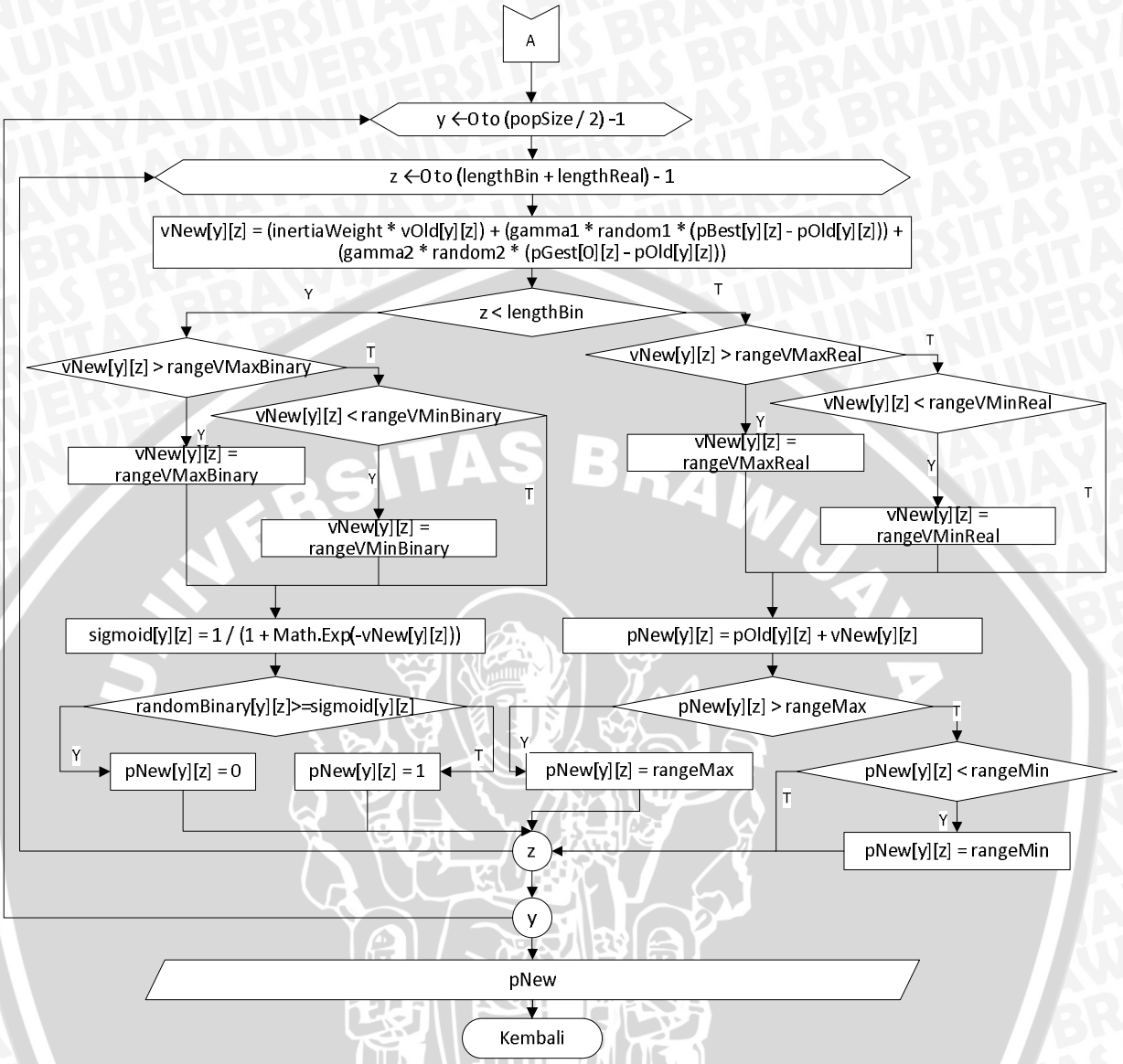
### 3.2.4.7 Perbarui kecepatan dan posisi partikel

Langkah selanjutnya adalah melakukan pembentukan *offspring* menggunakan metode PSO. Pertama-tama kromosom elite digunakan sebagai kecepatan partikel

serta 2N kromosom dengan nilai *fitness* terendah digunakan sebagai posisi partikel. Kemudian diperbarui kecepatan dan posisi partikelnya untuk mendapatkan *offspring* baru. Proses ini ditunjukkan oleh diagram alir pada Gambar 3.13.




**Gambar 3.13 Diagram alir perbarui kecepatan dan posisi partikel**




**Gambar 3.13 Diagram alir perbarui kecepatan dan posisi partikel (lanjutan)**

Gambar 3.13 dapat dijelaskan sebagai berikut:

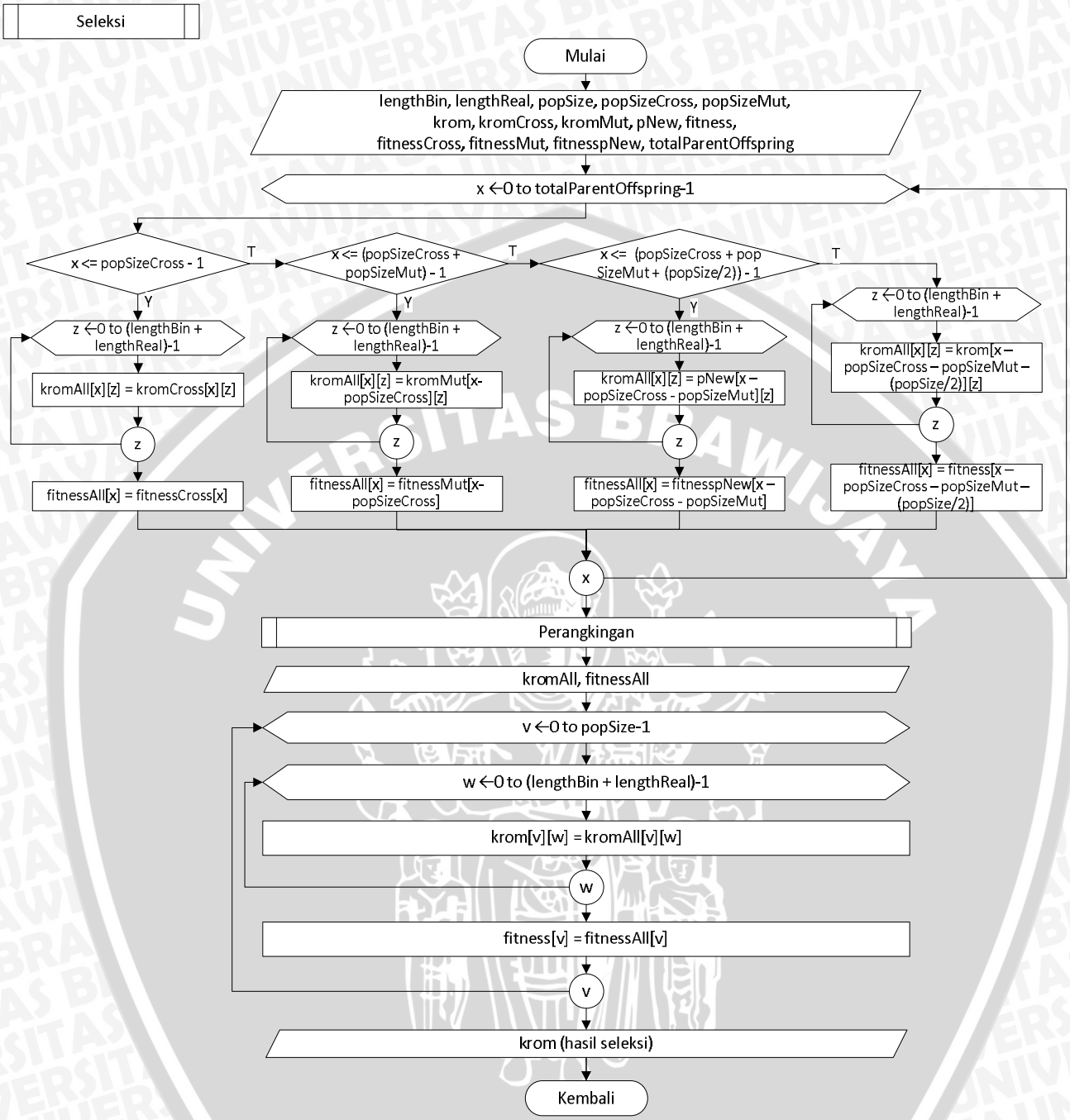
1. Memasukkan semua kromosom, jumlah populasi keseluruhan, panjang gen yang dikodekan dalam biner dan nilai real, rentang minimal dan maksimal nilai acak pembangkitan gen yang dikodekan dalam nilai real, *learning rate* atau *gamma* 1 ($\gamma_1$) dan *gamma* 2 ($\gamma_2$), serta partikel *personal best*.
2. Melakukan pembentukan kecepatan partikel awal dengan menggunakan kromosom elite dan pembentukan posisi partikel awal dengan menggunakan kromosom yang memiliki nilai *fitness* terendah.
3. Membangkitkan nilai acak yang akan digunakan untuk perbandingan dalam menghitung posisi baru pada partikel pengkodean nilai biner.

4. Menghitung kecepatan baru untuk partikel yang dikodekan dalam nilai real dan biner dengan Persamaan (2.15).
5. Apabila nilai kecepatan partikel baru biner lebih dari rentang maksimal kecepatan biner maka kecepatan akan diisi dengan rentang maksimal biner tersebut, dan sebaliknya.
6. Melakukan pengecekan sama dengan no 5 untuk kecepatan partikel baru real dengan melihat rentang minimal dan maksimal kecepatan partikel baru real.
7. Menghitung nilai fungsi *sigmoid* untuk partikel yang dikodekan dalam nilai biner dengan Persamaan (2.17).
8. Apabila hasil fungsi *sigmoid* lebih besar dari nilai acak yang telah dibangkitkan pada langkah no. 3, maka posisi baru untuk partikel yang dikodekan dalam nilai biner akan diisi dengan 0.
9. Apabila hasil fungsi *sigmoid* lebih kecil dari nilai acak yang telah dibangkitkan pada langkah no. 3, maka posisi baru untuk partikel yang dikodekan dalam nilai biner akan diisi dengan 1.
10. Menghitung nilai posisi baru untuk partikel yang dikodekan dalam nilai real dengan Persamaan (2.16).
11. Apabila nilai posisi partikel baru lebih dari rentang maksimal maka posisi partikel yang dikodekan dalam nilai real akan diisi dengan rentang maksimal tersebut.
12. Apabila nilai posisi partikel baru kurang dari rentang minimal maka posisi partikel yang dikodekan dalam nilai real akan diisi dengan rentang minimal tersebut.
13. Menghasilkan keluaran berupa posisi partikel terbaru untuk proses selanjutnya yaitu seleksi.

#### 3.2.4.8 Seleksi

Proses seleksi dilakukan untuk memperoleh hasil kromosom terbaik yang akan dijadikan populasi pada iterasi selanjutnya. Proses seleksi disini menggunakan metode *elitism* yaitu mendapatkan individu terbaik sejumlah ukuran populasi yang telah ditentukan berdasarkan nilai *fitness* yang diurutkan dari tertinggi ke terendah. Proses seleksi ditunjukkan oleh diagram alir pada Gambar 3.14 yang dapat dijelaskan sebagai berikut:

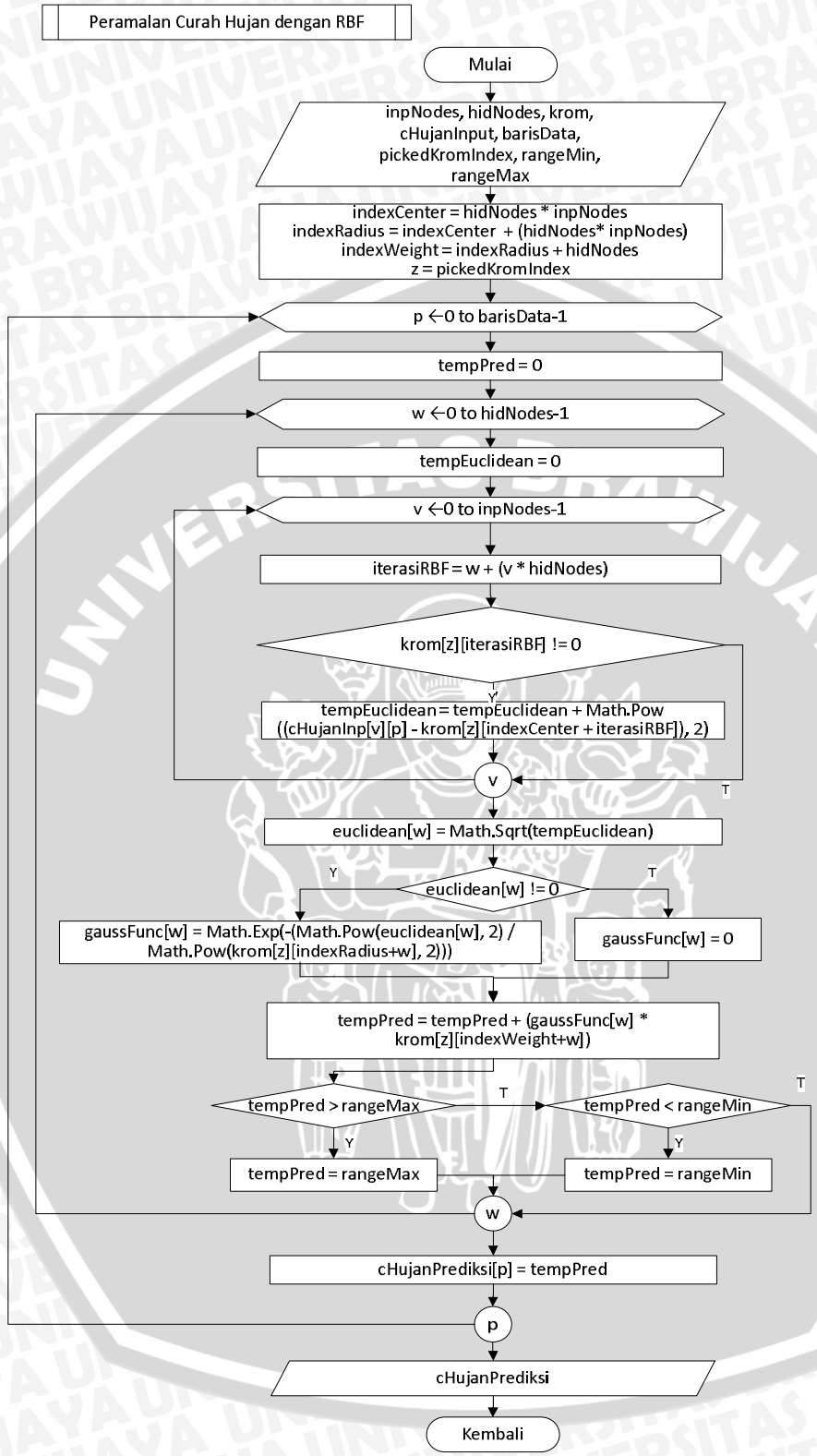
1. Memasukkan semua kromosom awal, kromosom *offspring* hasil metode GA dan PSO, jumlah populasi keseluruhan, jumlah populasi hasil tukar silang dan mutasi, *fitness* kromosom awal, *fitness* kromosom *offspring* hasil metode GA dan PSO, panjang gen yang dikodekan dalam biner dan nilai real.
2. Menggabungkan kromosom awal dengan kromosom hasil *offspring* metode GA dan PSO.
3. Melakukan perangkingan berdasarkan nilai *fitness*.
4. Membentuk populasi baru dengan kromosom-kromosom yang memiliki nilai *fitness* tertinggi terurut.
5. Menghasilkan keluaran berupa populasi baru untuk proses selanjutnya.




**Gambar 3.14 Diagram alir seleksi**

### 3.2.5 Peramalan curah hujan dengan RBF

Pada tahap ini parameter-parameter yang telah didapatkan pada proses sebelumnya digunakan untuk melakukan peramalan curah hujan dengan RBF pada data latih ataupun uji. Proses peramalan curah hujan dengan metode RBF ditunjukkan diagram alir pada Gambar 3.15.




**Gambar 3.15 Diagram alir peramalan curah hujan dengan RBF**

Gambar 3.15 dapat dijelaskan seperti berikut ini:

1. Memasukkan semua kromosom, data curah hujan masukan, jumlah *node* masukan dan tersembunyi, indeks kromosom terpilih, serta jumlah data masukan dan keluaran yang digunakan (baris data).
2. Apabila kromosom yang menunjukkan koneksi dari *node* masukan sampai ke *node* keluaran tidak bernilai 0 maka hitung *Euclidean norm* antara vektor masukan dengan titik pusat vektor masukan dengan Persamaan (2.2). Apabila kromosom tersebut bernilai 0 maka *Euclidean norm* bernilai 0.
3. Apabila hasil perhitungan *Euclidean norm* tidak sama dengan 0, maka hitung fungsi *radial basis* tiap *node* tersembunyi dengan menggunakan Persamaan (2.1). Apabila hasil perhitungan *Euclidean norm* bernilai 0, maka fungsi *radial basis* tiap *node* tersembunyi bernilai 0.
4. Menghitung nilai keluaran (prediksi) dari jaringan RBF dengan menggunakan Persamaan (2.3).
5. Menghasilkan keluaran berupa hasil prediksi curah hujan untuk proses selanjutnya.

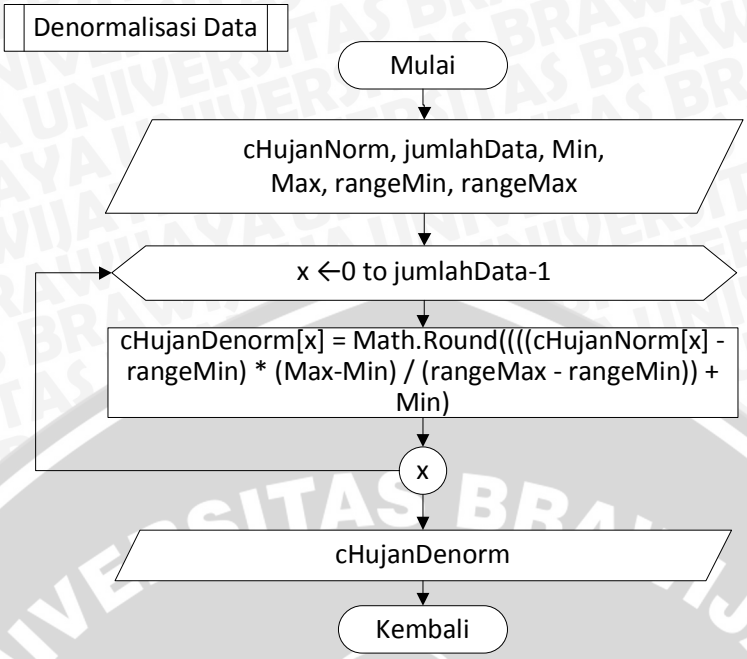
### 3.2.6 Evaluasi hasil peramalan curah hujan dengan RBF-HPSOGA

Setelah mendapatkan hasil peramalan curah hujan menggunakan metode RBF, diperlukan proses evaluasi untuk mengetahui performa model peramalan yang diterapkan. Proses evaluasi ini meliputi proses denormalisasi data (baik data curah hujan uji aktual maupun curah hujan hasil prediksi) dan perhitungan evaluasi.

#### 3.2.6.1 Denormalisasi data

Sebelum dilakukan proses evaluasi, data curah hujan uji aktual maupun curah hujan hasil prediksi perlu di denormalisasi terlebih dahulu sehingga nilainya kembali berada di rentang data curah hujan yang sebenarnya. Gambar 3.16 merupakan gambar diagram alir proses denormalisasi data dengan penjelasan sebagai berikut:

1. Memasukkan nilai curah hujan uji dalam bentuk normalisasi, jumlah data curah hujan uji, nilai curah hujan terendah dan tertinggi, serta range minimal dan maksimal dari data curah hujan ternormalisasi.
2. Menghitung denormalisasi curah hujan dengan menggunakan Persamaan (2.19).
3. Melakukan proses perulangan sebanyak jumlah data curah hujan uji.
4. Menghasilkan keluaran berupa curah hujan hasil denormalisasi untuk proses selanjutnya yaitu evaluasi.

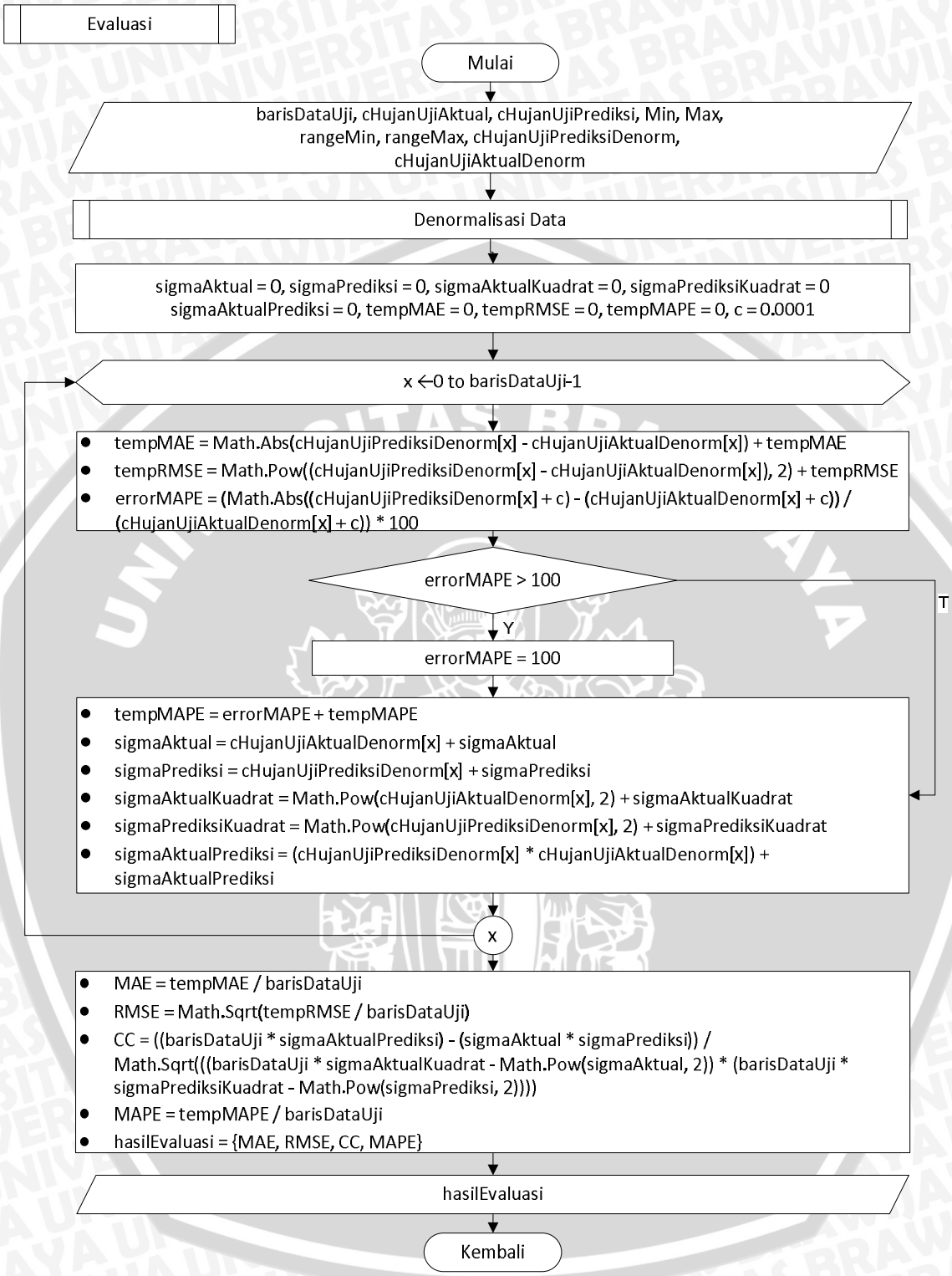



**Gambar 3.16 Diagram alir denormalisasi data**

### 3.2.6.2 Evaluasi

Dalam penelitian ini proses evaluasi dilakukan dengan empat macam statistik standar yaitu MAE, MAPE, RMSE, dan CC. Gambar 3.17 merupakan gambar diagram alir proses evaluasi dengan penjelasan sebagai berikut:

1. Memasukkan nilai curah hujan uji aktual dan prediksi dalam bentuk normalisasi dan denormalisasi, jumlah baris data curah hujan uji, nilai curah hujan terendah dan tertinggi, serta *range* minimal dan maksimal dari data curah hujan ternormalisasi.
2. Melakukan denormalisasi terhadap curah hujan uji aktual dan curah hujan uji prediksi.
3. Menghitung nilai evaluasi MAE menggunakan Persamaan (2.20), nilai RMSE dengan Persamaan (2.21), nilai CC dengan Persamaan (2.22), dan nilai MAPE dengan Persamaan (2.23) serta Persamaan (2.24).
4. Memasukkan nilai-nilai evaluasi ke dalam *array* hasilEvaluasi.
5. Menghasilkan keluaran berupa hasilEvaluasi yang dapat digunakan untuk mengetahui performa model peramalan yang diterapkan.




**Gambar 3.17 Diagram alir evaluasi**

## 3.3 Perhitungan Manual Proses Peramalan Curah Hujan dengan RBF yang Dioptimasi Menggunakan HPSOGA

Pada bagian ini akan dijelaskan perhitungan manual proses peramalan curah hujan dengan RBF yang dioptimasi menggunakan HPSOGA yang meliputi perhitungan proses normalisasi data curah hujan, tabulasi data *input* dan *output* jaringan RBF, optimasi RBF dengan HPSOGA, peramalan curah hujan dengan RBF, serta evaluasi hasil peramalan curah hujan dengan RBF-HPSOGA.

### 3.3.1 Normalisasi data

Pertama-tama proses yang dilakukan adalah normalisasi data curah hujan. Tabel 3.1 merupakan tabel data curah hujan dasarian daerah Karangploso Kabupaten Malang yang didapatkan dari Stasiun Klimatologi Karangploso Malang. Untuk detail data curah hujan terdapat pada Lampiran.

**Tabel 3.1 Data curah hujan dasarian tahun 2009-2014 daerah Karangploso Kabupaten Malang**

| | Januari | | | Februari | | | | November | | | Desember | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Tahun | 1 | 2 | 3 | 1 | 2 | 3 | ... | 1 | 2 | 3 | 1 | 2 | 3 |
| 2009 | 89 | 58 | 112 | 120 | 36 | 279 | ... | 0 | 121 | 79 | 67 | 6 | 151 |
| 2010 | 123 | 59 | 165 | 73 | 110 | 36 | ... | 295 | 16 | 155 | 190 | 50 | 21 |
| 2011 | 35 | 20 | 84 | 99 | 23 | 60 | ... | 151 | 52 | 73 | 77 | 56 | 136 |
| 2012 | 145 | 80 | 62 | 191 | 152 | 79 | ... | 10 | 102 | 38 | 217 | 109 | 156 |
| 2013 | 102 | 90 | 174 | 42 | 112 | 60 | ... | 32 | 75 | 63 | 212 | 152 | 57 |
| 2014 | 89 | 55 | 161 | 38 | 121 | 35 | ... | 60 | 46 | 35 | 109 | 160 | 69 |

Proses perhitungan normalisasi dilakukan dengan Persamaan (2.18) sehingga nilai curah hujan yang ada pada Tabel 3.1 menjadi ternormalisasi seperti pada Tabel 3.2. Nilai minimum dari atribut numerik curah hujan dasarian ($min_{cHujan}$) adalah 0 mm, sedangkan nilai maksimumnya ($min_{cHujan}$) adalah 295 mm. Normalisasi data dilakukan agar data berada di rentang -10 dan 10, oleh karena itu *range* minimum dari nilai hasil normalisasi ($new\_min_{cHujan}$) adalah -10 dan *range* maksimumnya ($new\_max_{cHujan}$) adalah 10.

**Tabel 3.2 Data curah hujan dasarian tahun 2009-2014 daerah Karangploso Kabupaten Malang ternormalisasi**

| | Jan | | | Feb | | | | Nov | | | Des | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Tahun | 1 | 2 | 3 | 1 | 2 | 3 | ... | 1 | 2 | 3 | 1 | 2 | 3 |
| 2009 | -3.97 | -6.07 | -2.41 | -1.86 | -7.6 | 8.92 | ... | -10 | -1.8 | -4.6 | -5.46 | -9.59 | 0.237 |
| 2010 | -1.66 | -6 | 1.19 | -5.05 | -2.5 | -7.6 | ... | 10 | -8.92 | 0.51 | 2.881 | -6.61 | -8.58 |
| 2011 | -7.63 | -8.64 | -4.31 | -3.29 | -8.4 | -5.9 | ... | 0.24 | -6.47 | -5.1 | -4.78 | -6.2 | -0.78 |
| 2012 | -0.17 | -4.58 | -5.8 | 2.95 | 0.31 | -4.6 | ... | -9.3 | -3.08 | -7.4 | 4.712 | -2.61 | 0.576 |

**Tabel 3.2 Data curah hujan dasarian tahun 2009-2014 daerah Karangploso Kabupaten Malang ternormalisasi (lanjutan)**

| | Jan | | | Feb | | | | Nov | | | Des | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Tahun | 1 | 2 | 3 | 1 | 2 | 3 | ... | 1 | 2 | 3 | 1 | 2 | 3 |
| 2013 | -3.08 | -3.9 | 1.8 | -7.15 | -2.4 | -5.9 | ... | -7.8 | -4.92 | -5.7 | 4.373 | 0.305 | -6.14 |
| 2014 | -3.97 | -6.27 | 0.92 | -7.42 | -1.8 | -7.6 | ... | -5.9 | -6.88 | -7.6 | -2.61 | 0.847 | -5.32 |

Nilai pada kolom-kolom di atas diperoleh dari contoh perhitungan berikut:

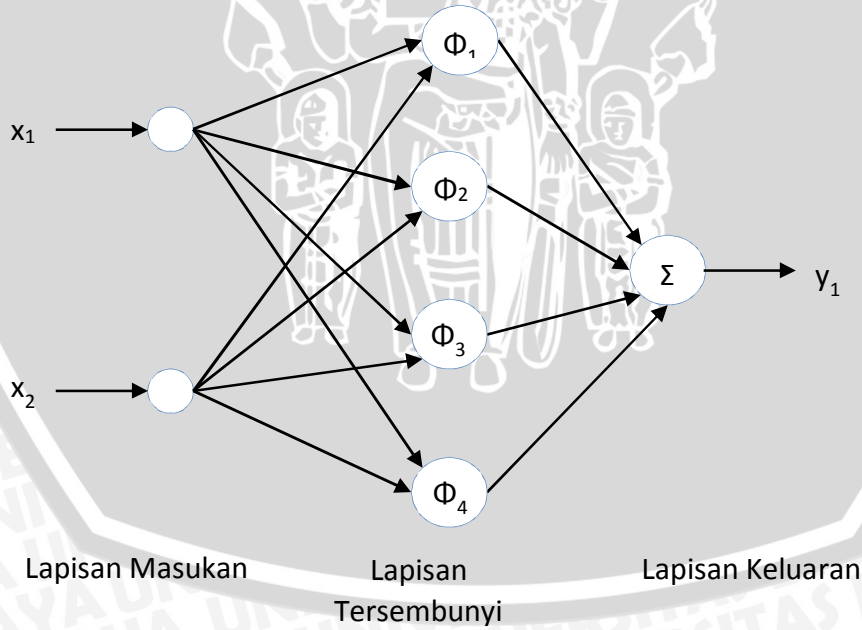
- Curah hujan ternormalisasi Bulan Januari Dasarian 1 Tahun 2009

$$= \frac{V_i - min_{cHujan}}{max_{cHujan} - min_{cHujan}} \left(new\_max_{cHujan} - new\_min_{cHujan}\right) + new\_min_{cHujan}$$

$$= \frac{89 - 0}{295 - 0} (10 - (-10)) + (-10) = -3.9661 = -3.97$$

### 3.3.2 Tabulasi data *input* dan *output*

Langkah selanjutnya setelah mendapatkan data curah hujan ternormalisasi adalah mentabulasi data sebagai *node* masukan dan keluaran jaringan syaraf tiruan RBF untuk data latih. Pada perhitungan manual ini dibatasi jumlah data latih yang digunakan sebanyak 5 saja. Arsitektur jaringan syaraf tiruan yang digunakan dalam perhitungan manual ini terdiri dari 3 lapisan yaitu 2 *node* masukan (*input nodes*), 4 *node* tersembunyi (*hidden nodes*), dan 1 *node* keluaran (*output nodes*) seperti yang digambarkan pada Gambar 3.18.




**Gambar 3.18 Arsitektur jaringan syaraf tiruan RBF pada perhitungan manual**

Karena telah ditetapkan jumlah *node* masukannya adalah 2, maka untuk mendapatkan hasil prediksi peramalan curah hujan pada satu dasarian tertentu akan digunakan data masukan dari data curah hujan 2 dasarian sebelumnya. Sebagai contoh apabila ingin mendapatkan hasil prediksi peramalan curah hujan pada bulan Januari dasarian 3 tahun 2009 (data ini sebagai $y_1$), maka data masukan yang dimasukkan dalam jaringan RBF adalah data curah hujan pada bulan Januari dasarian 1 tahun 2009 sebagai $x_1$ dan bulan Januari dasarian 2 tahun 2009 sebagai $x_2$ (seperti yang dapat dilihat pada data latih 1 pada Tabel 3.3). Data curah hujan kemudian ditabulasi sesuai dengan jumlah *node* masukan dan keluarannya sehingga menghasilkan data latih seperti pada Tabel 3.3.

**Tabel 3.3 Data latih jaringan RBF pada perhitungan manual**

| No. | $x_1$ | $x_2$ | $y_1$ |
|---|---|---|---|
| 1 | -3.9661 | -6.0678 | -2.40678 |
| 2 | -6.0678 | -2.40678 | -1.86441 |
| 3 | -2.4068 | -1.86441 | -7.55932 |
| 4 | -1.8644 | -7.55932 | 8.915254 |
| 5 | -7.5593 | 8.915254 | -6.47458 |

### 3.3.3 Optimasi RBF dengan HPSOGA

Dalam menyelesaikan permasalahan pencarian parameter RBF optimal untuk peramalan curah hujan menggunakan HPSOGA, diperlukan beberapa tahapan yaitu inisialisasi parameter, membangkitkan populasi awal, menghitung *fitness*, melakukan metode GA pada kromosom terbaik, serta melakukan metode PSO. Pada perhitungan manual ini, inisialisasi parameter awal adalah sebagai berikut:

a. Jumlah populasi = 8
b. Jumlah iterasi = 1
c. Probabilitas tukar silang (*pc*) = 0.8
d. Probabilitas mutasi (*pm*) = 0.2
e. $\delta$ tukar silang = 0
f. $\delta$ mutasi = 0.8
g. $\gamma_1$ dan $\gamma_2$ = 2

#### 3.3.3.1 Pembangkitan populasi awal

Pada tahap ini, populasi awal dibangkitkan sebanyak jumlah populasi yang telah ditentukan. Seperti yang telah dijelaskan pada bab sebelumnya, kromosom di penelitian ini direpresentasikan dengan pengkodean biner (untuk menunjukkan koneksi antara *node* masukan dengan *node* tersembunyi di jaringan RBF) dan nilai real (untuk nilai titik pusat vektor masukan, radius, dan bobot dari keluaran). Satu komponen jaringan RBF akan dianggap sebagai satu gen penyusun kromosom individu.

Panjang gen yang dikodekan dalam biner serta nilai real dapat dihitung dengan Persamaan (2.7) dan (2.8), sedangkan untuk panjang kromosom dapat dihitung dengan Persamaan (2.9). Diketahui jumlah *node* masukan (*n*) jaringan RBF pada perhitungan manual sebanyak 2, *node* tersembunyinya (*N*) sebanyak 4, dan *node* keluarannya (*m*) sebanyak 1.

- Panjang gen biner (*G*) = *n* x *N*
  = 2 x 4 = 8
- Panjang gen real (*H*) = (*N* x *n*) + *N* + (*N* x *m*)
  = (4 x 2) + 4 + (4 x 1) = 8 + 4 + 4 = 16
- Panjang kromosom (*L*) = Panjang gen biner (*G*) + panjang gen real (*H*)
  = 16 + 8 = 34

Untuk tiap gen yang dikodekan dalam biner akan dibangkitkan nilai random 0 atau 1 dan yang dikodekan dalam nilai real akan dibangkitkan nilai random pada interval -10 sampai 10. Hasil pembangkitan populasi awal ini ditunjukkan pada Tabel 3.4 dan 3.5.

**Tabel 3.4 Representasi kromosom pengkodean biner pada populasi awal**

| Kromosom | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| $P_1$ | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| $P_2$ | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| $P_3$ | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| $P_4$ | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| $P_5$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| $P_6$ | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| $P_7$ | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| $P_8$ | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |

**Tabel 3.5 Representasi kromosom pengkodean nilai real pada populasi awal**

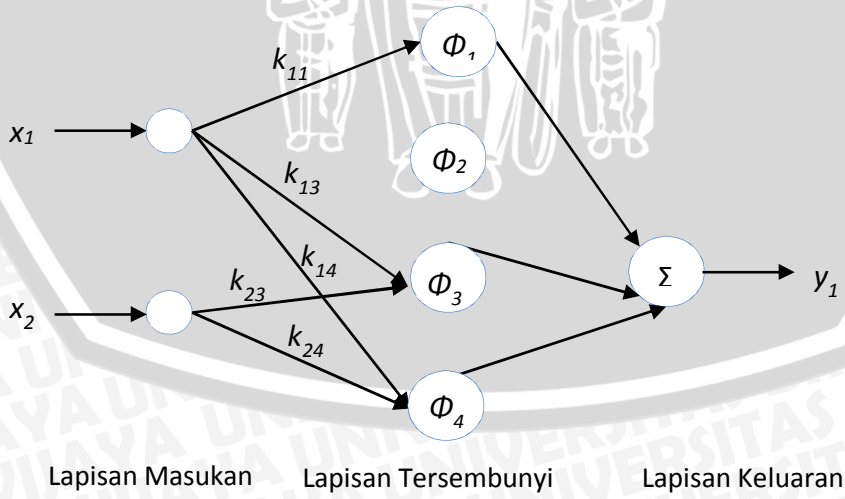
| Krom | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $P_1$ | -2.7 | -3.8 | 5.54 | -9.8 | 1.47 | -5 | -0.7 | 1.71 | -4.46 | -2.3 | -7.7 | 3.63 | 8.87 | -0.6 | -8.4 | -7.7 |
| $P_2$ | 4.29 | 1.77 | -8.6 | -2.5 | -0.6 | 4.89 | 5.94 | 8.07 | 0.525 | -0.8 | -3.5 | 0.55 | -1.2 | -1.4 | -3.6 | -2.7 |
| $P_3$ | -5.8 | 3.07 | -6.8 | -1.9 | 9.06 | -8.2 | -8.6 | 3.27 | 2.775 | -3.1 | 9.63 | -7.7 | 2.76 | 2.36 | -7.5 | -8.3 |
| $P_4$ | 7.85 | -9.4 | -1 | -1.7 | -9 | -7 | 4.07 | 9.35 | -7.72 | 3.21 | 7.61 | -1.8 | -8.9 | 8.59 | -5.5 | -5.7 |
| $P_5$ | 6.89 | 9.02 | 2.37 | 6.16 | -1.9 | -7.7 | 6.66 | -3.5 | 8.783 | 1.02 | 4.46 | 4.05 | -2.3 | 0.73 | -2.5 | -9 |
| $P_6$ | -2.4 | 6.12 | 6.58 | 1.9 | 2.78 | 9.1 | -7.3 | 4.56 | -0.21 | 1.41 | -7.2 | -7.5 | -5.3 | -2.2 | -1.5 | -8.4 |
| $P_7$ | 5.63 | -8.9 | 3.36 | 1.48 | 7.76 | 8.68 | 5.83 | -7.6 | -6.62 | 3.34 | 3.84 | 6.02 | 4.09 | 5.42 | 7.32 | -5.7 |
| $P_8$ | -8.5 | -9 | 0.98 | 3.66 | -8.4 | 2.67 | -6.1 | -4 | 2.38 | -9.9 | 1.23 | 5.18 | 7.87 | -2.1 | -0.2 | 4.17 |

### 3.3.3.2 Perhitungan *fitness* dan perangkingan

Setelah menginisialisasi populasi awal, kromosom digunakan untuk melakukan peramalan curah hujan dengan metode RBF dan dihitung nilai *fitness*-nya. Berdasarkan Tabel 3.4 dan 3.5, salah satu solusi yang dibangkitkan pada populasi awal tersebut adalah $P_1$ dengan nilai sebagai berikut:

- Koneksi dari *node* masukan ke-1 ke *node* tersembunyi ke-1 ($k_{11}$)= 1
- Koneksi dari *node* masukan ke-1 ke *node* tersembunyi ke-2 ($k_{12}$) = 0
- Koneksi dari *node* masukan ke-1 ke *node* tersembunyi ke-3 ($k_{13}$) = 1
- Koneksi dari *node* masukan ke-1 ke *node* tersembunyi ke-4 ($k_{14}$) = 1
- Koneksi dari *node* masukan ke-2 ke *node* tersembunyi ke-1 ($k_{21}$) = 0
- Koneksi dari *node* masukan ke-2 ke *node* tersembunyi ke-2 ($k_{22}$) = 0
- Koneksi dari *node* masukan ke-2 ke *node* tersembunyi ke-3 ($k_{23}$) = 1
- Koneksi dari *node* masukan ke-2 ke *node* tersembunyi ke-4 ($k_{24}$) = 1
- Vektor titik pusat *node* tersembunyi ke-1 ($cn_1$) = (-2.739260972, 1.474673996)
- Vektor titik pusat *node* tersembunyi ke-2 ($cn_2$) = (-3.794211005, -4.99744697)
- Vektor titik pusat *node* tersembunyi ke-3 ($cn_3$) = (5.541334979, -0.681585529)
- Vektor titik pusat *node* tersembunyi ke-4 ($cn_4$) = (-9.788657763, 1.714348932)
- Radius *node* tersembunyi ke-1 ($r_1$) = -4.458976999
- Radius *node* tersembunyi ke-2 ($r_2$) = -2.345431483
- Radius *node* tersembunyi ke-3 ($r_3$) = -7.681216014
- Radius *node* tersembunyi ke-4 ($r_4$) = 3.631821277
- Bobot *node* tersembunyi ke-1 ke *node* keluaran ($\omega_{11}$) = 8.868858543
- Bobot *node* tersembunyi ke-2 ke *node* keluaran ($\omega_{12}$) = -0.579209056
- Bobot *node* tersembunyi ke-3 ke *node* keluaran ($\omega_{13}$) = -8.399558593
- Bobot *node* tersembunyi ke-4 ke *node* keluaran ($\omega_{14}$) = -7.683082918

Dari penjabaran di atas, maka dapat digambarkan arsitektur jaringan RBF solusi $P_1$ seperti Gambar 3.19.




**Gambar 3.19 Arsitektur jaringan syaraf tiruan untuk solusi $P_1$**

Parameter-parameter yang telah diinisialisasi sebelumnya kemudian digunakan untuk melakukan peramalan curah hujan dengan RBF pada data latih. Dalam metode RBF, pertama-tama yang dihitung adalah fungsi *radial basis* dengan menggunakan fungsi *Gaussian* dalam Persamaan (2.1). Contoh berikut adalah fungsi *radial basis* untuk solusi $P_1$ dan data latih ke-1, dimana vektor *node* masukan data latih ke-1 (*x*) adalah (-3.9661, -6.0678).

- $\Phi_1(x, c_1) = exp\left(-\frac{\|x - cn_1\|_2}{{r_1}^2}\right)$

  $= exp\left(-\frac{\sqrt{|-3.9661-(-2.739260972)|^2}}{(-4.458976999)^2}\right)$

  = 0.940160546

- $\Phi_2(x, c_2)$ = 0 karena tidak ada koneksi sama sekali dari *node* masukan ke *node* tersembunyi ke-2

- $\Phi_3(x, c_3) = exp\left(-\frac{\|x - cn_3\|_2}{{r_3}^2}\right)$

  $= exp\left(-\frac{\sqrt{|-3.9661-5.541334979|^2 + |-6.0678-(-0.681585529)|^2}}{(-7.681216014)^2}\right)$

  = 0.830936

- $\Phi_4(x, c_4) = exp\left(-\frac{\|x - cn_4\|_2}{{r_4}^2}\right)$

  $= exp\left(-\frac{\sqrt{|-3.9661-(-9.788657763)|^2 + |-6.0678-1.714348932|^2}}{(3.631821277)^2}\right)$

  = 0.478615116

Langkah selanjutnya adalah menghitung nilai keluaran (prediksi) dari jaringan RBF dengan menggunakan Persamaan (2.3). Contoh berikut merupakan perhitungan nilai keluaran (prediksi) untuk solusi $P_1$ dan data latih ke-1.

$y_{prediksi} = \sum_{i=1}^{N} \omega_{ti}\, \Phi_i(x, cn_i)$

$= \omega_{11}\Phi_1(x, cn_1) + \omega_{12}\Phi_2(x, cn_2) + \omega_{13}\Phi_1(x, cn_3) + \omega_{14}\Phi_4(x, cn_4)$

= (8.868858543 x 0.940160546) + 0 + (-8.399558593 x 0.830936) +
(-7.683082918 x 0.478615116)

= -2.318584 = -2.319

Perhitungan fungsi *radial basis* dan nilai keluaran (prediksi) menggunakan jaringan RBF untuk solusi $P_1$ ini dilakukan pada semua data latih sehingga mendapatkan nilai seperti pada Tabel 3.6.

**Tabel 3.6 Perhitungan fungsi *radial basis* dan nilai keluaran jaringan RBF untuk solusi $P_1$**

| No. | $x_1$ | $x_2$ | $y_{aktual}$ | $\Phi_1(x, cn_1)$ | $\Phi_2(x, cn_2)$ | $\Phi_3(x, cn_3)$ | $\Phi_4(x, cn_4)$ | $y_{prediksi}$ |
|---|---|---|---|---|---|---|---|---|
| 1 | -3.966 | -6.068 | -2.407 | 0.94016 | 0 | 0.83094 | 0.47862 | -2.319 |
| 2 | -6.068 | -2.407 | -1.864 | 0.84585 | 0 | 0.81961 | 0.65642 | -4.426 |
| 3 | -2.407 | -1.864 | -7.559 | 0.98342 | 0 | 0.87267 | 0.5369 | -2.733 |
| 4 | -1.864 | -7.559 | 8.9153 | 0.95695 | 0 | 0.84257 | 0.39661 | -1.637 |
| 5 | -7.559 | 8.9153 | -6.475 | 0.78472 | 0 | 0.75939 | 0.56468 | -3.757 |

Setelah mendapatkan nilai keluaran (prediksi) dari jaringan RBF, kemudian hitung nilai *error* peramalan curah hujan dan nilai *fitness* dengan menggunakan Persamaan (2.10) dan Persamaan (2.11). Contoh berikut merupakan perhitungan nilai *fitness* untuk solusi $P_1$.

$$Error\ (E_i) = \frac{1}{m}\sum_{k=1}^{m}\left|e_i(x_j) - y_j\right|$$

$$= \frac{1}{5}(|-2.319-(-2.407)| + |-4.426-(-1.864)| + |-2.733-(-7.559)| + |-1.637-8.9153| + |-3.757-(-6.475)|)$$

$$= 20.74558683 \times \frac{1}{5} = 4.149117$$

$$Fitness\ (f) = \frac{1}{(1+E_i)} = \frac{1}{(1+4.149117)} = 0.194208$$

Peramalan curah hujan dengan menggunakan metode RBF, perhitungan *error* hasil peramalan, dan perhitungan *fitness* ini dilakukan pada seluruh solusi (kromosom) dalam populasi awal sehingga mendapatkan nilai *error* dan *fitness* untuk tiap kromosom seperti pada Tabel 3.7.

**Tabel 3.7 Perhitungan nilai *error* dan *fitness* untuk seluruh solusi pada populasi awal**

| *Parent* | *Error* | *Fitness* |
|---|---|---|
| 1 | 4.149117 | 0.194208 |
| 2 | 4.130202 | 0.194924 |
| 3 | 10.12005 | 0.089928 |
| 4 | 5.049194 | 0.165311 |
| 5 | 7.551399 | 0.11694 |
| 6 | 7.376705 | 0.119379 |

**Tabel 3.7 Perhitungan nilai *error* dan *fitness* untuk seluruh solusi pada populasi awal (lanjutan)**

| *Parent* | *Error* | *Fitness* |
|---|---|---|
| 7 | 11.71762 | 0.078631 |
| 8 | 5.041054 | 0.165534 |

Langkah berikutnya setelah mendapatkan nilai *fitness* dari semua kromosom di populasi awal adalah mengurutkan kromosom berdasarkan nilai *fitness* terbaik serta memperbarui indeks kromosom untuk memudahkan proses selanjutnya yaitu pengoptimalan parameter jaringan RBF. Hasil proses perangkingan dan perbaruan indeks kromosom ditunjukkan pada Tabel 3.8.

**Tabel 3.8 Perangkingan kromosom pada populasi awal**

| Krom | Asal Krom | *Fitness* |
|---|---|---|
| $P_1$ | $P_2$ | 0.194924 |
| $P_2$ | $P_1$ | 0.194208 |
| $P_3$ | $P_8$ | 0.165534 |
| $P_4$ | $P_4$ | 0.165311 |
| $P_5$ | $P_6$ | 0.119379 |
| $P_6$ | $P_5$ | 0.11694 |
| $P_7$ | $P_3$ | 0.089928 |
| $P_8$ | $P_7$ | 0.078631 |

Seperti yang telah dijelaskan pada bab sebelumnya bahwa proses pengoptimalan parameter jaringan RBF akan dilakukan dengan metode PSO dan GA. Dari 4N kromosom yang tersedia, gunakan operator GA untuk 2N kromosom dengan nilai *fitness* tertinggi (elite) dan gunakan operator PSO untuk 2N kromosom elite sebagai kecepatan partikel serta 2N kromosom dengan nilai *fitness* terendah sebagai posisi partikel. 2N kromosom disini adalah sejumlah setengah dari populasi awal yang ditentukan (4N). Apabila populasi awal yaitu 8 kromosom dianggap sebagai 4N, maka 2N berarti 8/2 = 4 kromosom. Pengaturan kromosom yang akan digunakan untuk operator GA dan PSO dapat ditunjukkan dalam Tabel 3.9.

**Tabel 3.9 Pengaturan kromosom untuk optimasi parameter jaringan RBF**

| Operator | Kromosom |
|---|---|
| GA | $P_1$ sampai $P_4$ |
| PSO – kecepatan partikel | $P_1$ sampai $P_4$ |
| PSO – posisi partikel | $P_5$ sampai $P_8$ |

### 3.3.3.3 Tukar silang

Jumlah *offspring* hasil proses tukar silang dalam perhitungan manual ini dihitung dengan rumus *pc* x populasi metode GA = 0,8 x 4 = 3 *offspring*. Setiap kali proses tukar silang dilakukan akan dihasilkan 2 *offspring,* maka dari itu harus dilakukan proses tukar silang sebanyak 2 kali untuk menghasilkan 3 offspring. Pada tahap ini terlebih dulu dipilih dua kromosom secara acak sebagai *parent*, yaitu $P_4$ dan $P_3$.

Nilai gen yang dikodekan dalam biner pada masing-masing *offspring* didapatkan dengan metode tukar silang *one-point* pada titik ke-4. Hasil dari perhitungan tukar silang pertama untuk gen yang dikodekan dalam biner ditunjukkan pada Tabel 3.10.

**Tabel 3.10 Hasil tukar silang pertama pengkodean biner**

| Kromosom | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| *Parent* 4 ($P_4$) | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| *Parent* 3 ($P_3$) | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| *Offspring* 1 ($O_1$) | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| *Offspring* 2 ($O_2$) | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |

Sedangkan untuk nilai gen yang dikodekan dalam nilai real pada masing-masing *offspring* dihitung dengan Persamaan (2.12). Dalam perhitungannya dibutuhkan nilai $c_i$ yang merupakan nilai acak yang dibangkitkan pada rentang [-$\alpha$, 1+ $\alpha$] dimana $\alpha$ tukar silang adalah 0 sehingga rentang nilai acak menjadi [0,1]. Nilai acak untuk masing-masing gen ditunjukkan pada Tabel 3.11 dan hasil dari perhitungan tukar silang ditunjukkan pada Tabel 3.12.

**Tabel 3.11 Pembangkitan nilai acak (*c*)**

| $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ | $c_7$ | $c_8$ | $c_9$ | $c_{10}$ | $c_{11}$ | $c_{12}$ | $c_{13}$ | $c_{14}$ | $c_{15}$ | $c_{16}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.96 | 0.59 | 0.47 | 0.69 | 0.37 | 0.41 | 0.4 | 0.56 | 0.56 | 0.37 | 0.98 | 0.02 | 0.89 | 0.58 | 0.89 | 0.95 |

**Tabel 3.12 Hasil tukar silang pertama pengkodean nilai real**

| Krom | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $P_4$ | 7.85 | -9.4 | -1 | -1.7 | -9 | -7 | 4.07 | 9.35 | -7.72 | 3.21 | 7.61 | -1.8 | -8.9 | 8.59 | -5.5 | -5.7 |
| $P_3$ | -8.5 | -9 | 0.98 | 3.66 | -8.4 | 2.67 | -6.1 | -4 | 2.38 | -9.9 | 1.23 | 5.18 | 7.87 | -2.1 | -0.2 | 4.17 |
| $O_1$ | -7.8 | -9.1 | -0.1 | 2.02 | -8.7 | -3 | 0.03 | 1.88 | -2.1 | -1.7 | 1.35 | -1.6 | 6.06 | 2.42 | -0.8 | 3.68 |
| $O_2$ | 7.2 | -9.2 | 0.03 | -0.1 | -8.6 | -1.3 | -2.1 | 3.42 | -3.3 | -5.1 | 7.48 | 5.06 | -7.1 | 4.08 | -4.9 | -5.2 |

Hasil perhitungan tukar silang tersebut dapat dicontohkan sebagai berikut :

- $cn_{11}$ untuk *offspring* 1 ($O_1$) = 7.85 + 0.96 (-8.5 – 7.85) = -7.8
- $cn_{11}$ untuk *offspring* 2 ($O_2$) = -8.5 + 0.96 (7.85 – (-8.5)) = 7.2

Cara yang sama dilakukan pada proses *crossover* yang kedua. Misalnya yang terpilih sebagai *parent* adalah $P_1$ dan $P_3$, sehingga diperoleh nilai *offspring* yang ditunjukkan pada Tabel 3.13 dan 3.14.

**Tabel 3.13 Hasil tukar silang kedua pengkodean biner**

| Kromosom | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| *Parent* 1 ($P_1$) | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| *Parent* 3 ($P_3$) | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| *Offspring* 3 ($O_3$) | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |

**Tabel 3.14 Hasil tukar silang kedua pengkodean nilai real**

| Krom | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $P_1$ | 4.29 | 1.77 | -8.6 | -2.5 | -0.6 | 4.89 | 5.94 | 8.07 | 0.525 | -0.8 | -3.5 | 0.55 | -1.2 | -1.4 | -3.6 | -2.7 |
| $P_3$ | -8.5 | -9 | 0.98 | 3.66 | -8.4 | 2.67 | -6.1 | -4 | 2.38 | -9.9 | 1.23 | 5.18 | 7.87 | -2.1 | -0.2 | 4.17 |
| $O_3$ | -7.9 | -4.5 | -4.1 | 1.78 | -3.5 | 3.97 | 1.15 | 1.32 | 1.56 | -4.2 | 1.13 | 0.63 | 6.89 | -1.8 | -0.5 | 3.83 |

### 3.3.3.4 Mutasi

Jumlah *offspring* hasil proses mutasi dalam perhitungan manual ini dihitung dengan rumus *pm* x populasi metode GA = 0,2 x 4 = 1 *offspring*. Setiap kali proses mutasi berlangsung, akan dihasilkan 1 *offspring*. Pada tahap ini terlebih dulu dipilih satu kromosom secara acak sebagai *parent* , yaitu $P_1$.

Nilai gen yang dikodekan dalam biner pada masing-masing *offspring* didapatkan dengan metode mutasi *bitwise*. Sebelumnya dilakukan pemilihan gen secara acak dan terpilih metode *bitwise* dilakukan pada gen pertama kromosom $P_1$. Hasil dari perhitungan mutasi untuk gen yang dikodekan dalam biner ditunjukkan pada Tabel 3.15.

**Tabel 3.15 Hasil mutasi pengkodean biner**

| Kromosom | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| *Parent* 1 ($P_1$) | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| *Offspring* 4 ($O_4$) | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |

Sedangkan untuk nilai gen yang dikodekan dalam nilai real pada masing-masing *offspring* dihitung dengan Persamaan (2.13). Dalam perhitungannya

dibutuhkan nilai $c_i$ yang merupakan nilai acak yang dibangkitkan pada rentang [0 – $\delta$ – nilai gen sebelumnya, 1 + $\delta$ + nilai gen sebelumnya]. Apabila dalam perhitungannya nilai gen hasil mutasi lebih besar dari rentang maksimal nilai real kromosom maka hasil mutasi akan diisi dengan rentang maksimal tersebut yaitu 10. Begitu juga sebaliknya apabila kurang dari rentang minimal nilai real kromosom maka hasil mutasi akan diisi dengan -10. Nilai acak untuk masing-masing gen ditunjukkan pada Tabel 3.16 dan hasil dari perhitungan tukar silang ditunjukkan pada Tabel 3.17.

**Tabel 3.16 Pembangkitan nilai acak (c)**

| $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ | $c_7$ | $c_8$ | $c_9$ | $c_{10}$ | $c_{11}$ | $c_{12}$ | $c_{13}$ | $c_{14}$ | $c_{15}$ | $c_{16}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -4.3 | -1.43 | 3.68 | 0.41 | 0.13 | 0.33 | -6.16 | -5.33 | 0.27 | 0.33 | -1.24 | 1.64 | 0.55 | 0.5 | -1.03 | 0.75 |

**Tabel 3.17 Hasil mutasi pengkodean nilai real**

| Krom | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $P_1$ | 4.29 | 1.77 | -8.6 | -2.5 | -0.6 | 4.89 | 5.94 | 8.07 | 0.525 | -0.8 | -3.5 | 0.55 | -1.2 | -1.4 | -3.6 | -2.7 |
| $O_4$ | -0.1 | 0.34 | -4.95 | -2.1 | -0.5 | 5.22 | -0.2 | 2.74 | 0.8 | -0.47 | -4.7 | 2.18 | -0.6 | -0.88 | -4.7 | -1.9 |

Hasil perhitungan rentang nilai acak dan proses mutasi tersebut dapat dicontohkan sebagai berikut :

- rentang nilai acak minimal $cn_{11}$ pada *offspring* 4 = 0 – 0.8 – 4.2874 = -5.0874
- rentang nilai acak maksimal $cn_{11}$ pada *offspring* 4 = 1 + 0.8 + 4.2874 = 6.0874
- nilai acak (*c*) untuk $cn_{11}$ pada *offspring* 4 = -4.34895
- $cn_{11}$ untuk *offspring* 4 ($O_4$) = 4.287379 + (-4.34895) = -0.1

### 3.3.3.5 Perbarui partikel *personal best* (*Pbest*) dan *global best* (*Pgest*)

Pada bab sebelumnya dijelaskan bahwa gunakan operator PSO untuk 2N kromosom elite sebagai kecepatan partikel serta 2N kromosom dengan nilai *fitness* terendah sebagai posisi partikel. 2N kromosom disini adalah sejumlah setengah dari populasi awal yang ditentukan (4N). Apabila populasi awal yaitu 8 kromosom dianggap sebagai 4N, maka 2N berarti 8/2 = 4 kromosom. Sehingga dalam perhitungan manual ini 4 kromosom dengan *fitness* terbaik (kromosom *elite*) akan digunakan sebagai kecepatan partikel dan 4 kromosom dengan *fitness* terendah akan digunakan sebagai posisi partikel. Kecepatan dan posisi partikel pada perhitungan manual untuk yang dikodekan dalam biner maupun nilai real ditunjukkan pada Tabel 3.18 sampai Tabel 3.19.

**Tabel 3.18 Kecepatan awal partikel pengkodean biner**

| Partikel | Krom | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| $O_5$ | $P_1$ | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| $O_6$ | $P_2$ | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| $O_7$ | $P_3$ | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| $O_8$ | $P_4$ | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |

**Tabel 3.19 Kecepatan awal partikel pengkodean nilai real**

| Par-tikel | Krom | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $O_5$ | $P_1$ | 4.29 | 1.77 | -8.6 | -2.5 | -0.6 | 4.89 | 5.94 | 8.07 | 0.53 | -0.8 | -3.5 | 0.55 | -1.2 | -1.4 | -3.6 | -2.7 |
| $O_6$ | $P_2$ | -2.7 | -3.8 | 5.54 | -9.8 | 1.47 | -5 | -0.7 | 1.71 | -4.5 | -2.3 | -7.7 | 3.63 | 8.87 | -0.6 | -8.4 | -7.7 |
| $O_7$ | $P_3$ | -8.5 | -9 | 0.98 | 3.66 | -8.4 | 2.67 | -6.1 | -4 | 2.38 | -9.9 | 1.23 | 5.18 | 7.87 | -2.1 | -0.2 | 4.17 |
| $O_8$ | $P_4$ | 7.85 | -9.4 | -1 | -1.7 | -9 | -7 | 4.07 | 9.35 | -7.7 | 3.21 | 7.61 | -1.8 | -8.9 | 8.59 | -5.5 | -5.7 |

**Tabel 3.20 Posisi awal partikel pengkodean biner**

| Partikel | Krom | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| $O_5$ | $P_5$ | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| $O_6$ | $P_6$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| $O_7$ | $P_7$ | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| $O_8$ | $P_8$ | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |

**Tabel 3.21 Posisi awal partikel pengkodean nilai real**

| Par-tikel | Krom | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $O_5$ | $P_5$ | -2.4 | 6.12 | 6.58 | 1.9 | 2.78 | 9.1 | -7.3 | 4.56 | -0.2 | 1.41 | -7.2 | -7.5 | -5.3 | -2.2 | -1.5 | -8.4 |
| $O_6$ | $P_6$ | 6.89 | 9.02 | 2.37 | 6.16 | -1.9 | -7.7 | 6.66 | -3.5 | 8.78 | 1.02 | 4.46 | 4.05 | -2.3 | 0.73 | -2.5 | -9 |
| $O_7$ | $P_7$ | -5.8 | 3.07 | -6.8 | -1.9 | 9.06 | -8.2 | -8.6 | 3.27 | 2.78 | -3.1 | 9.63 | -7.7 | 2.76 | 2.36 | -7.5 | -8.3 |
| $O_8$ | $P_8$ | 5.63 | -8.9 | 3.36 | 1.48 | 7.76 | 8.68 | 5.83 | -7.6 | -6.6 | 3.34 | 3.84 | 6.02 | 4.09 | 5.42 | 7.32 | -5.7 |

Kemudian gunakan 2N kromosom yang memiliki *fitness* terburuk untuk memperbarui partikel *personal best* (*Pbest*). Syaratnya adalah apabila nilai *fitness* dari partikel *Pbest* lebih kecil daripada nilai *fitness* dari kromosom non elite maka perbarui partikel *Pbest* dengan kromosom yang memiliki *fitness* terburuk secara terurut. Jika syarat tersebut tidak terpenuhi maka nilai dari partikel *Pbest* akan tetap seperti sebelumnya. Sedangkan untuk partikel *global best* (*Pgest*) akan selalu diperbarui dengan kromosom $P_1$ karena $P_1$ akan selalu memiliki nilai *fitness*

tertinggi apabila dibandingkan dengan kromosom lainnya. Karena dalam perhitungan manual ini masih pada iterasi pertama, maka semua *Pbest* akan diperbarui dengan 2N kromosom yang memiliki *fitness* terburuk secara terurut seperti pada perhitungan berikut:

- $Pbest_1$ diambil dari kromosom $P_5$ dengan *fitness* = 0.119379
- $Pbest_2$ diambil dari kromosom $P_6$ dengan *fitness* = 0.116939
- $Pbest_3$ diambil dari kromosom $P_7$ dengan *fitness* = 0.089928
- $Pbest_4$ diambil dari kromosom $P_8$ dengan *fitness* = 0.078631
- *Pgest* diambil dari kromosom $P_1$ dengan *fitness* = 0.194924

### 3.3.3.6 Perbarui kecepatan dan posisi partikel

Di langkah selanjutnya, kecepatan partikel dan posisinya diperbarui untuk mendapatkan *offspring* baru. Rentang kecepatan baru partikel baik untuk pengkodean biner maupun real ditetapkan sesuai rumus $[-V_{max}, V_{max}]$ dimana $V_{max}$ adalah 60% dari rentang maksimum awal partikel yaitu -10 untuk nilai real dan 1 untuk nilai biner. Maka dari itu rentang kecepatan baru partikel adalah [-0.6, 0.6] untuk yang dikodekan dalam biner dan [-6, 6] untuk nilai real. Apabila dalam perhitungan kecepatan baru partikel mendapatkan nilai yang lebih besar dari rentang maksimal maka nilainya akan diganti dengan rentang maksimal tersebut. Sebaliknya apabila kecepatan baru partikel nilainya lebih kecil dari rentang minimal maka nilainya akan diganti dengan rentang minimal tersebut. Hasil kecepatan partikel baru untuk yang dikodekan dalam biner dan nilai real ini ditunjukkan pada Tabel 3.22 dan 3.23.

**Tabel 3.22 Kecepatan baru partikel pengkodean biner**

| Partikel | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| $O_5$ | -0.6 | -0.6 | 0.51 | 0.51 | 0 | 0.6 | -0.6 | 0.6 |
| $O_6$ | 0.51 | -0.6 | 0.6 | 0.6 | -0.6 | 0.6 | 0.51 | 0.51 |
| $O_7$ | -0.6 | -0.6 | 0.6 | 0 | 0.51 | 0.51 | -0.6 | 0.6 |
| $O_8$ | -0.6 | -0.6 | 0.51 | 0.6 | 0 | 0.51 | 0 | 0.6 |

**Tabel 3.23 Kecepatan baru partikel pengkodean nilai real**

| Par-tikel | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $O_5$ | 6 | -6 | -6 | -6 | -5.9 | -4.4 | 6 | 6 | 1.47 | -4 | 4.3 | 6 | 6 | 0.6 | -5.3 | 6 |
| $O_6$ | -5.7 | -6 | -6 | -6 | 2.93 | 6 | -1.5 | 6 | -6 | -4.2 | -6 | -3.9 | 6 | -3.8 | -6 | 6 |
| $O_7$ | 6 | -6 | -2.5 | 0.86 | -6 | 6 | 6 | 5.8 | -2.5 | -1.4 | -6 | 6 | -2.4 | -6 | 6 | 6 |
| $O_8$ | 1.83 | 6 | -6 | -6 | -6 | -6 | 2.3 | 6 | 6 | -5.2 | -6 | -6 | -6 | -6 | -6 | 2.1 |

Dimana telah ditetapkan nilai acak untuk mencari bobot inertia adalah 0.028455 serta nilai acak pertama dan kedua untuk mencari kecepatan baru

partikel adalah 0.727138 dan 0.821957. Kecepatan baru partikel baik yang dikodekan dalam biner maupun nilai real dihitung menggunakan Persamaan (2.15) dengan hasil sebagai berikut:

- Bobot inersia ($\varpi$) = 0.5 + (rand / 2) = 0.5 + (0.028455 / 2) = 0.514227
- Kecepatan baru $k_{11}$ pada partikel $O_5$ ($V_{k11}^{New}$)
  $= \varpi \,.\, V_{k11}^{Old} + \gamma_1 \,.rand\,.\, (P_{best1} - x_{k11}^{Old}) + \gamma_2 \,.rand\,.\, (P_{gest} - x_{k11}^{Old})$
  = 0.514227 x 0 + 2 x 0.727138 x (1 - 1) + 2 x 0.821957 x (0 - 1) = -1.64 karena lebih dari rentang minimal nilainya menjadi -0.6
- Kecepatan baru $cn_{11}$ pada partikel $O_5$ ($V_{cn11}^{New}$)
  $= \varpi \,.\, V_{cn11}^{Old} + \gamma_1 \,.rand\,.\, (P_{best1} - x_{cn11}^{Old}) + \gamma_2 \,.rand\,.\, (P_{gest} - x_{cn11}^{Old})$
  = 0.514227 x 4.287379 + 2 x 0.727138 x (-2.37135 – (-2.37135)) + 2 x 0.821957 x (4.28737– (-2.37135))
  = 13.15107 karena lebih dari rentang maksimal nilainya menjadi 6
- Kecepatan baru $r_1$ pada partikel $O_5$ ($V_{r1}^{New}$)
  $= \varpi \,.\, V_{r1}^{Old} + \gamma_1 \,.rand\,.\, (P_{best1} - x_{r1}^{Old}) + \gamma_2 \,.rand\,.\, (P_{gest} - x_{r1}^{Old})$
  = 0.514227 x 0.525302 + 2 x 0.727138 x (-0.20968– (-0.20968)) + 2 x 0.821957 x (-0.79647– (-0.20968))
  = 1.48
- Kecepatan baru $\omega_{11}$ pada partikel $O_5$ ($V_{\omega 11}^{New}$)
  $= \varpi \,.\, V_{\omega 11}^{Old} + \gamma_1 \,.rand\,.\, (P_{best1} - x_{\omega 11}^{Old}) + \gamma_2 \,.rand\,.\, (P_{gest} - x_{\omega 11}^{Old})$
  = 0.514227 x (-1.18616) + 2 x 0.727138 x (-5.34436 – (-5.34436)) + 2 x 0.821957 x (-1.18616 – (-5.34436))
  = 6.23 karena lebih dari rentang maksimal nilainya menjadi 6

Setelah mendapatkan kecepatan partikel yang baru, maka posisi partikel juga diperbarui. Apabila dalam perhitungannya nilai posisi baru lebih besar dari rentang maksimal nilai real kromosom, maka posisi partikel yang dikodekan dengan nilai real akan diperbarui dengan rentang maksimal tersebut yaitu 10. Begitu juga sebaliknya apabila kurang dari rentang minimal nilai real kromosom maka posisi diperbarui menjadi -10. Pada perhitungan posisi baru pengkodean biner juga dibangkitkan nilai acak pada rentang 0 sampai 1 yang digunakan pada Persamaan (2.17) ditunjukkan pada Tabel 3.24. Hasil posisi partikel baru ini ditunjukkan pada Tabel 3.25 dan 3.26.

**Tabel 3.24 Nilai acak untuk perbaruan posisi partikel pengkodean biner**

| Partikel | Nilai Acak | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| $O_5$ | 0.21 | 0.27 | 0.7 | 0.92 | 0.07 | 0.507 | 0.39 | 0.55 |
| $O_6$ | 0.68 | 0.77 | 0.34 | 0.19 | 0.05 | 0.173 | 0.07 | 0.85 |
| $O_7$ | 0.29 | 0.66 | 0.03 | 0.99 | 0.78 | 0.217 | 0.22 | 0.76 |
| $O_8$ | 0.87 | 0.52 | 0.58 | 0.07 | 0.32 | 0.551 | 0.84 | 0.37 |

**Tabel 3.25 Posisi baru partikel pengkodean biner**

| Partikel | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| $O_5$ | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| $O_6$ | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| $O_7$ | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| $O_8$ | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |

**Tabel 3.26 Posisi baru partikel pengkodean nilai real**

| Par-tikel | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $O_5$ | 3.63 | 0.12 | 0.58 | -4.1 | -3.1 | 4.69 | -1.3 | 10 | 1.27 | -2.6 | -2.9 | -1.5 | 0.66 | -1.6 | -6.8 | -2.4 |
| $O_6$ | 1.2 | 3.02 | -3.6 | 0.16 | 1 | -1.7 | 5.12 | 2.47 | 2.78 | -3.2 | -1.5 | 0.16 | 3.66 | -3 | -8.5 | -3 |
| $O_7$ | 0.22 | -2.9 | -9.3 | -1 | 3.06 | -2.2 | -2.6 | 9.08 | 0.3 | -4.4 | 3.63 | -1.7 | 0.32 | -3.6 | -1.5 | -2.3 |
| $O_8$ | 7.46 | -2.9 | -2.6 | -4.5 | 1.76 | 2.68 | 8.1 | -1.6 | -0.6 | -1.8 | -2.2 | 0.02 | -1.9 | -0.6 | 1.32 | -3.6 |

Posisi baru partikel yang dikodekan dalam biner dapat diperbarui dengan Persamaan (2.17) dan yang dikodekan dalam nilai real diperbarui dengan Persamaan (2.16). Hasil perhitungan posisi baru tersebut adalah sebagai berikut :

- Fungsi *sigmoid* biner untuk posisi baru $k_{11}$ pada partikel $O_5$
  $= \frac{1}{1+\exp(-V_{k11}^{New})} = \frac{1}{1+\exp(-(-0.6))} = 0.35$
- Posisi baru $k_{11}$ pada partikel $O_5$ ($X_{k11}^{New}$)
  Nilai acak (0.21) < nilai fungsi *sigmoid* biner untuk $k_{11}$ (0.35), maka $X_{k11}^{New} = 1$
- Posisi baru $cn_{11}$ pada partikel $O_5$ ($X_{cn11}^{New}$)
  $= X_{cn11}^{Old} + V_{cn11}^{New}$
  = -2.37135 + 6 = 3.63
- Posisi baru $r_1$ pada partikel $O_5$ ($X_{r1}^{New}$)
  $= X_{r1}^{Old} + V_{r1}^{New}$
  = -0.20968 + 1.48 = 1.27
- Posisi baru $\omega_{11}$ pada partikel $O_5$ ($X_{\omega 11}^{New}$)
  $= X_{\omega 11}^{Old} + V_{\omega 11}^{New}$
  = -5.34436 + 6 = 0.66

### 3.3.3.7 Seleksi

Pada proses seleksi ini digunakan metode seleksi elitism. Pertama-tama kumpulan individu baik *parent* maupun *offspring* dari metode GA dan PSO akan dihitung *fitness*-nya seperti yang telah dijelaskan pada subbab sebelumnya. Kemudian dipilih individu yang memiliki nilai *fitness* terbesar sebanyak jumlah

populasi awal, sehingga hasil seleksi akan menghasilkan 8 individu terbaik sebagai populasi baru. Hasil dari proses ini ditunjukkan pada Tabel 3.27.

**Tabel 3.27 Hasil proses seleksi**

| Kromosom | Asal | *Fitness* |
|---|---|---|
| 1 | $P_1$ | 0.194924117 |
| 2 | $P_2$ | 0.194208042 |
| 3 | $O_4$ | 0.17681287 |
| 4 | $O_7$ | 0.173682672 |
| 5 | $O_2$ | 0.170819024 |
| 6 | C5 | 0.1706266 |
| 7 | C6 | 0.167072003 |
| 8 | P3 | 0.165534036 |

### 3.3.4 Peramalan curah hujan dengan RBF

Proses selanjutnya adalah melakukan pemilihan individu terbaik hasil seleksi. Individu terbaik adalah individu dengan nilai *fitness* tertinggi yang akan digunakan sebagai parameter metode RBF untuk peramalan curah hujan. Dari Tabel 3.27 disimpulkan bahwa parameter RBF optimum untuk peramalan curah hujan dihasilkan dari individu dengan nilai *fitness* sebesar 0.194924. Hasil solusi terbaik pada iterasi pertama ini ditunjukkan pada Tabel 3.28 dan 3.29.

**Tabel 3.28 Individu terbaik pengkodean biner hasil proses seleksi**

| Partikel | Koneksi *Node* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{21}$ | $k_{22}$ | $k_{23}$ | $k_{24}$ |
| $P_1$ | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |

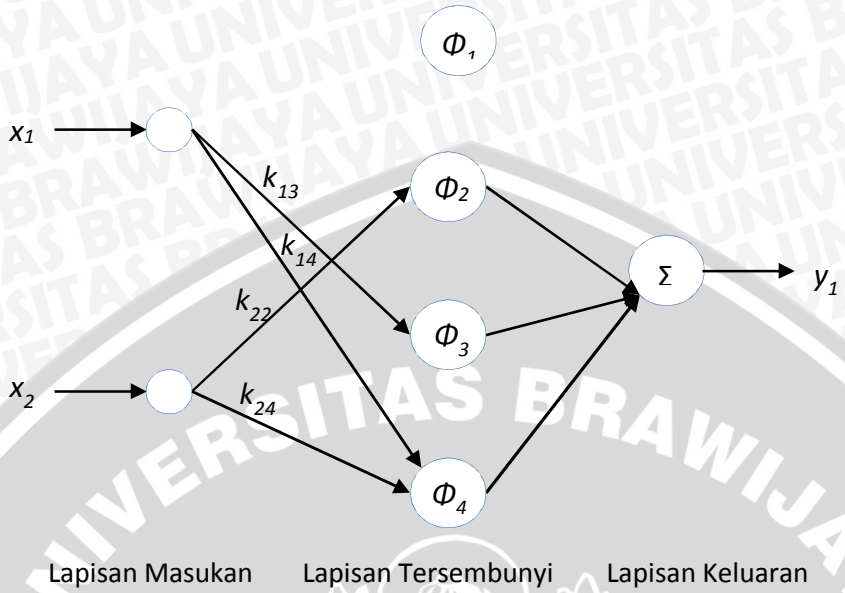
**Tabel 3.29 Individu terbaik pengkodean real hasil proses seleksi**

| Par-tikel | Titik Pusat | | | | | | | | *Radius* | | | | Bobot | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $cn_{11}$ | $cn_{21}$ | $cn_{31}$ | $cn_{41}$ | $cn_{12}$ | $cn_{22}$ | $cn_{32}$ | $cn_{42}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $\omega_{11}$ | $\omega_{12}$ | $\omega_{13}$ | $\omega_{14}$ |
| $P_1$ | 4.29 | 1.77 | -8.6 | -2.5 | -0.6 | 4.89 | 5.94 | 8.07 | 0.525 | -0.8 | -3.5 | 0.55 | -1.2 | -1.4 | -3.6 | -2.7 |

Dari Tabel 3.28 dapat dilihat koneksi dari *node* masukan ke *node* tersembunyi paling optimal yang dapat digambarkan arsitektur jaringan RBF nya seperti Gambar 3.20. Pada individu terbaik parameter $k_{11}$, $k_{13}$, $k_{14}$, $k_{23}$, dan $k_{24}$ bernilai 1 yang berarti dalam jaringan RBF paling optimal terdapat koneksi dari:

- *node* masukan ke-1 ke *node* tersembunyi ke-1 ($k_{11}$)
- *node* masukan ke-1 ke *node* tersembunyi ke-3 ($k_{13}$)
- *node* masukan ke-1 ke *node* tersembunyi ke-4 ($k_{14}$)
- *node* masukan ke-2 ke *node* tersembunyi ke-3 ($k_{23}$)
- *node* masukan ke-2 ke *node* tersembunyi ke-4 ($k_{24}$)

Sedangkan selain dari pasangan *node-node* tersebut pada jaringan RBF tidak terkoneksi karena parameternya bernilai 0 ($k_{12}$, $k_{21}$, dan $k_2$).




**Gambar 3.20 Arsitektur jaringan syaraf tiruan RBF untuk individu terbaik**

Parameter-parameter pada Tabel 3.28 dan 3.29 kemudian digunakan untuk melakukan peramalan curah hujan dengan RBF pada data uji. Pada perhitungan manual ini dibatasi jumlah data uji yang digunakan sebanyak 5 saja. Data uji peramalan curah hujan yang telah dinormalisasi dan ditabulasi menjadi data *input* dan *output* jaringan RBF dapat dilihat pada Tabel 3.30.

**Tabel 3.30 Data uji jaringan RBF pada perhitungan manual**

| No. | $x_1$ | $x_2$ | $y_1$ |
|---|---|---|---|
| 1 | -3.966 | -6.271 | 0.915 |
| 2 | -6.271 | 0.915 | -7.424 |
| 3 | 0.915 | -7.424 | -1.797 |
| 4 | -7.424 | -1.797 | -7.627 |
| 5 | -1.797 | -7.627 | -4.441 |

Proses peramalan dengan menggunakan metode RBF meliputi perhitungan fungsi *radial basis* pada tiap *node* tersembunyi dengan menggunakan Persamaan (2.1) dan perhitungan nilai keluaran (prediksi) dari jaringan RBF dengan menggunakan Persamaan (2.3) sehingga mendapatkan nilai seperti pada Tabel 3.31.

**Tabel 3.31 Perhitungan fungsi *radial basis* dan nilai keluaran jaringan RBF untuk individu terbaik**

| No. | $x_1$ | $x_2$ | $y_{aktual}$ | $\Phi_1(x, cn_1)$ | $\Phi_2(x, cn_2)$ | $\Phi_3(x, cn_3)$ | $\Phi_4(x, cn_4)$ | $y_{prediksi}$ |
|---|---|---|---|---|---|---|---|---|
| 1 | -3.966 | -6.271 | 0.915 | 0 | 2.2843E-08 | 0.67845 | 8.52E-22 | -2.458 |
| 2 | -6.271 | 0.915 | -7.424 | 0 | 0.00189976 | 0.822091 | 1.56E-12 | -2.982 |
| 3 | 0.915 | -7.424 | -1.797 | 0 | 3.7129E-09 | 0.451753 | 6.31E-24 | -1.637 |
| 4 | -7.424 | -1.797 | -7.627 | 0 | 2.6431E-05 | 0.904943 | 7.89E-17 | -3.279 |
| 5 | -1.797 | -7.627 | -4.441 | 0 | 2.6944E-09 | 0.566267 | 1.07E-23 | -2.052 |

Di bawah ini adalah perhitungan fungsi *radial basis* dan nilai keluaran (prediksi) untuk individu terbaik dan data uji ke-1, dimana vektor *node* masukan data uji ke-1 (*x*) adalah (-3.966, -6.271).

- $\Phi_1(x, c_1)$ = 0 karena tidak ada koneksi sama sekali dari *node* masukan ke *node* tersembunyi ke-1
- $\Phi_2(x, c_2) = exp\left(-\frac{\|x - cn_2\|_2}{r_2{}^2}\right) = exp\left(-\frac{\sqrt{|-6.271-(-4.89)|^2}}{(-0.8)^2}\right)$

  $= 2.2843 \times 10^{-8}$
- $\Phi_3(x, c_3) = exp\left(-\frac{\|x - cn_3\|_2}{r_3{}^2}\right) = exp\left(-\frac{\sqrt{|-3.966-(-8.6)|^2}}{(-3.5)^2}\right)$

  $= 0.67845$
- $\Phi_4(x, c_4) = exp\left(-\frac{\|x - cn_4\|_2}{r_4{}^2}\right)$

  $= exp\left(-\frac{\sqrt{|-3.966-(-2.5)|^2 + |-6.271-8.07|^2}}{(0.55)^2}\right)$

  $= 8.52 \times 10^{-22}$
- $y_{prediksi} = \sum_{i=1}^{N} \omega_{ti}\, \Phi_i(x, cn_i)$

  $= \omega_{11}\Phi_1(x, cn_1) + \omega_{12}\Phi_2(x, cn_2) + \omega_{13}\Phi_1(x, cn_3) + \omega_{14}\Phi_4(x, cn_4)$

  $= 0 + (-1.4 \times (2.2 \times 10^{-8})) + (-3.6 \times 0.6745) + (-2.7 \times (8.52 \times 10^{-22}))$

  $= -2.458$

### 3.3.5 Evaluasi Hasil Peramalan Curah Hujan dengan RBF-HPSOGA

Proses berikutnya setelah mendapatkan hasil peramalan curah hujan menggunakan metode RBF adalah melakukan proses denormalisasi data dan

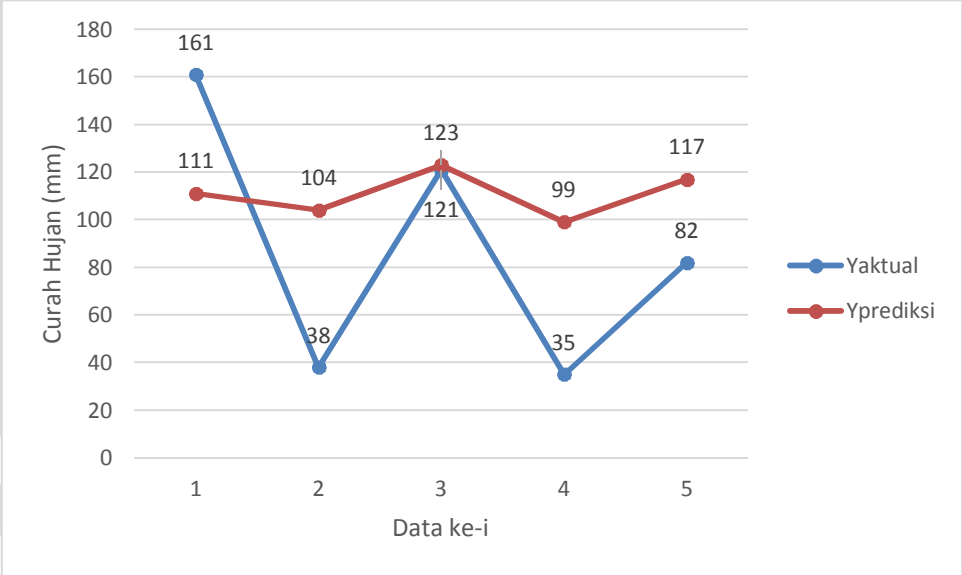
perhitungan nilai evaluasi untuk mengetahui performa model peramalan yang diterapkan.

### 3.3.5.1 Denormalisasi data

Denormalisasi data dilakukan pada data curah hujan uji aktual maupun curah hujan hasil prediksi dengan menggunakan Persamaan (2.19). Hasil data curah hujan uji aktual maupun curah hujan hasil prediksi setelah proses denormalisasi dapat dilihat pada Gambar 3.21 serta Tabel 3.32.

**Tabel 3.32 Data curah hujan uji aktual dan prediksi hasil proses denormalisasi**

| No. | $y_{aktual}$ | $y_{prediksi}$ |
|---|---|---|
| 1 | 161 | 111 |
| 2 | 38 | 104 |
| 3 | 121 | 123 |
| 4 | 35 | 99 |
| 5 | 82 | 117 |




**Gambar 3.21 Data curah hujan uji aktual dan prediksi hasil proses denormalisasi**

Nilai pada kolom-kolom di Tabel 3.32 diperoleh dari contoh perhitungan berikut:

- Data curah hujan uji prediksi ke-1 hasil denormalisasi

$$V_i = \frac{(V'_i - new\_min_A)(max_A - min_A)}{new\_max_A - new\_min_A} + min_A$$

$$= \frac{(-2.458 - (-10)) * (295 - 0)}{(10 - (-10))} + 0 = 111.2389 = 111$$

### 3.3.5.2 Evaluasi

Setelah mendapatkan data curah hujan uji aktual dan prediksi yang telah melalui proses denormalisasi, perhitungan nilai evaluasi dapat dilakukan. Untuk menghitung nilai evaluasi MAE dapat menggunakan Persamaan (2.20), nilai RMSE dengan Persamaan (2.21), nilai CC dengan Persamaan (2.22), dan nilai MAPE dengan Persamaan (2.23).

$$MAE = \frac{1}{5}\sum_{i=1}^{5}|Xm_i - Xa_i|$$

$$= \frac{1}{5}(50 + 66 + 2 + 64 + 35)$$

$$= \frac{1}{5}(217) = 43.4$$

$$RMSE = \sqrt{\frac{1}{5}\sum_{t=1}^{5}(Y_t^s - Y_t^a)^2}$$

$$= \sqrt{\frac{1}{5}(2500 + 4356 + 4 + 4096 + 1225)}$$

$$= \sqrt{\frac{1}{5}(12181)} = 49.35788$$

$$CC = \frac{N \sum XY - \sum X \sum Y}{\sqrt{[N \sum X^2 - (\sum X)^2][N \sum Y^2 - (\sum Y)^2]}}$$

$$= \frac{(5\ x\ 49765) - (437\ x\ 554)}{\sqrt{[(5\ x\ 49955) - (437)^2][(5\ x\ 61756) - (554)^2]}}$$

$$= \frac{6727}{\sqrt{109614384}} = \frac{6727}{10469.68882} = 0.642521484$$

$$MAPE = \frac{1}{5}\sum_{i=1}^{5} Error_i$$

$$= \frac{1}{5}(31.0559 + 100 + 1.652893 + 100 + 42.68293)$$

$$= \frac{1}{5} \times 275.3917 = 55.078$$

Dari perhitungan sebelumnya, dapat disimpulkan bahwa performa model peramalan dalam perhitungan manual ini untuk nilai evaluasi *Mean Absolute Error* (MAE) adalah sebesar 43.4, nilai *Root Mean Square Error* (RMSE) sebesar 49.35788, nilai *Correlation Coefficient* (CC) sebesar 0.642521484, dan nilai *Mean Absolute Percentage Error* (MAPE) sebesar 55.078.

Melihat nilai MAE dan RMSE yang didapat bisa dikatakan performa model peramalan masih belum baik karena tingkat *error* yang masih termasuk besar apabila dibandingkan dengan rentang data curah hujan yang berkisar dari 0 sampai 295 serta dari jumlah data uji yang hanya sebanyak 5 data, sehingga perlu dilakukan perhitungan untuk iterasi selanjutnya apabila ingin untuk memperbaiki performanya. Untuk nilai CC sudah cukup baik karena 0.642521484 cukup dekat dengan nilai 1.00 yang menunjukkan hubungan antar variabel dalam model peramalan cukup kuat dengan sifat hubungan yang langsung (berbanding lurus). Sedangkan untuk nilai MAPE masih belum terlalu baik karena nilainya masih tinggi dan menyebabkan keakurasian sistem menjadi rendah yaitu 45% (dihitung dari 100 – nilai MAPE).

## 3.4 Perancangan antarmuka

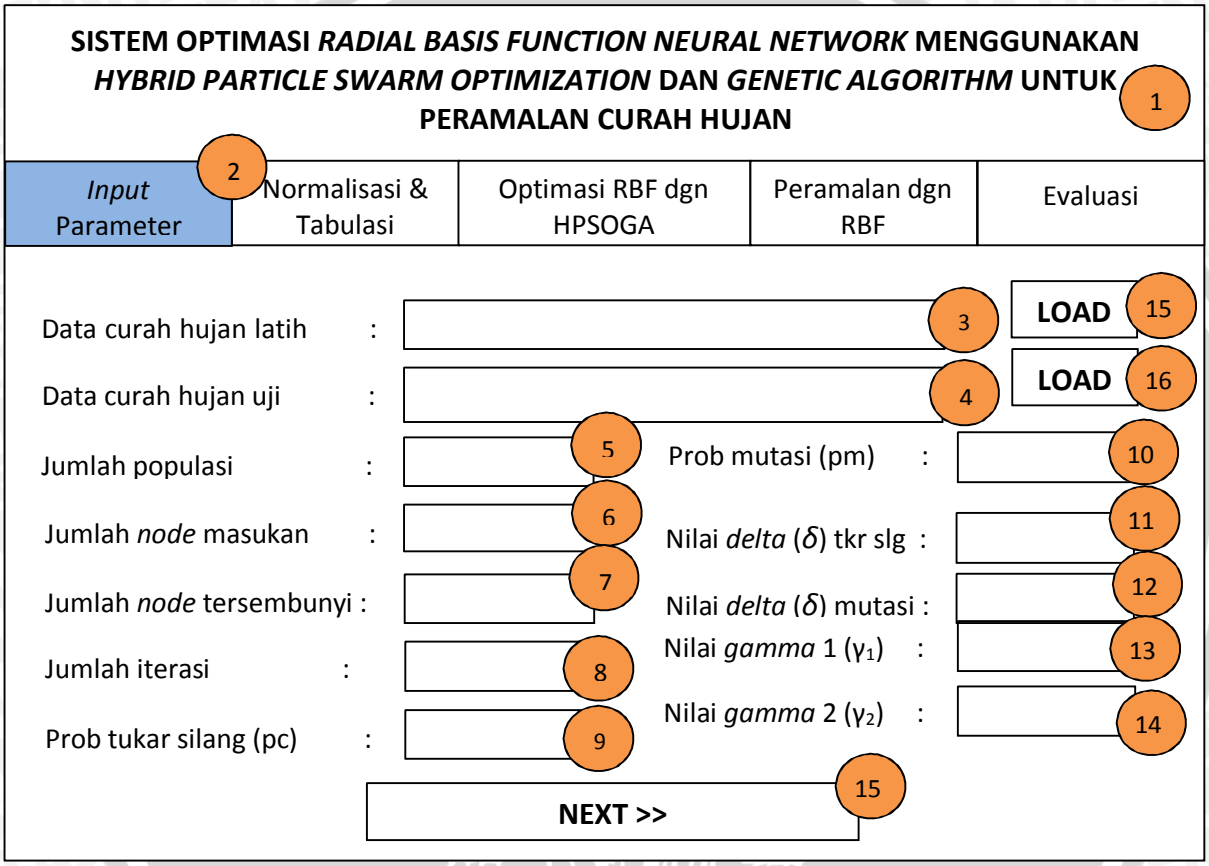
Pada perancangan antarmuka dari sistem optimasi *Radial Basis Function Neural Network* menggunakan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan ini akan dibagi menjadi beberapa halaman utama yaitu halaman *input* parameter, halaman normalisasi dan tabulasi, halaman optimasi RBF dengan HPSOGA, halaman peramalan dengan RBF, dan halaman evaluasi. Rancangan halaman tersebut akan dijelaskan pada sub-bab 3.4.1 sampai dengan sub-bab 3.4.5.

### 3.4.1 Halaman *input* parameter

Halaman *input* parameter merupakan sebuah halaman untuk memasukkan semua parameter yang akan digunakan dalam sistem optimasi peramalan curah hujan ini. Rancangan halaman *input* parameter sistem ditunjukkan pada Gambar 3.22 dengan keterangan sebagai berikut:

1. Judul sistem
2. *Tab control* untuk melakukan *input* parameter.
3. *Text box* untuk menampilkan nama *file* data curah hujan latih terpilih.
4. *Text box* untuk menampilkan nama *file* data curah hujan uji terpilih.
5. *Text box* untuk memasukkan jumlah populasi.
6. *Text box* untuk memasukkan *node* masukan.
7. *Text box* untuk memasukkan *node* tersembunyi.
8. *Text box* untuk memasukkan jumlah iterasi.
9. *Text box* untuk memasukkan probabilitas tukar silang (pc).
10. *Text box* untuk memasukkan probabilitas mutasi (pm).
11. *Text box* untuk memasukkan nilai *delta* ($\delta$) tukar silang.

12. *Text box* untuk memasukkan nilai *delta* ($\delta$) mutasi.
13. *Text box* untuk memasukkan nilai *learning rate* 1 / *gamma* 1 ($\gamma_1$).
14. *Text box* untuk memasukkan nilai *learning rate* 2 / *gamma* 2 ($\gamma_2$).
15. *Button* NEXT untuk melanjutkan perhitungan ke proses normalisasi data serta tabulasi data *input* dan *output* jaringan RBF.
16. *Button* LOAD untuk menampilkan dialog pembukaan *file* yang berfungsi untuk memilih *file* data curah hujan yang digunakan sebagai data latih.
17. *Button* LOAD untuk menampilkan dialog pembukaan *file* yang berfungsi untuk memilih *file* data curah hujan yang digunakan sebagai data uji.



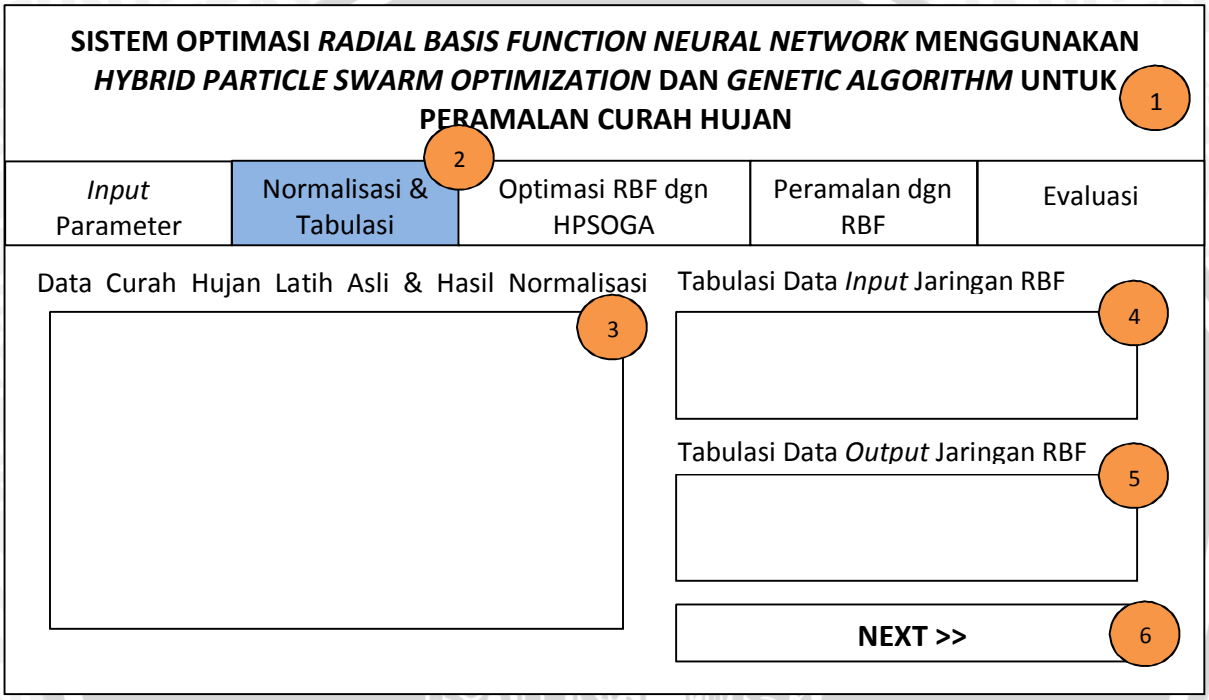

**Gambar 3.22 Rancangan antarmuka halaman i*nput* parameter**

### 3.4.2 Halaman normalisasi dan tabulasi

Halaman normalisasi dan tabulasi merupakan sebuah halaman untuk menampilkan hasil perhitungan proses normalisasi data serta tabulasi data *input* dan *output* jaringan RBF pada sistem optimasi peramalan curah hujan. Rancangan halaman normalisasi dan tabulasi ditunjukkan pada Gambar 3.23 dengan keterangan sebagai berikut:

1. Judul sistem

2. *Tab control* untuk menampilkan proses normalisasi dan tabulasi data.
3. *Data grid view* untuk menampilkan data curah hujan latih asli dan hasil normalisasi dalam bentuk tabel.
4. *Data grid view* untuk menampilkan tabulasi data *input* jaringan RBF dalam bentuk tabel.
5. *Data grid view* untuk menampilkan tabulasi data *output* jaringan RBF dalam bentuk tabel.
6. *Button* NEXT untuk melanjutkan perhitungan ke proses optimasi RBF dengan menggunakan metode HPSOGA.



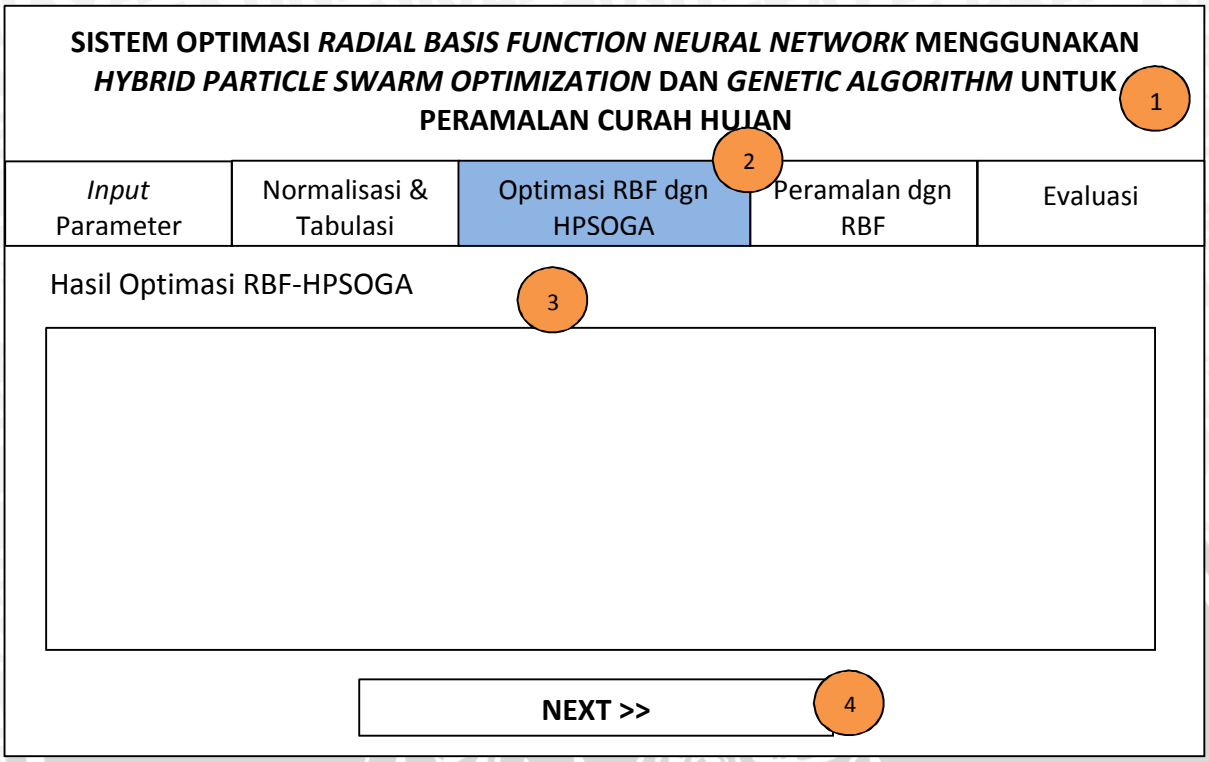

**Gambar 3.23 Rancangan antarmuka halaman normalisasi dan tabulasi**

### 3.4.3 Halaman optimasi RBF dengan HPSOGA

Halaman optimasi RBF dengan HPSOGA merupakan sebuah halaman untuk menampilkan hasil perhitungan proses optimasi RBF dengan metode HPSOGA yang meliputi proses pembangkitan populasi, perhitungan *fitness* dan perangkingan, tukar silang, mutasi, pembaruan kecepatan dan posisi partikel, serta seleksi. Rancangan halaman optimasi RBF dengan HPSOGA ditunjukkan pada Gambar 3.24 dengan keterangan sebagai berikut:

1. Judul sistem
2. *Tab control* untuk menampilkan proses optimasi RBF dengan HPSOGA.
3. *Rich text box* untuk menampilkan kromosom terbaik hasil optimasi RBF dengan HPSOGA untuk tiap iterasi.

4. *Button* NEXT untuk melanjutkan perhitungan ke peramalan curah hujan dengan menggunakan metode RBF.



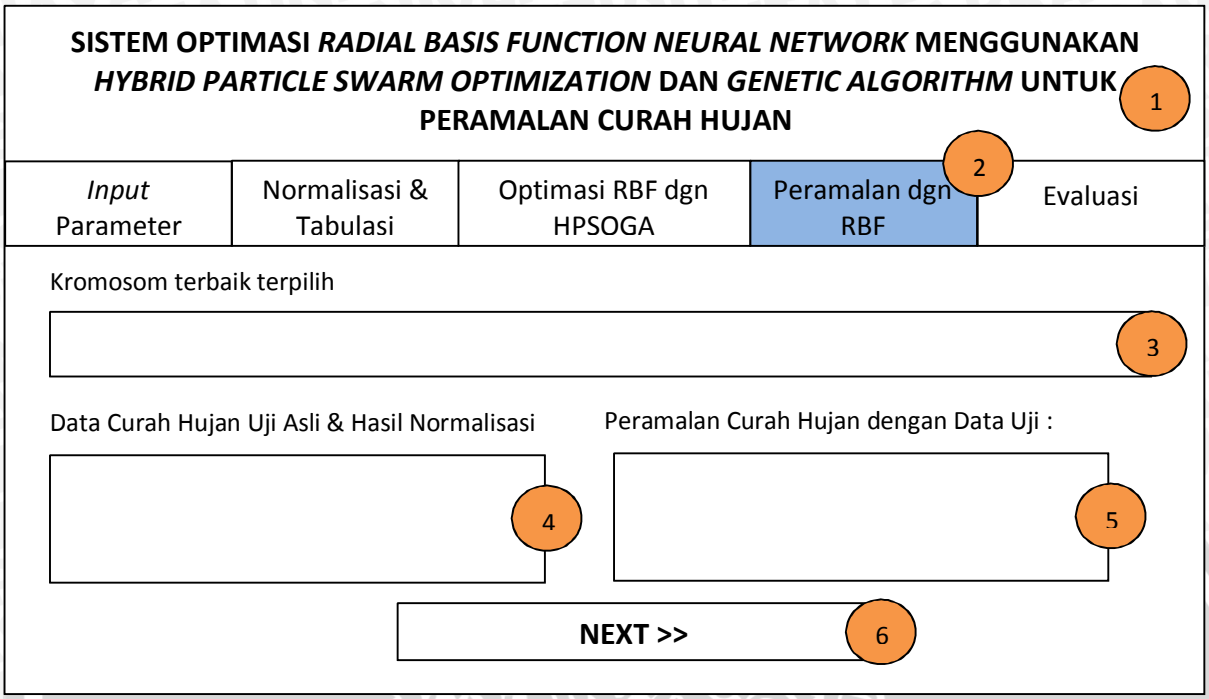

**Gambar 3.24 Rancangan antarmuka halaman optimasi RBF dengan HPSOGA**

### 3.4.4 Halaman peramalan dengan RBF

Halaman peramalan dengan RBF merupakan sebuah halaman untuk menampilkan hasil perhitungan peramalan curah hujan dengan RBF menggunakan parameter yang telah dioptimasi pada langkah sebelumnya. Rancangan halaman peramalan curah hujan dengan RBF ditunjukkan pada Gambar 3.25 dengan keterangan sebagai berikut:

1. Judul sistem
2. *Tab control* untuk menampilkan proses peramalan curah hujan dengan menggunakan RBF.
3. *Data grid view* untuk menampilkan kromosom terbaik terpilih yang akan digunakan sebagai parameter metose RBF dalam peramalan curah hujan.
4. *Data grid view* untuk menampilkan data curah hujan uji asli dan hasil normalisasi dalam bentuk tabel.
5. *Data grid view* untuk menampilkan hasil proses peramalan curah hujan menggunakan RBF dalam bentuk tabel.

6. *Button* NEXT untuk melanjutkan perhitungan ke proses evaluasi sistem optimasi peramalan curah hujan.

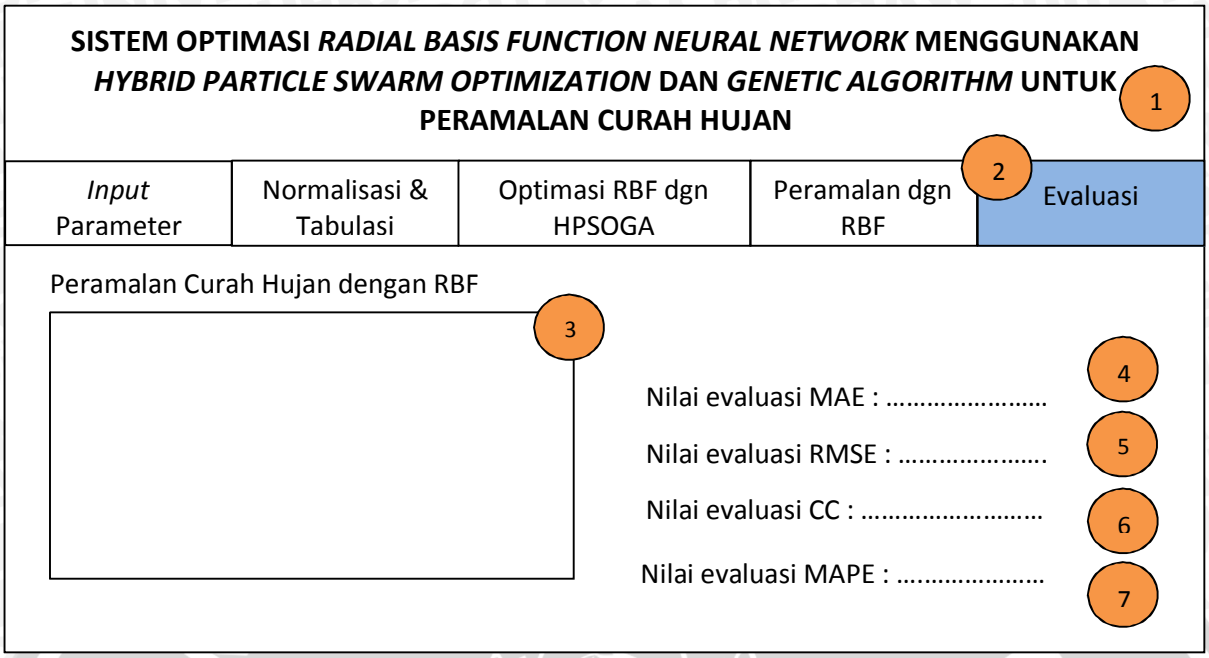



**Gambar 3.25 Rancangan antarmuka halaman peramalan dengan RBF**

### 3.4.5 Halaman evaluasi

Halaman evaluasi merupakan sebuah halaman untuk menampilkan hasil evaluasi dari perhitungan peramalan curah hujan dengan RBF pada langkah sebelumnya. Rancangan halaman evaluasi ditunjukkan pada Gambar 3.26 dengan keterangan sebagai berikut:

1. Judul sistem
2. *Tab control* untuk menampilkan proses peramalan curah hujan dengan menggunakan RBF.
3. *Data grid view* untuk menampilkan hasil proses peramalan curah hujan menggunakan RBF dalam bentuk tabel.
4. *Label* untuk menampilkan nilai evaluasi *Mean Absolute Error* (MAE).
5. *Label* untuk menampilkan nilai evaluasi *Root Mean Square Error* (RMSE).
6. *Label* untuk menampilkan nilai evaluasi *Correlation Coefficient* (CC).
7. *Label* untuk menampilkan nilai evaluasi *Mean Absolute Percentage Error* (MAPE).




**Gambar 3.26 Rancangan antarmuka halaman evaluasi**

## 3.5 Perancangan uji coba dan evaluasi

Langkah selanjutnya setelah membuat perancangan sistem adalah membuat perancangan uji coba dan evaluasi. Pengujian dibagi menjadi dua yaitu pengujian parameter dan pengujian data.

### 3.5.1 Pengujian parameter

Pengujian parameter dalam penelitian ini mencakup pengujian jumlah *node* masukan, jumlah *node* tersembunyi, jumlah populasi, jumlah iterasi, kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*), nilai *delta* tukar silang, serta kombinasi *learning rate* (*γ1* dan *γ2*). Skenario pengujian parameter yang akan dilakukan dijelaskan pada sub-bab 3.5.1.1 sampai 3.5.1.7.

#### 3.5.1.1 Pengujian jumlah *node* masukan yang optimal

Uji coba banyaknya *node* masukan dilakukan untuk mengetahui jumlah *node* masukan jaringan RBF yang dibutuhkan untuk mendapatkan hasil peramalan curah hujan yang paling akurat. Banyaknya *node* masukan yang di uji coba adalah kelipatan 1. Rancangan uji coba banyaknya *node* masukan ditunjukkan pada Tabel 3.33.

**Tabel 3.33 Rancangan uji coba jumlah *node* masukan**

| Jumlah *node* masukan | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1 | | | | | | | | | | | |
| 2 | | | | | | | | | | | |
| 3 | | | | | | | | | | | |
| 4 | | | | | | | | | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |

### 3.5.1.2 Pengujian jumlah *node* tersembunyi yang optimal

Uji coba banyaknya *node* tersembunyi dilakukan untuk mengetahui jumlah *node* tersembunyi jaringan RBF yang dibutuhkan untuk mendapatkan hasil peramalan curah hujan yang paling akurat. Banyaknya *node* masukan yang di uji coba adalah kelipatan 1. Rancangan uji coba banyaknya *node* tersembunyi ditunjukkan pada Tabel 3.34.

**Tabel 3.34 Rancangan uji coba jumlah *node* tersembunyi**

| Jumlah *node* tersembunyi | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1 | | | | | | | | | | | |
| 2 | | | | | | | | | | | |
| 3 | | | | | | | | | | | |
| 4 | | | | | | | | | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |

### 3.5.1.3 Pengujian jumlah populasi yang optimal

Uji coba jumlah populasi dilakukan untuk mengetahui jumlah populasi yang tepat untuk menghasilkan parameter jaringan RBF yang paling optimal. Banyaknya

populasi yang digunakan pada penelitian ini adalah kelipatan 40 sesuai dengan referensi utama dari skripsi ini yang menyatakan untuk membangkitkan populasi sebesar 4*N* individual (Wu, Long, dan Liu, 2015). Rancangan uji coba jumlah populasi ditunjukkan pada Tabel 3.35.

**Tabel 3.35 Rancangan uji coba jumlah populasi**

| Jumlah populasi | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke-*i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 40 | | | | | | | | | | | |
| 80 | | | | | | | | | | | |
| 120 | | | | | | | | | | | |
| 160 | | | | | | | | | | | |
| 200 | | | | | | | | | | | |
| 240 | | | | | | | | | | | |
| 280 | | | | | | | | | | | |
| 320 | | | | | | | | | | | |
| 360 | | | | | | | | | | | |
| 400 | | | | | | | | | | | |

### 3.5.1.4 Pengujian jumlah iterasi yang optimal

Uji coba banyaknya iterasi dilakukan untuk mengetahui jumlah iterasi yang dibutuhkan untuk mendapatkan parameter jaringan RBF yang paling optimal. Banyaknya iterasi yang di uji coba adalah kelipatan 100 dengan jumlah populasi yang yang memiliki nilai *fitness* tertinggi pada hasil pengujian populasi sebelumnya. Rancangan uji coba banyaknya iterasi ditunjukkan pada Tabel 3.36.

**Tabel 3.36 Rancangan uji coba jumlah iterasi**

| Jumlah iterasi | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 100 | | | | | | | | | | | |
| 200 | | | | | | | | | | | |
| 300 | | | | | | | | | | | |
| 400 | | | | | | | | | | | |
| 500 | | | | | | | | | | | |
| 600 | | | | | | | | | | | |
| 700 | | | | | | | | | | | |
| 800 | | | | | | | | | | | |
| 900 | | | | | | | | | | | |
| 1000 | | | | | | | | | | | |

### 3.5.1.5 Pengujian kombinasi probabilitas tukar silang (*pc*) dan probabilitas mutasi (*pm*) yang optimal

Uji coba kombinasi probabilitas tukar silang (*pc*) dan probabilitas mutasi (*pm*) dilakukan untuk mengetahui kombinasi *pc* dan *pm* terbaik untuk mendapatkan parameter jaringan RBF yang paling optimal. Rentang nilai *pc* dan *pm* yang digunakan pada pengujian ini adalah antara 0 sampai 1 dengan nilai yang berbeda antara *pc* dan *pm*. Rancangan uji coba kombinasi probabilitas tukar silang dan mutasi ditunjukkan pada Tabel 3.37.

**Tabel 3.37 Rancangan uji coba kombinasi *pc* dan *pm***

| Kombinasi | | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Percobaan ke- *i* | | | | | | | | | | |
| *pc* | *pm* | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 0 | 1 | | | | | | | | | | | |
| 0.1 | 0.9 | | | | | | | | | | | |
| 0.2 | 0.8 | | | | | | | | | | | |
| 0.3 | 0.7 | | | | | | | | | | | |
| 0.4 | 0.6 | | | | | | | | | | | |
| 0.5 | 0.5 | | | | | | | | | | | |
| 0.6 | 0.4 | | | | | | | | | | | |
| 0.7 | 0.3 | | | | | | | | | | | |
| 0.8 | 0.2 | | | | | | | | | | | |
| 0.9 | 0.1 | | | | | | | | | | | |
| 1 | 0 | | | | | | | | | | | |

### 3.5.1.6 Pengujian nilai *delta* (*δ*) tukar silang yang optimal

Uji coba nilai *delta* (*δ*) tukar silang dilakukan untuk mengetahui rentang nilai acak yang yang tepat pada metode *extended intermediate crossover* untuk menghasilkan parameter jaringan RBF yang paling optimal. Rancangan uji coba nilai *delta* (*δ*) tukar silang ditunjukkan pada Tabel 3.38.

**Tabel 3.38 Rancangan uji coba nilai *delta (δ)* tukar silang**

| Nilai *delta* | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke-*i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 0 | | | | | | | | | | | |
| 0.05 | | | | | | | | | | | |
| 0.15 | | | | | | | | | | | |
| 0.25 | | | | | | | | | | | |
| 0.35 | | | | | | | | | | | |
| 0.45 | | | | | | | | | | | |

**Tabel 3.38 Rancangan uji coba nilai *delta (δ)* tukar silang (lanjutan)**

| Nilai *delta* | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke-*i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 0.55 | | | | | | | | | | | |
| 0.65 | | | | | | | | | | | |
| 0.75 | | | | | | | | | | | |
| 0.85 | | | | | | | | | | | |

### 3.5.1.7 Pengujian kombinasi *learning rate* (*γ1* dan *γ2*) yang optimal

Uji coba penentuan nilai *learning rate* ($\gamma_1$ dan $\gamma_2$) dilakukan untuk mengetahui nilai *learning rate* pada metode PSO yang dibutuhkan untuk mendapatkan parameter jaringan RBF yang paling optimal. Rentang nilai $\gamma_1$ dan $\gamma_2$ yang digunakan pada pengujian ini adalah antara 1 sampai 2. Rancangan uji coba penentuan nilai *learning rate* ditunjukkan pada Tabel 3.39.

**Tabel 3.39 Rancangan uji coba kombinasi $\gamma_1$ dan $\gamma_2$**

| Kombinasi | | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Percobaan ke- *i* | | | | | | | | | | |
| $\gamma_1$ | $\gamma_2$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1.0 | 1.0 | | | | | | | | | | | |
| 1.0 | 1.5 | | | | | | | | | | | |
| 1.0 | 2.0 | | | | | | | | | | | |
| 1.5 | 1.0 | | | | | | | | | | | |
| 1.5 | 1.5 | | | | | | | | | | | |
| 1.5 | 2.0 | | | | | | | | | | | |
| 2.0 | 1.0 | | | | | | | | | | | |
| 2.0 | 1.5 | | | | | | | | | | | |
| 2.0 | 2.0 | | | | | | | | | | | |

## 3.5.2 Pengujian data

Pengujian data dalam penelitian ini mencakup pengujian peramalan data uji menggunakan data latih dengan parameter optimal dan pengujian jumlah data latih optimal. Skenario pengujian data yang akan dilakukan dijelaskan pada sub-bab 3.5.2.1 dan 3.5.2.2.

### 3.5.2.1 Pengujian peramalan data uji menggunakan data latih dengan parameter yang optimal

Uji coba peramalan data uji menggunakan data latih dilakukan untuk mengetahui apakah parameter yang dihasilkan pada pengujian sebelumnya dapat meramalkan data itu sendiri dengan baik atau tidak. Banyaknya data latih yang

digunakan adalah dua variasi dengan panjang interval data curah hujan masing-masing 1 tahun. Rancangan uji coba peramalan data uji dengan data latih ditunjukkan pada Tabel 3.40.

**Tabel 3.40 Rancangan uji coba peramalan data uji menggunakan data latih**

| Data tahun ke- | Nilai MAE | | | | | | | | | | Rata-rata MAE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 2011 | | | | | | | | | | | |
| 2012 | | | | | | | | | | | |

### 3.5.2.2 Pengujian jumlah data latih yang optimal

Uji coba banyaknya data latih dilakukan untuk mengetahui jumlah data latih yang dapat menghasilkan peramalan curah hujan yang paling akurat. Banyaknya data latih yang digunakan adalah kelipatan 3 dimulai dari 9 bulan dan dengan menggunakan parameter optimal yang dihasilkan pengujian sebelumnya. Rancangan uji coba banyaknya data latih ditunjukkan pada Tabel 3.41.

**Tabel 3.41 Rancangan uji coba jumlah data latih**

| Banyak data latih | Nilai MAE | | | | | | | | | | Rata-rata MAE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 9 | | | | | | | | | | | |
| 12 | | | | | | | | | | | |
| 15 | | | | | | | | | | | |
| 18 | | | | | | | | | | | |
| 21 | | | | | | | | | | | |
| 24 | | | | | | | | | | | |
| 27 | | | | | | | | | | | |
| 30 | | | | | | | | | | | |
| 33 | | | | | | | | | | | |
| 36 | | | | | | | | | | | |

# BAB 4 IMPLEMENTASI

Bab ini membahas mengenai implementasi perangkat lunak berdasarkan hasil yang telah diperoleh dari analisis kebutuhan dan proses perancangan perangkat lunak yang telah dibuat. Pembahasan pada bab ini terdiri dari lingkungan implementasi perangkat lunak, implementasi program, dan implementasi antarmuka sistem optimasi *Radial Basis Function Neural Network* menggunakan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan.

## 4.1 Lingkungan implementasi

Untuk melakukan implementasi perangkat lunak, terlebih dahulu perlu disiapkan lingkungan implementasi untuk memenuhi kebutuhan sistem. Lingkungan implementasi tersebut akan dijelaskan dalam sub bab lingkungan implementasi perangkat lunak dan perangkat keras sistem.

### 4.1.1 Lingkungan perangkat keras

Perangkat keras yang digunakan dalam dalam penelitian ini memiliki spesifikasi sebagai berikut :

1. *Processor Intel(R) Core(TM)* i3-2310M *CPU* @ 2.10 GHz (4 *CPUs*).
2. *Memory* RAM 4 GB.
3. *Hardisk* dengan kapasitas 500 GB.
4. Monitor 14 inch.
5. *Keyboard.*

### 4.1.2 Lingkungan perangkat lunak

Perangkat lunak yang digunakan dalam pengembangan perangkat lunak dalam penelitian ini adalah :

1. Sistem operasi Windows 8.1 Enterprise 32-bit.
2. Microsoft Visual C# 2010 Express digunakan untuk membangun aplikasi sistem optimasi RBF menggunakan HPSOGA untuk peramalan curah hujan.
3. Microsoft Office Excel 2013 digunakan untuk melakukan proses manualisasi perhitungan peramalan curah hujan dengan RBF yang dioptimasi dengan HPSOGA.
4. Microsoft Office Word 2013 digunakan untuk menyusun laporan penelitian.
5. Microsoft Office Visio 2007 digunakan untuk membuat diagram alir proses optimasi RBF menggunakan HPSOGA untuk peramalan curah hujan.

## 4.2 Implementasi program

Berdasarkan perancangan sistem yang telah dibuat pada bab 3, pada sub bab ini akan dibahas mengenai implementasi program. Sistem diimplementasikan pada kelas utama yang menampung seluruh proses, yaitu kelas *Form 1*. Pada *Form 1* dilakukan pemanggilan fungsi-fungsi seperti yang telah dijelaskan dalam perancangan diagram alir dalam sub bab 3.2. Fungsi-fungsi tersebut adalah sebagai berikut:

### 4.2.1 Fungsi normalisasi data

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan normalisasi data curah hujan dinamakan dengan fungsi Normalisasi. Implementasi fungsi normalisasi data curah hujan ditunjukkan pada *Source Code* 4.1.

```
1  static double[] Normalisasi(double[] cHujan, int jumlahData, double
2  Min, double Max, int rangeMin, int rangeMax)
3  {
4        double[] cHujanNorm = new double[jumlahData];
5
6        for (int x = 0; x <= jumlahData - 1; x++)
7        {
8              cHujanNorm[x] = ((cHujan[x] - Min) / (Max - Min)) *
9              (rangeMax - rangeMin) + rangeMin;
10       }
11
12       return cHujanNorm;
13 }
```

***Source Code* 4.1 Implementasi proses normalisasi data**

Penjelasan dari *Source Code* 4.1 adalah sebagai berikut :

1. Baris 4 merupakan proses deklarasi *array*.
2. Baris 6-10 merupakan proses untuk menghitung normalisasi data curah hujan sejumlah baris data curah hujan yang dimasukkan.
3. Baris 12 merupakan proses pengembalian data curah hujan yang telah dinormalisasi yaitu *array* cHujanNorm.

### 4.2.2 Fungsi tabulasi data *input* jaringan RBF

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan tabulasi data *input* pada jaringan RBF dinamakan dengan fungsi TabulasiInput. Implementasi fungsi tabulasi data *input* ditunjukkan pada *Source Code* 4.2.

```
1   static double[,] TabulasiInput(double[] cHujanNorm, int jumlahData,
2   int inpNodes, int barisData)
3   {
4         double[,] cHujanInp = new double[inpNodes, barisData];
5
6         for (int i = 0; i <= barisData - 1; i++)
7         {
8             for (int h = 0; h <= inpNodes - 1; h++)
9             {
10                cHujanInp[h, i] = cHujanNorm[i + h];
11            }
12        }
13
14        return cHujanInp;
15  }
```

***Source Code* 4.2 Implementasi proses tabulasi data *input* jaringan RBF**

Penjelasan dari *Source Code* 4.2 adalah sebagai berikut :

1. Baris 4 merupakan proses deklarasi *array*.
2. Baris 6-12 merupakan proses untuk melakukan tabulasi data *input* jaringan RBF sesuai dengan jumlah *node* masukan yang diinginkan.
3. Baris 14 merupakan proses pengembalian data curah hujan yang menjadi data *input* jaringan RBF yaitu *array* cHujanInp.

### 4.2.3 Fungsi tabulasi data *output* jaringan RBF

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan tabulasi data *output* pada jaringan RBF dinamakan dengan fungsi TabulasiOutput. Implementasi fungsi tabulasi data *output* ditunjukkan pada *Source Code* 4.3.

```
1   static double[] TabulasiOutput(double[] cHujanNorm, int jumlahData,
2   int inpNodes, int barisData)
3   {
4         double[] cHujanAktual = new double[barisData];
5
6         for (int i = 0; i <= barisData - 1; i++)
7         {
8             cHujanAktual[i] = cHujanNorm[i + inpNodes];
9         }
10
11        return cHujanAktual;
12  }
```

***Source Code* 4.3 Implementasi proses tabulasi data *output* jaringan RBF**

Penjelasan dari *Source Code* 4.3 adalah sebagai berikut :

1. Baris 4 merupakan proses deklarasi *array*.
2. Baris 6-9 merupakan proses untuk melakukan tabulasi data *output* jaringan RBF sesuai dengan jumlah *node* keluaran yang diinginkan yaitu 1.

3. Baris 11 merupakan proses pengembalian data curah hujan yang menjadi data *output* jaringan RBF yaitu *array* cHujanAktual.

### 4.2.4 Fungsi optimasi RBF dengan HPSOGA

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan optimasi RBF dengan HPSOGA dinamakan dengan fungsi OptimasiRBF. Dalam fungsi ini dilakukan pemanggilan fungsi-fungsi lain yaitu fungsi representasi kromosom, perhitungan *fitness*, perangkingan, tukar silang, mutasi, perbarui partikel *personal best* (*Pbest*) dan *global best* (*Pgest*), perbarui kecepatan dan posisi partikel, serta seleksi. Implementasi fungsi optimasi RBF dengan HPSOGA ditunjukkan pada *Source Code* 4.4.

```
1   static object[,] OptimasiRBF(int popSize, int inpNodes, int
2   hidNodes, int outNodes, double pc, double pm, double deltaMutasi,
3   double deltaTukarSilang, int iterasi, double rangeMin, double
4   rangeMax, double[,] cHujanInput, double[] cHujanAktual, int
5   barisData, RichTextBox richTextBox1, ProgressBar progressBar1){
6           lengthBin = hidNodes * inpNodes;
7           lengthReal = (hidNodes * inpNodes) + hidNodes + (hidNodes *
8           outNodes);
9           popGA = popSize / 2;
10          popSizeCross = Convert.ToInt32(Math.Round(pc *
11          Convert.ToDouble(popGA)));
12          popSizeMut = Convert.ToInt32(Math.Round(pm *
13          Convert.ToDouble(popGA)));
14          totalParentOffspring = popSizeCross + popSizeMut + (popSize
15          / 2) + popSize;
16
17          int iterasiTemp = iterasi;
18
19          object[,] krom = RepresentasiKromosom(popSize, lengthBin,
20          lengthReal, rangeMin, rangeMax);
21          object[,] kromCross = new object[popSizeCross, (lengthBin +
22          lengthReal)];
23          object[,] kromMut = new object[popSizeMut, (lengthBin +
24          lengthReal)];
25          object[,] pBest = new object[popSize / 2, (lengthBin +
26          lengthReal)];
27          object[,] pGest = new object[1, (lengthBin + lengthReal)];
28          double[] pFit = new double[popSize / 2];
29          double[] pGestFit = new double[1];
30          object[,] pNew = new object[popSize / 2, (lengthBin +
31          lengthReal)];
32          double[] fitnessCross;
33          double[] fitnessMut;
34          double[] fitnesspNew;
35          double[] fitness = HitungFitness(inpNodes, hidNodes,
36          cHujanInput, cHujanAktual, popSize, krom, barisData,
37          rangeMin, rangeMax);
38
39          krom = Perangkingan(lengthBin, lengthReal, popSize, krom,
40          fitness);
41
42          do
43          {
44              kromCross = TukarSilang(lengthBin, lengthReal, popGA,
45              popSizeCross, krom, deltaTukarSilang);
```

```
46              kromMut = Mutasi(lengthBin, lengthReal, popGA,
47              popSizeMut, krom, rangeMin, rangeMax, deltaMutasi);
48
49              pBest = UpdatePbestPgest(lengthBin, lengthReal, popSize,
50              krom, fitness, pBest, pFit, pGest, pGestFit);
51
52              pNew = UpdateVelocityPosition(lengthBin, lengthReal,
53              popSize, krom, pBest, pGest, rangeMin, rangeMax, gamma1,
54              gamma2);
55
56              fitnessCross = HitungFitness(inpNodes, hidNodes,
57              cHujanInput, cHujanAktual, popSizeCross, kromCross,
58              barisData, rangeMin, rangeMax);
59
60              fitnessMut = HitungFitness(inpNodes, hidNodes,
61              cHujanInput, cHujanAktual, popSizeMut, kromMut,
62              barisData, rangeMin, rangeMax);
63
64              fitnesspNew = HitungFitness(inpNodes, hidNodes,
65              cHujanInput, cHujanAktual, (popSize / 2), pNew,
66              barisData, rangeMin, rangeMax);
67
68              krom = Seleksi(lengthBin, lengthReal, popSize,
69              popSizeCross, popSizeMut, krom, kromCross, kromMut,
70              pNew, fitness, fitnessCross, fitnessMut, fitnesspNew,
71              totalParentOffspring);
72
73              double progressBar = ((Convert.ToDouble(iterasiTemp) -
74              Convert.ToDouble(iterasi)) /
75              Convert.ToDouble(iterasiTemp) * 100);
76              progressBar1.Value = Convert.ToInt32(progressBar);
77
78              iterasi = iterasi - 1;
79
80              richTextBox1.Text = richTextBox1.Text + "Iterasi ke-" +
81              (iterasiTemp - iterasi) + " :\n \n";
82              richTextBox1.Text = richTextBox1.Text + "Kromosom
83              terbaik : \t";
84
85              for (int k = 0; k <= (lengthBin + lengthReal) - 1; k++)
86              {
87                  richTextBox1.Text = richTextBox1.Text + krom[0, k] +
88                  "\t";
89              }
90
91              richTextBox1.Text = richTextBox1.Text + "\n\nFitness : "
92              + fitness[0] + "\n\n\n";
93          }
94
95          while (iterasi > 0);
96
97          progressBar1.Value = 100;
98          MessageBox.Show("Optimasi RBF-HPSOGA telah selesai
99          dilakukan");
100
101         return krom;
102 }
```

***Source Code* 4.4 Implementasi proses optimasi RBF dengan HPSOGA**

Penjelasan dari *Source Code* 4.4 adalah sebagai berikut :

1. Baris 6-17 merupakan proses inisialisasi variabel.
2. Baris 19-20 merupakan proses untuk membangkitkan kromosom awal.
3. Baris 21-34 merupakan proses deklarasi *array* dan objek.
4. Baris 35-37 merupakan proses untuk menghitung nilai *fitness* dari kromosom yang telah dibangkitkan.
5. Baris 39-40 merupakan proses untuk mengurutkan kromosom berdasarkan nilai *fitness* dari tertinggi ke terendah.
6. Baris 43-95 merupakan proses optimasi RBF dengan HPSOGA selama iterasi belum mencapai iterasi maksimal.
7. Baris 44-45 merupakan proses untuk melakukan tukar silang pada kromosom-kromosom yang memiliki nilai *fitness* tertinggi.
8. Baris 46-47 merupakan proses untuk melakukan mutasi pada kromosom-kromosom yang memiliki nilai *fitness* tertinggi.
9. Baris 49-50 merupakan proses untuk memperbarui nilai partikel *personal best* (*Pbest*) yang diambil dari kromosom-kromosom dengan nilai *fitness* terendah.
10. Baris 52-54 merupakan proses untuk memperbarui nilai kecepatan partikel yang dibentuk dari kromosom-kromosom dengan nilai *fitness* tertinggi dan memperbarui nilai posisi partikel yang dibentuk dari kromosom-kromosom dengan nilai *fitness* terendah.
11. Baris 56-66 merupakan proses menghitung nilai *fitness* untuk kromosom *offspring* hasil metode GA dan PSO.
12. Baris 68-71 merupakan proses seleksi kromosom gabungan dari kromosom awal dan semua *offspring* yang diperoleh dari metode GA dan PSO.
13. Baris 73-92 merupakan proses menampilkan *progress bar* perhitungan optimasi RBF-HPSOGA dan hasil kromosom terbaik tiap iterasi pada antarmuka sistem.
14. Baris 101 merupakan proses pengembalian kromosom berisi parameter metode RBF yang telah dioptimasi dengan metode HPSOGA yaitu objek krom.

#### *4.2.4.1 Fungsi representasi kromosom*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk membangkitkan kromosom dinamakan dengan fungsi RepresentasiKromosom. Implementasi fungsi representasi kromosom ditunjukkan pada *Source Code* 4.5.

```
1  static object[,] RepresentasiKromosom(int popSize, int lengthBin, int
2  lengthReal, int rangeMin, int rangeMax)
3  {
4          object[,] krom = new object[popSize, (lengthBin + lengthReal)];
5          int checkKrom = 0;
6          Random r = new Random();
7
8          for (int j = 0; j <= popSize - 1; j++)
9          {
```

```
10              checkKrom = 0;
11              do
12              {
13                  for (int k = 0; k <= lengthBin - 1; k++)
14                  {
15                    krom[j, k] = r.Next(0, 2);
16                    checkKrom = checkKrom + Convert.ToInt32(krom[j, k]);
17                  }
18              } while (checkKrom == 0);
19
20              for (int m = 0; m <= lengthReal - 1; m++)
21              {
22                   krom[j, lengthBin + m] = r.NextDouble() * (rangeMax -
23                   rangeMin) + rangeMin;
24              }
25          }
26
27          return krom;
28  }
```

***Source Code* 4.5 Implementasi proses representasi kromosom**

Penjelasan dari *Source Code* 4.5 adalah sebagai berikut :

1. Baris 4-6 merupakan proses deklarasi objek dan inisialisasi variabel.
2. Baris 8-25 merupakan proses untuk membangkitkan kromosom sebanyak jumlah populasi.
3. Baris 13-17 merupakan proses untuk membangkitkan nilai antara 0 atau 1 sebanyak panjang gen yang dikodekan dalam biner.
4. Baris 18 merupakan proses pengecekan. Apabila hasil pengecekan proses pembangkitan *string* biner (checkKrom) bernilai 0 yang berarti tidak ada koneksi dari *node* masukan ke *node* tersembunyi pada jaringan RBF, maka ulangi proses pembangkitan kromosom untuk yang dikodekan dalam biner.
5. Baris 20-24 merupakan proses untuk membangkitkan nilai real pada rentang minimal dan maksimal yang telah dimasukkan sebanyak panjang gen yang dikodekan dalam real.
6. Baris 27 merupakan proses pengembalian objek krom.

#### *4.2.4.2 Fungsi perhitungan fitness*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk menghitung *fitness* dari kromosom yang telah dibangkitkan dinamakan dengan fungsi HitungFitness. Implementasi fungsi perhitungan *fitness* ditunjukkan pada *Source Code* 4.6.

```
1   static double[] HitungFitness(int inpNodes, int hidNodes, double[,]
2   cHujanInput, double[] cHujanAktual, int popSize, object[,] krom, int
3   barisData, int rangeMin, int rangeMax)
4   {
5           double[] error = new double[popSize];
6           double[] fitness = new double[popSize];
7
8           for (int z = 0; z <= popSize - 1; z++)
```

```
9         {
10            double sumDiff = 0;
11            double[] cHujanPrediksi = PeramalanRBF(inpNodes, hidNodes,
12            krom, cHujanInput, barisData, z, rangeMin, rangeMax);
13
14            for (int x = 0; x <= barisData - 1; x++)
15            {
16                sumDiff = sumDiff + Math.Abs(cHujanPrediksi[x] -
17                cHujanAktual[x]);
18            }
19
20            error[z] = (1 / Convert.ToDouble(barisData)) * sumDiff;
21            fitness[z] = 1 / (1 + error[z]);
22        }
23
24        return fitness;
25    }
```

***Source Code* 4.6 Implementasi proses perhitungan *fitness***

Penjelasan dari *Source Code* 4.6 adalah sebagai berikut :

1. Baris 5-6 merupakan proses deklarasi *array*.
2. Baris 8-22 merupakan proses untuk menghitung nilai *fitness* kromosom sebanyak jumlah populasi kromosom.
3. Baris 11-12 merupakan proses peramalan curah hujan dengan metode RBF menggunakan kromosom yang ditunjuk sebagai parameter-parameternya.
4. Baris 14-18 merupakan proses menghitung perbedaan nilai antara hasil peramalan curah hujan dengan data curah hujan keluaran yang sebenarnya untuk keperluan perhitungan nilai *error* sebanyak jumlah data masukan dan keluaran yang digunakan.
5. Baris 20 merupakan proses menghitung nilai *error* dari hasil peramalan.
6. Baris 21 merupakan proses menghitung nilai *fitness* dari kromosom yang ditunjuk.
7. Baris 24 merupakan proses pengembalian *array fitness*.

#### *4.2.4.3 Fungsi perangkingan*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan perangkingan dari kromosom dari nilai *fitness* tertinggi ke terendah dinamakan dengan fungsi Perangkingan. Implementasi fungsi perangkingan ditunjukkan pada *Source Code* 4.7.

```
1   static object[,] Perangkingan(int lengthBin, int lengthReal, int
2   popSize, object[,] krom, double[] fitness)
3   {
4       int[] fitnessKey = new int[popSize];
5       object[,] kromSorted = new object[popSize, (lengthBin +
6       lengthReal)];
7
8       for (int z = 0; z <= popSize - 1; z++)
9       {
```

```
10              fitnessKey[z] = z;
11          }
12
13          Array.Sort(fitness, fitnessKey);
14          int tempPopSize = popSize - 1;
15
16          for (int y = 0; y <= popSize - 1; y++)
17          {
18              int indexSort = fitnessKey[y];
19
20              for (int v = 0; v <= (lengthBin + lengthReal) - 1; v++)
21              {
22                  kromSorted[tempPopSize, v] = krom[indexSort, v];
23              }
24
25              tempPopSize = tempPopSize - 1;
26          }
27
28          Array.Reverse(fitness);
29          return kromSorted;
30      }
```

***Source Code* 4.7 Implementasi proses perangkingan**

Penjelasan dari *Source Code* 4.7 adalah sebagai berikut :

1. Baris 4-6 merupakan proses deklarasi *array* dan objek.
2. Baris 8-13 merupakan proses untuk mengurutkan nilai *fitness* kromosom dari terendah ke tertinggi beserta dengan *key*-nya (indeks *array fitness*-nya).
3. Baris 14-26 merupakan proses menyalin krom dengan nilai *fitness* terendah ke tertinggi ke dalam kromSorted dimulai dari indeks yang paling besar.
4. Baris 28 merupakan proses membalik urutan *array fitness* dari *fitness* terendah ke tertinggi menjadi tertinggi ke terendah.
5. Baris 29 merupakan proses pengembalian kromosom yang telah diurutkan berdasarkan nilai *fitness* tertinggi ke terendah yaitu objek kromSorted.

#### *4.2.4.4 Fungsi tukar silang*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan metode tukar silang kromosom dinamakan dengan fungsi TukarSilang. Implementasi fungsi tukar silang ditunjukkan pada *Source Code* 4.8.

```
1   static object[,] TukarSilang(int lengthBin, int lengthReal, int
2   popGA, int popSizeCross, object[,] krom, double deltaTukarSilang)
3   {
4           object[,] kromCross = new object[popSizeCross, (lengthBin +
5       lengthReal)];
6           Random r = new Random();
7           double[] randomReal = new double[lengthReal];
8           int checkKrom = 0;
9           int kromRandom1 = 0, kromRandom2 = 0;
10          double minRandom = -deltaTukarSilang;
11          double maxRandom = 1 + deltaTukarSilang;
12
13          for (int k = 0; k <= lengthReal - 1; k++)
```

```
14          {
15              randomReal[k] = r.NextDouble() * (maxRandom - minRandom) +
16              minRandom;
17          }
18
19
20          for (int x = 0; x <= popSizeCross - 1; x++)
21          {
22
23              if ((x % 2) != 0)
24              {
25                  checkKrom = 0;
26
27                  for (int y = 0; y <= lengthBin - 1; y++)
28                  {
29                      if (y < Math.Round(Convert.ToDouble(lengthBin / 2)))
30                      {
31                          kromCross[x, y] = krom[kromRandom2, y];
32                          checkKrom = checkKrom +
33                          Convert.ToInt32(kromCross[x, y]);
34                      }
35                      else
36                      {
37                          kromCross[x, y] = krom[kromRandom1, y];
38                          checkKrom = checkKrom +
39                          Convert.ToInt32(kromCross[x, y]);
40                      }
41                  }
42
43                  for (int z = 0; z <= lengthReal - 1; z++)
44                  {
45                      kromCross[x, z + lengthBin] =
46                      Convert.ToDouble(krom[kromRandom2, z + lengthBin]) +
47                      (randomReal[z] * (Convert.ToDouble(krom[kromRandom1,
48                      z + lengthBin]) - Convert.ToDouble(krom[kromRandom2,
49                      z + lengthBin])));
50
51                      if (Convert.ToDouble(kromCross[x, z + lengthBin]) >
52                      Convert.ToDouble(rangeMax))
53                      {
54                          kromCross[x, z + lengthBin] =
55                          Convert.ToDouble(rangeMax);
56                      }
57                      else if (Convert.ToDouble(kromCross[x, z +
58                      lengthBin]) < Convert.ToDouble(rangeMin))
59                      {
60                          kromCross[x, z + lengthBin] =
61                          Convert.ToDouble(rangeMin);
62                      }
63                  }
64
65                  if (checkKrom == 0)
66                  {
67                      x = x - 1;
68                  }
69              }
70
71
72              if ((x % 2) == 0)
73              {
74                  checkKrom = 0;
75
```

```
76                  do
77                  {
78                      kromRandom1 = r.Next(0, popGA);
79                      kromRandom2 = r.Next(0, popGA);
80
81                      while (kromRandom1 == kromRandom2)
82                      {
83                          kromRandom2 = r.Next(0, popGA);
84                      }
85
86                      for (int y = 0; y <= lengthBin - 1; y++)
87                      {
88                          if (y < Math.Round(Convert.ToDouble(lengthBin
89                          / 2)))
90                          {
91                              kromCross[x, y] = krom[kromRandom1, y];
92                              checkKrom = checkKrom +
93                              Convert.ToInt32(kromCross[x, y]);
94                          }
95                          else
96                          {
97                              kromCross[x, y] = krom[kromRandom2, y];
98                              checkKrom = checkKrom +
99                              Convert.ToInt32(kromCross[x, y]);
100                         }
101                     }
102                 }
103
104                 while (checkKrom == 0);
105
106                 for (int z = 0; z <= lengthReal - 1; z++)
107                 {
108                     kromCross[x, z + lengthBin] =
109                     Convert.ToDouble(krom[kromRandom1, z +
110                     lengthBin]) + (randomReal[z] *
111                     (Convert.ToDouble(krom[kromRandom2, z +
112                     lengthBin]) - Convert.ToDouble(krom[kromRandom1,
113                     z + lengthBin])));
114
115                     if (Convert.ToDouble(kromCross[x, z +
116                     lengthBin]) > Convert.ToDouble(rangeMax))
117                     {
118                         kromCross[x, z + lengthBin] =
119                         Convert.ToDouble(rangeMax);
120                     }
121                     else if (Convert.ToDouble(kromCross[x, z +
122                     lengthBin]) < Convert.ToDouble(rangeMin))
123                     {
124                         kromCross[x, z + lengthBin] =
125                         Convert.ToDouble(rangeMin);
126                     }
127                 }
128             }
129
130         }
131
132         return kromCross;
133 }
```

***Source Code* 4.8 Implementasi proses tukar silang**

Penjelasan dari *Source Code* 4.8 adalah sebagai berikut :

1. Baris 4-11 merupakan proses deklarasi *array* dan objek serta inisialisasi variabel.
2. Baris 13-17 merupakan proses membangkitkan nilai acak yang akan digunakan dalam proses tukar silang kromosom pengkodean nilai real.
3. Baris 20-30 merupakan proses untuk melakukan tukar silang kromosom sebanyak jumlah populasi kromosom hasil tukar silang.
4. Baris 23-69 merupakan proses untuk melakukan tukar silang kromosom pada indeks ganjil. Baris 27-41 merupakan proses tukar silang untuk gen yang dikodekan dalam biner dengan metode tukar silang *one-point*. Baris 43-63 merupakan proses tukar silang untuk gen yang dikodekan dalam nilai real. Sedangkan baris 65-69 merupakan proses pengecekan hasil tukar silang pada *string* biner yang apabila bernilai 0 maka akan menghapus 1 kromosom hasil tukar silang yang telah dibentuk sebelumnya.
5. Baris 72-128 merupakan proses untuk melakukan tukar silang kromosom pada indeks genap. Baris 78-79 merupakan proses pemilihan 2 *parent* secara acak dari populasi. Baris 81-84 merupakan proses pengecekan *parent,* apabila yang dipilih sama maka ulangi pemilihan *parent* kedua sampai *parent*-nya berbeda. Baris 86-101 merupakan proses tukar silang untuk gen yang dikodekan dalam biner dengan metode tukar silang *one-point*. Baris 104 merupakan proses pengecekan hasil proses tukar silang kromosom yang apabila bernilai 0 berarti tidak ada koneksi dari *node* masukan ke *node* tersembunyi pada jaringan RBF sehingga perlu mengulangi proses tukar silang *one-point*. Baris 106-126 merupakan proses tukar silang untuk gen yang dikodekan dalam nilai real.
6. Baris 132 merupakan proses pengembalian kromosom hasil tukar silang yaitu objek kromCross.

### *4.2.4.5 Fungsi mutasi*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan metode mutasi kromosom dinamakan dengan fungsi Mutasi. Implementasi fungsi mutasi kromosom ditunjukkan pada Source Code 4.9.

```
1  static object[,] Mutasi(int lengthBin, int lengthReal, int popGA,
2  int popSizeMut, object[,] krom, int rangeMin, int rangeMax, double
3  deltaMutasi){
4          object[,] kromMut = new object[popSizeMut, (lengthBin +
5          lengthReal)];
6          Random r = new Random();
7          double[] randomReal = new double[lengthReal];
8          double[] random = new double[lengthReal];
9          double[] minRandom = new double[lengthReal];
10         double[] maxRandom = new double[lengthReal];
11         int checkKrom = 0;
12         int indexRandom, kromRandom, tempBin;
13
14         for (int x = 0; x <= popSizeMut - 1; x++)
15         {
```

```
16              do
17              {
18                  indexRandom = r.Next(0, lengthBin);
19                  kromRandom = r.Next(0, popGA);
20
21                  for (int y = 0; y <= lengthBin - 1; y++)
22                  {
23                     kromMut[x, y] = krom[kromRandom, y];
24                     checkKrom = checkKrom +
25                     Convert.ToInt32(kromMut[x, y]);
26                  }
27
28                  if (Convert.ToInt32(kromMut[x, indexRandom]) == 0)
29                  {
30                     tempBin = 1;
31                     checkKrom = checkKrom + 1;
32                  }
33
34                  else
35                  {
36                     tempBin = 0;
37                     checkKrom = checkKrom - 1;
38                  }
39              }
40
41              while (checkKrom == 0);
42
43              kromMut[x, indexRandom] = tempBin;
44
45              for (int z = 0; z <= lengthReal - 1; z++)
46              {
47                  random[z] = r.NextDouble();
48                  minRandom[z]     =     0     -     deltaMutasi     -
49                  Convert.ToDouble(krom[kromRandom, z + lengthBin]);
50                  maxRandom[z]     =     1     +     deltaMutasi     +
51                  Convert.ToDouble(krom[kromRandom, z + lengthBin]);
52                  randomReal[z]    =    random[z]    *    (maxRandom[z]    -
53                  minRandom[z]) + minRandom[z];
54                  kromMut[x,       z       +       lengthBin]       =
55                  Convert.ToDouble(krom[kromRandom, z + lengthBin]) +
56                  randomReal[z];
57
58                  if  (Convert.ToDouble(kromMut[x,  z  +  lengthBin])  >
59                  Convert.ToDouble(rangeMax))
60                  {
61                       kromMut[x,       z       +       lengthBin]       =
62                        Convert.ToDouble(rangeMax);
63                  }
64                  else if (Convert.ToDouble(kromMut[x, z + lengthBin])
65                  < Convert.ToDouble(rangeMin))
66                  {
67                       kromMut[x,       z       +       lengthBin]       =
68                        Convert.ToDouble(rangeMin);
69                  }
70              }
71          }
72
73          return kromMut;
74      }
```

***Source Code* 4.9 Implementasi proses mutasi**

Penjelasan dari *Source Code* 4.9 adalah sebagai berikut :

1. Baris 4-12 merupakan proses deklarasi *array* dan objek serta inisialisasi variabel.
2. Baris 14-71 merupakan proses untuk melakukan mutasi kromosom sebanyak jumlah populasi kromosom hasil mutasi.
3. Baris 18-19 merupakan proses pemilihan 1 *parent* secara acak dari populasi metode GA serta 1 indeks secara acak dari gen yang dikodekan dalam biner.
4. Baris 21-38 merupakan proses mutasi untuk gen yang dikodekan dalam biner dengan metode mutasi *bitwise* sesuai dengan indeks gen yang telah dipilih sebelumnya.
5. Baris 41 merupakan proses pengecekan. Apabila hasil pengecekan proses mutasi kromosom bernilai 0 yang berarti tidak ada koneksi dari *node* masukan ke *node* tersembunyi pada jaringan RBF, maka ulangi proses mutasi *bitwise*.
6. Baris 45-70 merupakan proses mutasi untuk gen yang dikodekan dalam nilai real. Baris 58-63 merupakan proses pengecekan gen hasil mutasi yang dikodekan dalam nilai real, apabila nilainya lebih dari rentang maksimal maka hasil mutasi akan diisi dengan rentang maksimal. Sedangkan baris 64-69 merupakan proses pengecekan sebaliknya yang apabila nilai gen hasil mutasi kurang dari rentang minimal maka hasil mutasi akan diisi dengan rentang minimal.
7. Baris 73 merupakan proses pengembalian kromosom hasil mutasi yaitu objek kromMut.

#### *4.2.4.6 Fungsi perbarui partikel personal best (Pbest) dan global best (Pgest)*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk memperbarui partikel *personal best* (*Pbest*) dan *global best* (*Pgest*) dinamakan dengan fungsi UpdatePbestPgest. Implementasi fungsi memperbarui partikel *personal best* (*Pbest*) dan *global best* (*Pgest*) ditunjukkan pada *Source Code* 4.10.

```
1   static object[,] UpdatePbestPgest(int lengthBin, int lengthReal, int
2   popSize, object[,] krom, double[] fitness, object[,] pBest, double[]
3   pFit, object[,] pGest, double[] pGestFit)
4   {
5         for (int x = 0; x <= (popSize / 2) - 1; x++)
6         {
7
8               if (pFit[x] < fitness[(popSize / 2) + x])
9               {
10               pFit[x] = fitness[(popSize / 2) + x];
11
12               for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
13               {
14                     pBest[x, z] = krom[(popSize / 2) + x, z];
15               }
16              }
17        }
18
19        if (pGestFit[0] < fitness[0])
20        {
```

```
21              for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
22              {
23                  pGest[0, z] = krom[0, z];
24              }
25          }
26
27          return pBest;
28      }
```

***Source Code* 4.10 Implementasi proses perbarui partikel *personal best* (*Pbest*) dan *global best* (*Pgest*)**

Penjelasan dari *Source Code* 4.10 adalah sebagai berikut :

1. Baris 5-17 merupakan proses memperbarui partikel *personal best* (*Pbest*) sejumlah setengah dari populasi awal. Baris 8-16 merupakan proses penyalinan objek krom ke objek *pBest* apabila *fitness personal best* dari iterasi yang sebelumnya lebih kecil dari *fitness personal* kromosom terbaik iterasi saat ini.
2. Baris 19-25 merupakan proses memperbarui partikel *global best* (*Pbest*). Baris 21-24 merupakan proses penyalinan objek krom dengan indeks 0 (individu dengan nilai *fitness* terbaik) ke objek *pGest* apabila *fitness global best* dari iterasi yang sebelumnya lebih kecil dari *fitness* kromosom terbaik iterasi saat ini.
3. Baris 27 merupakan proses pengembalian partikel *personal best* (*Pbest*) yaitu objek *pBest*.

### *4.2.4.7 Fungsi perbarui kecepatan dan posisi partikel*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk memperbarui kecepatan dan posisi partikel dinamakan dengan fungsi UpdateVelocityPosition. Implementasi fungsi memperbarui kecepatan dan posisi partikel ditunjukkan pada *Source Code* 4.11.

```
1   static object[,] UpdateVelocityPosition(int lengthBin, int
2   lengthReal, int popSize, object[,] krom, object[,] pBest, object[,]
3   pGest, double rangeMin, double rangeMax, double gamma1, double
4   gamma2)
5   {
6           object[,] vOld = new object[popSize / 2, (lengthBin +
7           lengthReal)];
8           object[,] pOld = new object[popSize / 2, (lengthBin +
9           lengthReal)];
10          object[,] vNew = new object[popSize / 2, (lengthBin +
11          lengthReal)];
12          object[,] pNew = new object[popSize / 2, (lengthBin +
13          lengthReal)];
14          object[,] sigmoid = new object[popSize / 2, lengthBin];
15
16          Random r = new Random();
17          double random1 = r.NextDouble();
18          double random2 = r.NextDouble();
19          double randomInertia = r.NextDouble();
```

```
20          double[,] randomBinary = new double[popSize/2, lengthBin];
21          double inertiaWeight = 0.5 + (randomInertia / 2);
22          double rangeVMaxReal = 0.6 * rangeMax;
23          double rangeVMinReal = -1 * rangeVMaxReal;
24          double rangeVMaxBinary = 0.6 * 1;
25          double rangeVMinBinary = -1 * rangeVMaxBinary;
26
27          for (int x = 0; x <= popSize - 1; x++)
28          {
29              if (x <= (popSize / 2) - 1)
30              {
31                  for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
32                  {
33                      vOld[x, z] = krom[x, z];
34                  }
35              }
36
37              else
38              {
39                  for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
40                  {
41                      pOld[x - (popSize / 2), z] = krom[x, z];
42                  }
43              }
44          }
45
46          for (int j = 0; j <= (popSize/2) - 1; j++)
47          {
48              for (int k = 0; k <= (lengthBin) - 1; k++)
49              {
50                   randomBinary[j, k] = r.NextDouble();
51              }
52          }
53
54          for (int y = 0; y <= (popSize / 2) - 1; y++)
55          {
56              for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
57              {
58                 vNew[y, z] = (inertiaWeight * Convert.ToDouble(vOld[y,
59                 z])) + (gamma1 * random1 * (Convert.ToDouble(pBest[y,
60                 z]) - Convert.ToDouble(pOld[y, z]))) + (gamma2 *
61                 random2 * (Convert.ToDouble(vOld[0, z]) -
62                 Convert.ToDouble(pOld[y, z])));
63
64                 if (z < lengthBin)
65                 {
66                     if (Convert.ToDouble(vNew[y, z]) >
67                     Convert.ToDouble(rangeVMaxBinary))
68                     {
69                        vNew[y, z] = Convert.ToDouble(rangeVMaxBinary);
70                     }
71                     else if (Convert.ToDouble(vNew[y, z]) <
72                     Convert.ToDouble(rangeVMinBinary))
73                     {
74                       vNew[y, z] = Convert.ToDouble(rangeVMinBinary);
75                     }
76
77                     sigmoid[y, z] = 1 / (1 + Math.Exp(-
78                    (Convert.ToDouble(vNew[y, z]))));
79
80                    if (randomBinary[y, z] >=
81                     Convert.ToDouble(sigmoid[y, z]))
```

```
82                  {
83                    pNew[y, z] = 0;
84                  }
85                  else
86                  {
87                    pNew[y, z] = 1;
88                  }
89              }
90
91              else
92              {
93                  if (Convert.ToDouble(vNew[y, z]) >
94                  Convert.ToDouble(rangeVMaxReal))
95                  {
96                     vNew[y, z] = Convert.ToDouble(rangeVMaxReal);
97                  }
98                  else if (Convert.ToDouble(vNew[y, z]) <
99                  Convert.ToDouble(rangeVMinReal))
100                 {
101                   vNew[y, z] = Convert.ToDouble(rangeVMinReal);
102                 }
103
104                 pNew[y,   z]   =   Convert.ToDouble(pOld[y,   z])   +
105                 Convert.ToDouble(vNew[y, z]);
106
107                 if       (Convert.ToDouble(pNew[y,       z])       >
108                 Convert.ToDouble(rangeMax))
109                 {
110                   pNew[y, z] = Convert.ToDouble(rangeMax);
111                 }
112                 else    if    (Convert.ToDouble(pNew[y,    z])    <
113                 Convert.ToDouble(rangeMin))
114                 {
115                   pNew[y, z] = Convert.ToDouble(rangeMin);
116                 }
117             }
118         }
119     }
120
121     return pNew;
122 }
```

***Source Code* 4.11 Implementasi proses perbarui kecepatan dan posisi partikel**

Penjelasan dari *Source Code* 4.11 adalah sebagai berikut :

1. Baris 6-25 merupakan proses untuk deklarasi objek dan *array* serta inisialisasi variabel.
2. Baris 27-43 merupakan proses untuk pembentukan kecepatan partikel awal dengan menggunakan kromosom yang memiliki nilai *fitness* tertinggi dan pembentukan posisi partikel awal dengan menggunakan kromosom yang memiliki nilai *fitness* terendah.
3. Baris 46-52 merupakan proses pembangkitan nilai acak yang akan digunakan untuk perbandingan dalam menghitung posisi baru pada partikel pengkodean nilai biner.
4. Baris 58-62 merupakan proses perhitungan kecepatan baru untuk partikel yang dikodekan dalam nilai real dan biner.

5. Baris 66-75 merupakan proses pengecekan agar kecepatan baru partikel yang dikodekan dalam nilai biner tidak keluar dari rentang minimal dan maksimalnya.
6. Baris 77-88 merupakan proses perhitungan nilai fungsi *sigmoid* dan posisi baru untuk partikel yang dikodekan dalam nilai biner.
7. Baris 93-102 merupakan proses pengecekan agar kecepatan baru partikel yang dikodekan dalam nilai real tidak keluar dari rentang minimal dan maksimalnya.
8. Baris 104-116 merupakan proses perhitungan nilai posisi baru untuk partikel yang dikodekan dalam nilai real.
9. Baris 121 merupakan proses pengembalian posisi partikel baru yaitu objek pNew.

#### *4.2.4.8 Fungsi seleksi*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan seleksi kromosom dinamakan dengan fungsi Seleksi. Implementasi fungsi seleksi kromosom dengan metode *elitism* ini ditunjukkan pada *Source Code* 4.12.

```
1   static object[,] Seleksi(int lengthBin, int lengthReal, int popSize,
2   int popSizeCross, int popSizeMut, object[,] krom, object[,]
3   kromCross, object[,] kromMut, object[,] pNew, double[] fitness,
4   double[] fitnessCross, double[] fitnessMut, double[] fitnesspNew, int
5   totalParentOffspring)
6   {
7         object[,] kromAll = new object[totalParentOffspring,
8         (lengthBin + lengthReal)];
9         double[] fitnessAll = new double[totalParentOffspring];
10
11        for (int x = 0; x <= totalParentOffspring - 1; x++)
12        {
13              if (x <= popSizeCross - 1)
14              {
15                 for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
16                 {
17                       kromAll[x, z] = kromCross[x, z];
18                 }
19                 fitnessAll[x] = fitnessCross[x];
20              }
21
22              else if (x <= (popSizeCross + popSizeMut) - 1)
23              {
24                 for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
25                 {
26                       kromAll[x, z] = kromMut[(x - popSizeCross), z];
27                 }
28                 fitnessAll[x] = fitnessMut[x - popSizeCross];
29              }
30
31              else if (x <= (popSizeCross + popSizeMut + (popSize / 2))
32              - 1)
33              {
34                 for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
35                 {
```

```
36                      kromAll[x, z] = pNew[(x - popSizeCross -
37                      popSizeMut), z];
38                  }
39                  fitnessAll[x] = fitnesspNew[x - popSizeCross -
40                   popSizeMut];
41              }
42
43              else
44              {
45                  for (int z = 0; z <= (lengthBin + lengthReal) - 1; z++)
46                  {
47                      kromAll[x, z] = krom[(x - popSizeCross -
48                      popSizeMut - (popSize / 2)), z];
49                  }
50                  fitnessAll[x] = fitness[x - popSizeCross - popSizeMut
51                   - (popSize / 2)];
52              }
53          }
54
55          kromAll = Perangkingan(lengthBin, lengthReal,
56          totalParentOffspring, kromAll, fitnessAll);
57
58          for (int v = 0; v <= popSize - 1; v++)
59          {
60              for (int w = 0; w <= (lengthBin + lengthReal) - 1; w++)
61              {
62                  krom[v, w] = kromAll[v, w];
63              }
64
65              fitness[v] = fitnessAll[v];
66          }
67
68          return krom;
69      }
```

***Source Code* 4.12 Implementasi proses seleksi**

Penjelasan dari *Source Code* 4.12 adalah sebagai berikut :

1. Baris 7-9 merupakan proses deklarasi *array* dan objek.
2. Baris 11-53 merupakan proses penggabungan kromosom awal dengan kromosom hasil *offspring* metode GA dan PSO ke dalam objek bernama kromAll. Baris 13-20 merupakan proses penyalinan kromosom hasil metode GA proses tukar silang ke kromAll. Baris 22-29 merupakan proses penyalinan kromosom hasil metode GA proses mutasi ke kromAll. Baris 31-41 merupakan proses penyalinan kromosom hasil metode PSO ke kromAll. Sedangkan baris 43-52 merupakan proses penyalinan kromosom awal ke kromAll.
3. Baris 55-56 merupakan proses perangkingan kromAll berdasarkan nilai *fitness*.
4. Baris 58-66 merupakan proses penyalinan kromAll yang telah diurutkan sesuai nilai *fitness* ke objek krom awal sejumlah populasi awal.
5. Baris 68 merupakan proses pengembalian kromosom hasil seleksi yaitu objek krom.

### 4.2.5 Fungsi peramalan curah hujan dengan RBF

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan peramalan curah hujan dengan metode RBF dinamakan dengan fungsi PeramalanRBF. Implementasi fungsi peramalan curah hujan dengan metode RBF ditunjukkan pada *Source Code* 4.13.

```
1   static double[] PeramalanRBF(int inpNodes, int hidNodes, object[,]
2   krom, double[,] cHujanInput, int barisData, int pickedKromIndex, int
3   rangeMin, int rangeMax)
4   {
5           int indexCenter = hidNodes * inpNodes;
6           int indexRadius = indexCenter + (hidNodes * inpNodes);
7           int indexWeight = indexRadius + hidNodes;
8           int z = pickedKromIndex;
9           double[] act = new double[hidNodes];
10          double[] gaussFunc = new double[hidNodes];
11          double[] cHujanPrediksi = new double[barisData];
12
13          for (int p = 0; p <= barisData - 1; p++)
14          {
15              double tempPred = 0;
16
17              for (int w = 0; w <= hidNodes - 1; w++)
18              {
19                  double tempAct = 0;
20
21                  for (int v = 0; v <= inpNodes - 1; v++)
22                  {
23                      int iterasiRBF = w + (v * hidNodes);
24
25                      if (Convert.ToInt32(krom[z, iterasiRBF]) != 0)
26                      {
27                          tempAct = tempAct + Math.Pow((cHujanInput[v,
28                           p] - Convert.ToDouble(krom[z, indexCenter +
29                           iterasiRBF])), 2);
30                      }
31                  }
32
33                  act[w] = Math.Sqrt(tempAct);
34
35                  if (act[w] != 0)
36                  {
37                      gaussFunc[w]      =      Math.Exp(-((act[w])      /
38                      Math.Pow(Convert.ToDouble(krom[z,  indexRadius  +
39                      w]), 2)));
40                  }
41                  else
42                  {
43                      gaussFunc[w] = 0;
44                  }
45
46                  tempPred     =     tempPred     +     (gaussFunc[w]     *
47                  Convert.ToDouble(krom[z, indexWeight + w]));
48              }
49
50          if (Convert.ToDouble(tempPred) > Convert.ToDouble(rangeMax))
51          {
52                  tempPred = Convert.ToDouble(rangeMax);
53          }
```

```
54          else if (Convert.ToDouble(tempPred) <
55          Convert.ToDouble(rangeMin))
56          {
57                  tempPred = Convert.ToDouble(rangeMin);
58          }
59
60          cHujanPrediksi[p] = tempPred;
61        }
62
63        return cHujanPrediksi;
64  }
```

***Source Code* 4.13 Implementasi proses peramalan curah hujan dengan RBF**

Penjelasan dari *Source Code* 4.13 adalah sebagai berikut :

1. Baris 4-10 merupakan proses deklarasi *array* dan inisialisasi variabel.
2. Baris 13-61 merupakan proses peramalan curah hujan RBF sebanyak baris data curah hujan yang dimasukkan.
3. Baris 21-33 merupakan proses perhitungan *Euclidean norm* antara vektor masukan dengan titik pusat vektor masukan.
4. Baris 35-44 merupakan proses perhitungan fungsi *radial basis* dari tiap *node* tersembunyi.
5. Baris 46-47 merupakan proses perhitungan nilai keluaran (prediksi) dari jaringan RBF.
6. Baris 50-58 merupakan proses pengecekan hasil nilai keluaran jaringan RBF agar tetap berada pada *range* minimal dan maksimal yang telah ditetapkan.
7. Baris 60 merupakan proses penyalinan perhitungan nilai keluaran jaringan RBF ke *array* cHujanPrediksi.
8. Baris 63 merupakan proses pengembalian hasil prediksi curah hujan yaitu *array* cHujanPrediksi.

### 4.2.6 Fungsi evaluasi hasil peramalan curah hujan dengan RBF-HPSOGA

Pada evaluasi peramalan curah hujan terdapat proses denormalisasi data dan perhitungan evaluasi. Berikut adalah penjelasan implementasi program untuk kedua proses tersebut.

#### *4.2.6.1 Fungsi denormalisasi data*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan denormalisasi data curah hujan dinamakan dengan fungsi Denormalisasi. Implementasi fungsi denormalisasi data curah hujan ditunjukkan pada S*ource Code* 4.14.

```
1   static double[] Denormalisasi(double[] cHujanNorm, int jumlahData,
2   double Min, double Max, int rangeMin, int rangeMax)
3   {
4           double[] cHujanDenorm = new double[jumlahData];
5
6           for (int x = 0; x <= jumlahData - 1; x++)
7           {
8               cHujanDenorm[x] = Math.Round((((cHujanNorm[x] - rangeMin)
9               * (Max - Min)) / (rangeMax - rangeMin)) + Min);
10          }
11
12          return cHujanDenorm;
13  }
```

***Source Code* 4.14 Implementasi proses denormalisasi data**

Penjelasan dari *Source Code* 4.14 adalah sebagai berikut :

1. Baris 4 merupakan proses deklarasi *array*.
2. Baris 6-10 merupakan proses untuk menghitung denormalisasi data curah hujan sejumlah baris data curah hujan ternormalisasi yang dimasukkan.
3. Baris 12 merupakan proses pengembalian data curah hujan yang telah didenormalisasi yaitu *array* cHujanDenorm.

### *4.2.6.2 Fungsi evaluasi*

Pada implementasi program di penelitian ini fungsi yang digunakan untuk melakukan perhitungan evaluasi dinamakan dengan fungsi Evaluasi. Implementasi fungsi perhitungan evaluasi dengan empat macam statistik standar yaitu MAE, MAPE, RMSE, dan CC ditunjukkan pada *Source Code* 4.15.

```
1   static double[] Evaluasi(int barisDataUji, double[] cHujanUjiAktual,
2   double[] cHujanUjiPrediksi, double Min, double Max, int rangeMin,
3   int    rangeMax,    double[]    cHujanUjiPrediksiDenorm,    double[]
4   cHujanUjiAktualDenorm)
5   {
6           cHujanUjiPrediksiDenorm   =   Denormalisasi(cHujanUjiPrediksi,
7           barisDataUji, Min, Max, rangeMin, rangeMax);
8
9           cHujanUjiAktualDenorm     =     Denormalisasi(cHujanUjiAktual,
10          barisDataUji, Min, Max, rangeMin, rangeMax);
11
12          double sigmaAktual = 0;
13          double sigmaPrediksi = 0;
14          double sigmaAktualKuadrat = 0;
15          double sigmaPrediksiKuadrat = 0;
16          double sigmaAktualPrediksi = 0;
17          double tempMAE = 0;
18          double tempRMSE = 0;
19          double tempMAPE = 0;
20          double c = 0.0001;
21          double errorMAPE;
22
23          for (int x = 0; x <= barisDataUji - 1; x++)
24          {
25            tempMAE = Math.Abs(cHujanUjiPrediksiDenorm[x] -
26            cHujanUjiAktualDenorm[x]) + tempMAE;
```

```
27              tempRMSE = Math.Pow((cHujanUjiPrediksiDenorm[x] -
28              cHujanUjiAktualDenorm[x]), 2) + tempRMSE;
29              errorMAPE = (Math.Abs(((cHujanUjiPrediksiDenorm[x] + c) -
30              (cHujanUjiAktualDenorm[x] + c)) / (cHujanUjiAktualDenorm[x]
31              + c)) * 100);
32
33              if (errorMAPE > 100)
34              {
35                  errorMAPE = 100;
36              }
37
38              tempMAPE = errorMAPE + tempMAPE;
39              sigmaAktual = cHujanUjiAktualDenorm[x] + sigmaAktual;
40              sigmaPrediksi = cHujanUjiPrediksiDenorm[x] + sigmaPrediksi;
41               sigmaAktualKuadrat = Math.Pow(cHujanUjiAktualDenorm[x], 2)
42              + sigmaAktualKuadrat;
43              sigmaPrediksiKuadrat = Math.Pow(cHujanUjiPrediksiDenorm[x],
44              2) + sigmaPrediksiKuadrat;
45               sigmaAktualPrediksi = (cHujanUjiPrediksiDenorm[x] *
46               cHujanUjiAktualDenorm[x]) + sigmaAktualPrediksi;
47             }
48
49            double MAE = tempMAE / barisDataUji;
50            double RMSE = Math.Sqrt(tempRMSE / barisDataUji);
51            double CC = ((barisDataUji * sigmaAktualPrediksi) -
52             (sigmaAktual * sigmaPrediksi)) / Math.Sqrt(((barisDataUji *
53            sigmaAktualKuadrat - Math.Pow(sigmaAktual, 2)) *
54             (barisDataUji * sigmaPrediksiKuadrat -
55            Math.Pow(sigmaPrediksi, 2))));
56            double MAPE = tempMAPE / barisDataUji;
57
58            double[] hasilEvaluasi = new double[4] {MAE, RMSE, CC,MAPE};
59            return hasilEvaluasi;
60  }
```

***Source Code* 4.15 Implementasi proses evaluasi**

Penjelasan dari *Source Code* 4.15 adalah sebagai berikut :

1. Baris 6-10 merupakan proses denormalisasi terhadap curah hujan uji aktual dan curah hujan uji prediksi.
2. Baris 12-18 merupakan proses inisialisasi variabel.
3. Baris 20-47 merupakan proses perhitungan nilai evaluasi MAE, MAPE, RMSE, dan nilai CC.
4. Baris 58 merupakan proses penyalinan nilai-nilai evaluasi ke dalam *array* hasilEvaluasi.
5. Baris 59 merupakan proses pengembalian hasil perhitungan nilai evaluasi peramalan curah hujan yaitu *array* hasilEvaluasi.

## 4.3 Implementasi antarmuka

Antarmuka sistem optimasi *Radial Basis Function Neural Network* menggunakan *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan dibuat untuk mempermudah pengguna berinteraksi

dengan sistem perangkat lunak. Antarmuka perangkat lunak terdiri dari beberapa halaman utama yaitu halaman *input* parameter, halaman normalisasi dan tabulasi, halaman optimasi RBF dengan HPSOGA, halaman peramalan dengan RBF, dan halaman evaluasi. Berikut hasil implementasi antarmuka pada masing-masing halaman pada sistem ini.

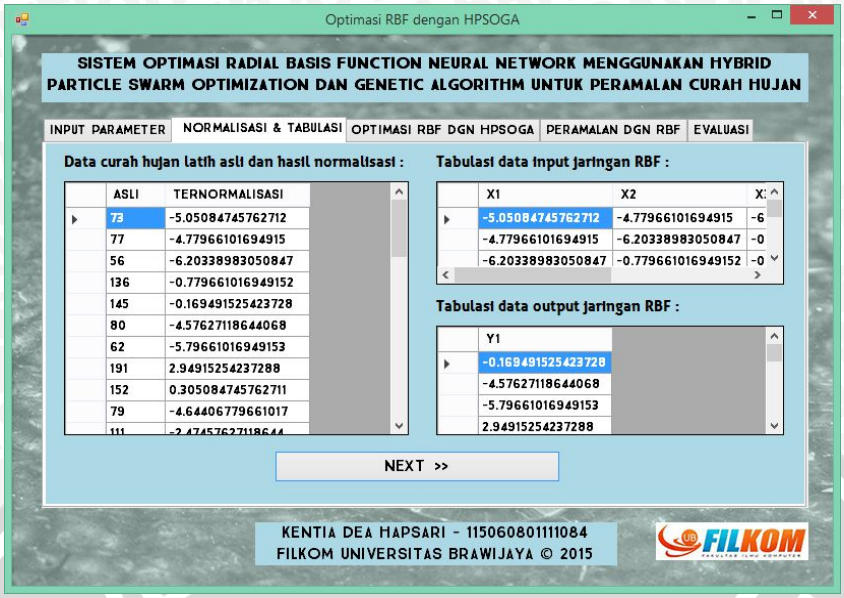
### 4.3.1 Tampilan halaman *input* parameter

Halaman ini merupakan halaman awal ketika pengguna masuk ke dalam sistem. Pertama-tama pengguna harus memasukkan dua data curah hujan sebagai data latih dan data uji dengan menekan *button* LOAD lalu memilih *file* bertipe .xls yang diinginkan. Kemudian pengguna memasukkan parameter-parameter baik dalam metode RBF, GA, maupun PSO yang dibutuhkan oleh sistem untuk melakukan peramalan curah hujan. Setelah itu pengguna dapat menekan *button* NEXT untuk melihat hasil perhitungan proses normalisasi dan tabulasi data curah hujan pada halaman selanjutnya. Gambar 4.1 menunjukkan antarmuka halaman *input* parameter sistem.


**Gambar 4.1 Antarmuka halaman *input* parameter**

### 4.3.2 Tampilan halaman normalisasi dan tabulasi

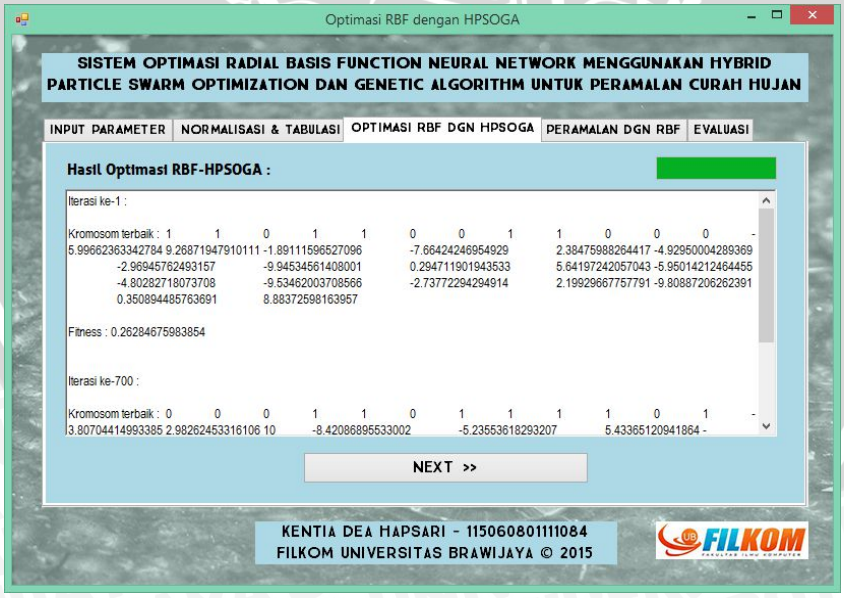
Halaman ini akan menampilkan hasil perhitungan proses normalisasi dan tabulasi data jaringan RBF baik *input* maupun *output* pada *data grid view*. Pengguna dapat melanjutkan ke proses selanjutnya yaitu proses optimasi RBF dengan HPSOGA dengan menekan *button* NEXT. Gambar 4.2 menunjukkan antarmuka halaman normalisasi dan tabulasi.




**Gambar 4.2 Antarmuka halaman normalisasi dan tabulasi**

### 4.3.3 Tampilan halaman optimasi RBF dengan HPSOGA

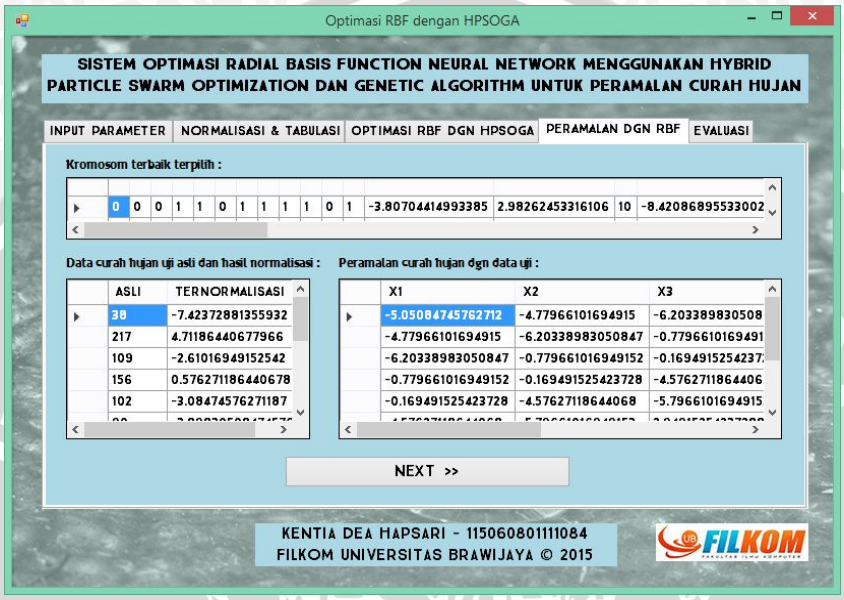
Halaman ini akan menampilkan kromosom terbaik hasil optimasi RBF-HPSOGA beserta nilai *fitness*-nya untuk iterasi pertama dan terakhir pada *rich text box*. Pengguna dapat melanjutkan ke proses selanjutnya yaitu proses peramalan curah hujan dengan RBF menggunakan kromosom terbaik sebagai parameter-parameternya dengan menekan *button* NEXT. Gambar 4.3 menunjukkan antarmuka halaman optimasi RBF dengan HPSOGA.




**Gambar 4.3 Antarmuka halaman optimasi RBF dengan HPSOGA**

### 4.3.4 Tampilan halaman peramalan dengan RBF

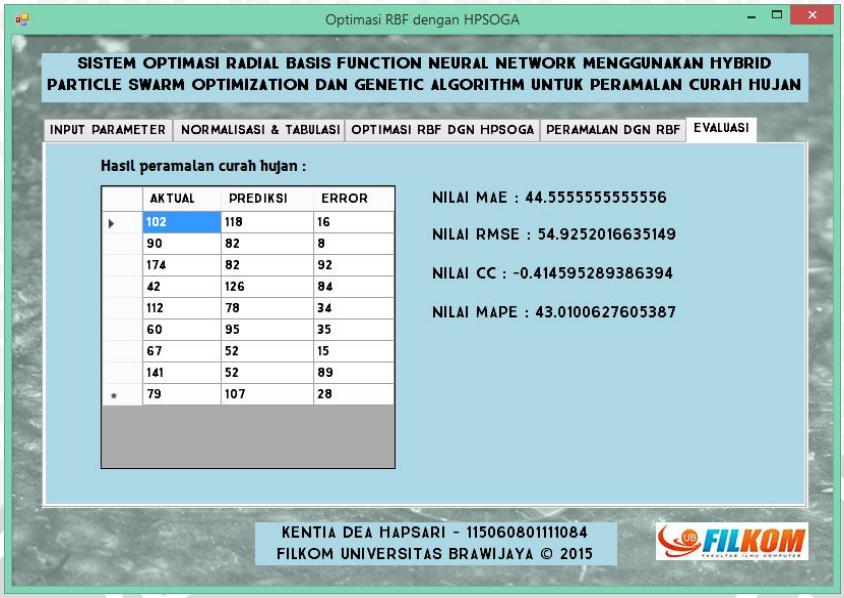
Halaman ini akan menampilkan kromosom terbaik hasil optimasi RBF-HPSOGA, hail normalisasi data curah hujan uji, serta hasil peramalan data uji dengan RBF menggunakan kromosom terbaik sebagai parameter-parameternya pada *data grid view*. Pengguna dapat melanjutkan ke proses selanjutnya yaitu proses perhitungan nilai evaluasi dengan menekan *button* NEXT. Gambar 4.4 menunjukkan antarmuka halaman peramalan dengan RBF.




**Gambar 4.4 Antarmuka halaman peramalan dengan RBF**

### 4.3.5 Tampilan halaman evaluasi

Halaman ini akan menampilkan hasil peramalan data uji dengan RBF beserta nilai *error* hasil peramalannya pada *data grid view*. Selain itu akan ditampilkan juga perhitungan nilai evaluasi statistik standar yaitu *Mean Absolute Error* (MAE), *Mean Absolute Percentage Error* (MAPE), *Root Mean Square Error* (RMSE), dan *Correlation Coefficient* (CC) untuk melihat performa dari model peramalan yang dibuat. Gambar 4.5 menunjukkan antarmuka halaman evaluasi.




**Gambar 4.5 Antarmuka halaman evaluasi**

# BAB 5 PENGUJIAN DAN ANALISIS

Bab pengujian dan analisis membahas tentang pengujian dan analisis terhadap sistem yang telah diimplementasikan sebelumnya. Bab ini berisi empat sub bab, yaitu pengujian, hasil dan analisis hasil pengujian parameter, hasil dan analisis hasil pengujian data, serta analisis global keseluruhan pengujian.

## 5.1 Pengujian

Pada sub bab ini akan dijelaskan mengenai pengujian sistem. Pengujian pada penelitian ini dilakukan berdasarkan skenario sebagai berikut:

- Pengujian penentuan jumlah *node* masukan yang optimal.
- Pengujian penentuan jumlah *node* tersembunyi yang optimal.
- Pengujian penentuan jumlah populasi yang optimal.
- Pengujian penentuan jumlah iterasi yang optimal.
- Pengujian kombinasi probabilitas tukar silang (*pc*) dan probabilitas mutasi (*pm*) yang optimal.
- Pengujian penentuan nilai *delta* (δ) tukar silang yang optimal
- Pengujian penentuan nilai learning rate (*γ1* dan *γ2*) yang optimal.
- Pengujian peramalan data uji menggunakan data latih dengan parameter yang optimal
- Pengujian penentuan jumlah data latih yang optimal.

## 5.2 Hasil dan analisis pengujian parameter

Pengujian parameter pada penelitian ini terdiri dari pengujian jumlah *node* tersembunyi sampai dengan pengujian nilai *learning rate* yang akan dijelaskan dalam sub bab 5.2.1 sampai dengan sub bab 5.2.7. Data latih yang digunakan pada pengujian ini adalah data curah hujan dasarian tahun 2011-2012 daerah Karangploso Kabupaten Malang yang ada pada Lampiran. Tolak ukur dalam pengujian ini adalah nilai rata-rata *fitness* sehingga nilai parameter paling optimal didapatkan dari pengujian dengan nilai rata-rata *fitness* terbaik.

### 5.2.1 Hasil dan analisis pengujian jumlah *node* masukan yang optimal

Pengujian jumlah *node* masukan digunakan untuk mengetahui jumlah *node* masukan jaringan RBF yang dibutuhkan untuk mendapatkan hasil peramalan curah hujan yang paling akurat. Jumlah *node* yang akan diuji adalah 1 sampai dengan 10, sedangkan untuk nilai parameter lain yang digunakan pada pengujian ini adalah sebagai berikut:

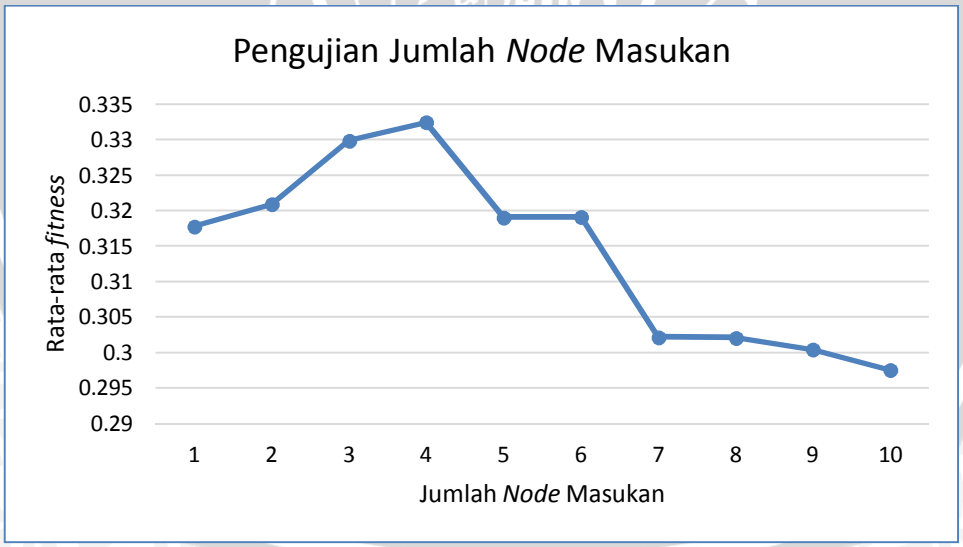
a. Jumlah *node* tersembunyi = 5
b. Jumlah populasi = 20
c. Jumlah iterasi = 100

d. Probabilitas tukar silang = 0.8
e. Probabilitas mutasi = 0.2
f. Nilai *delta* ($\delta$) tukar silang = 0.25
g. *Learning rate* ($\gamma 1$ dan $\gamma 2$) = 2

Untuk setiap satu variasi nilai *node* masukan dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Hasil pengujian jumlah *node* masukan dapat dilihat pada Tabel 5.1.

**Tabel 5.1 Hasil pengujian jumlah *node* masukan**

| Jumlah *node* masukan | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke-*i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1 | 0.3252 | 0.3384 | 0.2832 | 0.3242 | 0.3095 | 0.3021 | 0.3370 | 0.3009 | 0.3473 | 0.3102 | 0.31781 |
| 2 | 0.3308 | 0.3434 | 0.3132 | 0.3373 | 0.3305 | 0.3271 | 0.3209 | 0.3124 | 0.2837 | 0.3100 | 0.3209262 |
| 3 | 0.3446 | 0.3151 | 0.3586 | 0.3109 | 0.3300 | 0.3160 | 0.3278 | 0.3419 | 0.3301 | 0.3241 | 0.3299207 |
| 4 | 0.3385 | 0.3375 | 0.3604 | 0.3542 | 0.3307 | 0.3161 | 0.3397 | 0.3515 | 0.2669 | 0.3292 | 0.3324729 |
| 5 | 0.3099 | 0.2911 | 0.3299 | 0.3014 | 0.3037 | 0.3169 | 0.3245 | 0.3505 | 0.3372 | 0.3259 | 0.3190984 |
| 6 | 0.3272 | 0.2987 | 0.3117 | 0.3337 | 0.3293 | 0.3428 | 0.3392 | 0.2917 | 0.3089 | 0.3080 | 0.3191104 |
| 7 | 0.3013 | 0.3266 | 0.3054 | 0.2907 | 0.2887 | 0.2619 | 0.2872 | 0.3365 | 0.3348 | 0.2889 | 0.3021886 |
| 8 | 0.3246 | 0.3067 | 0.3161 | 0.2745 | 0.2878 | 0.3153 | 0.2991 | 0.2781 | 0.3165 | 0.3023 | 0.3021079 |
| 9 | 0.2899 | 0.3158 | 0.2537 | 0.2982 | 0.2853 | 0.3198 | 0.3162 | 0.2858 | 0.2852 | 0.3546 | 0.300456 |
| 10 | 0.3673 | 0.2993 | 0.2941 | 0.2881 | 0.2807 | 0.3045 | 0.2925 | 0.3014 | 0.2877 | 0.2602 | 0.2975767 |




**Gambar 5.1 Grafik pengujian jumlah *node* masukan**

Berdasarkan grafik pada Gambar 5.1, dapat dilihat bahwa jumlah *node* masukan berpengaruh terhadap nilai *fitness*. Nilai rata-rata *fitness* cenderung

meningkat dari jumlah *node* masukan 1 sampai dengan 4, kemudian menurun pada jumlah *node* masukan di atas 4. Dari hal ini dapat dikatakan bahwa jumlah *node* masukan sebesar 4 merupakan batas pola optimal yang dapat memberikan hasil keluaran jaringan yang paling baik, dimana pola ini berulang untuk setiap 4 data dari dasarian yang sebelumnya. Oleh karena itu jumlah *node* masukan jaringan RBF optimum adalah 4 dengan rata-rata nilai *fitness* sebesar 0.33247

### 5.2.2 Hasil dan analisis pengujian jumlah *node* tersembunyi yang optimal

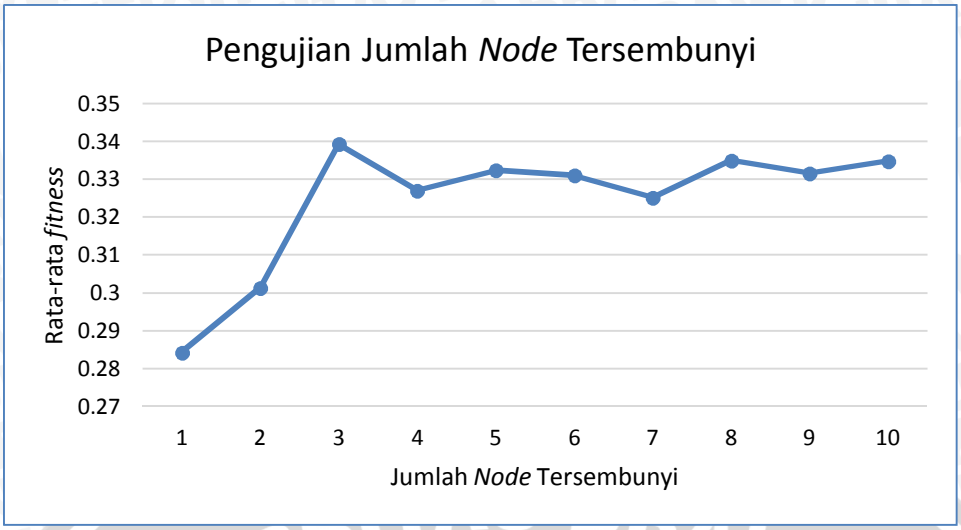
Pengujian jumlah *node* tersembunyi digunakan untuk mengetahui jumlah *node* tersembunyi jaringan RBF yang dibutuhkan untuk mendapatkan hasil peramalan curah hujan yang paling akurat. Jumlah *node* yang akan diuji adalah 1 sampai dengan 10, sedangkan untuk nilai parameter lain yang digunakan pada pengujian ini adalah sebagai berikut:

a. Jumlah *node* masukan = 4
b. Jumlah populasi = 20
c. Jumlah iterasi = 100
d. Probabilitas tukar silang = 0.8
e. Probabilitas mutasi = 0.2
f. Nilai *delta* ($\delta$) tukar silang = 0.25
g. *Learning rate* ($\gamma 1$ dan $\gamma 2$) = 2

Untuk setiap satu variasi nilai *node* tersembunyi dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Hasil pengujian jumlah *node* tersembunyi dapat dilihat pada Tabel 5.2.

**Tabel 5.2 Hasil pengujian jumlah *node* tersembunyi**

| Jumlah *node* tersembunyi | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke-*i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1 | 0.2825 | 0.3017 | 0.2964 | 0.2942 | 0.2974 | 0.2838 | 0.2740 | 0.2480 | 0.2607 | 0.3042 | 0.2842926 |
| 2 | 0.2747 | 0.2496 | 0.3211 | 0.3162 | 0.2937 | 0.3041 | 0.3070 | 0.3298 | 0.2867 | 0.3313 | 0.3014103 |
| 3 | 0.3423 | 0.3246 | 0.3440 | 0.3563 | 0.3230 | 0.3368 | 0.3329 | 0.3343 | 0.3489 | 0.3504 | 0.3393531 |
| 4 | 0.3538 | 0.2997 | 0.3382 | 0.3548 | 0.3083 | 0.3276 | 0.2937 | 0.3282 | 0.3332 | 0.3340 | 0.3271464 |
| 5 | 0.3385 | 0.3375 | 0.3604 | 0.3542 | 0.3307 | 0.3161 | 0.3397 | 0.3515 | 0.2669 | 0.3292 | 0.3324729 |
| 6 | 0.3237 | 0.3342 | 0.3351 | 0.3308 | 0.3336 | 0.3139 | 0.3345 | 0.3391 | 0.3294 | 0.3368 | 0.3310954 |
| 7 | 0.3218 | 0.3210 | 0.3278 | 0.3286 | 0.3315 | 0.3334 | 0.3330 | 0.3251 | 0.2853 | 0.3451 | 0.3252521 |
| 8 | 0.3441 | 0.3315 | 0.3291 | 0.3238 | 0.3293 | 0.3384 | 0.3409 | 0.3203 | 0.3436 | 0.3499 | 0.335091 |
| 9 | 0.3294 | 0.3449 | 0.3351 | 0.3182 | 0.3229 | 0.3447 | 0.3150 | 0.3373 | 0.3275 | 0.3416 | 0.3316599 |
| 10 | 0.3419 | 0.3367 | 0.3411 | 0.3232 | 0.3233 | 0.3382 | 0.3484 | 0.3318 | 0.3335 | 0.3306 | 0.3348696 |




**Gambar 5.2 Grafik pengujian jumlah *node* tersembunyi**

Berdasarkan grafik pada Gambar 5.2, dapat dilihat bahwa jumlah *node* tersembunyi berpengaruh terhadap nilai *fitness*. Nilai rata-rata *fitness* cenderung meningkat dari jumlah *node* tersembunyi 1 sampai dengan 3. Hal ini sesuai dengan *rule-of-thumb* pemilihan jumlah *node* tersembunyi yang banyak digunakan pada penelitian dengan topik jaringan syaraf tiruan, antara lain:

- Jumlah *node* tersembunyi adalah 2/3 dari jumlah *node* masukan ditambah dengan *node* keluaran (Boger dan Guterman, 1997).
- Jumlah *node* tersembunyi harus kurang dari dua kali dari jumlah *node* masukan (Berry dan Linoff, 2004).
- Jumlah *node* tersembunyi adalah diantara jumlah *node* masukan dan jumlah *node* keluaran (Blum, 1992).

Dimana 3 adalah benar 2/3 dari 4 (jumlah *node* masukan) ditambah 1 (jumlah *node* keluaran), kurang dari 8 (dua kali jumlah *node* masukan), dan berada pada rentang antara 4 dan 1. Nilai rata-rata *fitness* cenderung menurun pada jumlah *node* tersembunyi di atas 3 karena semakin besar jumlah *node* tersembunyi menyebabkan semakin banyak variasi arsitektur jaringan RBF pada kromosom. Hali ini memungkinkan proses pembangkitan populasi awal memiliki arsitektur jaringan RBF yang jauh dari solusi optimal. Oleh karena itu jumlah *node* tersembunyi jaringan RBF optimum adalah 4 dengan rata-rata nilai *fitness* sebesar 0.3394.

### 5.2.3 Hasil dan analisis pengujian jumlah populasi yang optimal

Pengujian jumlah populasi digunakan untuk mengetahui jumlah populasi yang tepat untuk menghasilkan parameter jaringan RBF yang paling optimal. Jumlah populasi yang akan diuji adalah kelipatan 40 yaitu populasi 40 sampai dengan 400,

sedangkan untuk nilai parameter lain yang digunakan pada pengujian ini adalah sebagai berikut:

a. Jumlah *node* masukan = 4
b. Jumlah *node* tersembunyi = 3
c. Jumlah iterasi = 100
d. Probabilitas tukar silang = 0.8
e. Probabilitas mutasi = 0.2
f. Nilai *delta* ($\delta$) tukar silang = 0.25
g. *Learning rate* ($\gamma 1$ dan $\gamma 2$) = 2

Untuk setiap satu variasi populasi dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Hasil pengujian jumlah populasi dapat dilihat pada Tabel 5.3.

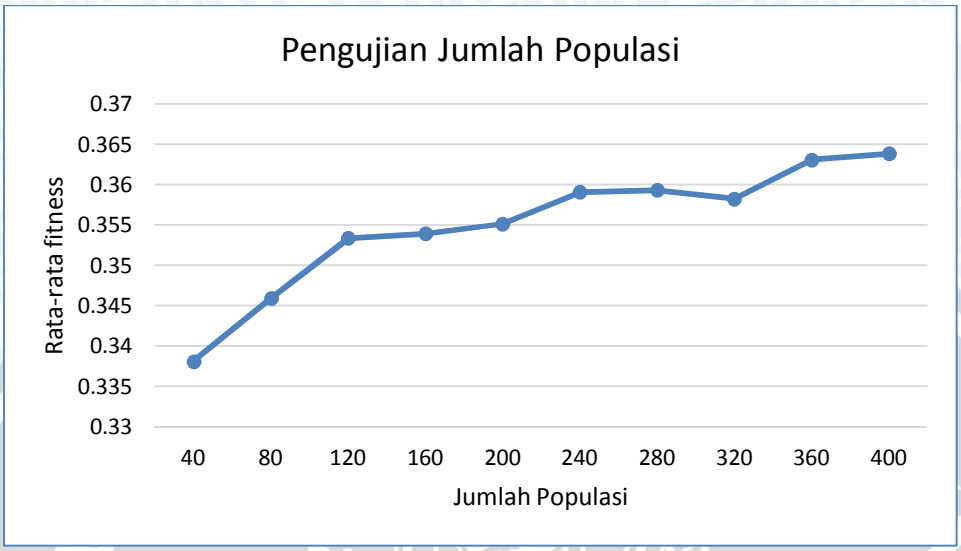
**Tabel 5.3 Hasil pengujian jumlah populasi**

| Jumlah populasi | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke-*i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 40 | 0.3474 | 0.3423 | 0.3375 | 0.3298 | 0.3311 | 0.3202 | 0.3215 | 0.3472 | 0.3592 | 0.3448 | 0.3381066 |
| 80 | 0.3520 | 0.3296 | 0.3312 | 0.3340 | 0.3443 | 0.3323 | 0.3730 | 0.3518 | 0.3565 | 0.3544 | 0.345901 |
| 120 | 0.3379 | 0.3546 | 0.3665 | 0.3747 | 0.3344 | 0.3571 | 0.3589 | 0.3638 | 0.3510 | 0.3347 | 0.3533526 |
| 160 | 0.3536 | 0.3403 | 0.3583 | 0.3623 | 0.3423 | 0.3607 | 0.3454 | 0.3580 | 0.3536 | 0.3650 | 0.3539391 |
| 200 | 0.3290 | 0.3601 | 0.3589 | 0.3431 | 0.3648 | 0.3594 | 0.3627 | 0.3700 | 0.3581 | 0.3452 | 0.3551397 |
| 240 | 0.3640 | 0.3695 | 0.3581 | 0.3651 | 0.3439 | 0.3601 | 0.3579 | 0.3756 | 0.3561 | 0.3407 | 0.3590914 |
| 280 | 0.3609 | 0.3648 | 0.3609 | 0.3541 | 0.3439 | 0.3788 | 0.3683 | 0.3623 | 0.3386 | 0.3606 | 0.3593169 |
| 320 | 0.3480 | 0.3282 | 0.3610 | 0.3630 | 0.3638 | 0.3683 | 0.3660 | 0.3633 | 0.3600 | 0.3610 | 0.3582574 |
| 360 | 0.3503 | 0.3588 | 0.3595 | 0.3687 | 0.3580 | 0.3744 | 0.3673 | 0.3688 | 0.3626 | 0.3625 | 0.3630791 |
| 400 | 0.3711 | 0.3572 | 0.3626 | 0.3654 | 0.3424 | 0.3687 | 0.3578 | 0.3792 | 0.3608 | 0.3733 | 0.3638527 |

Berdasarkan grafik pada Gambar 5.3, dapat dilihat bahwa jumlah populasi berpengaruh terhadap nilai *fitness*. Semakin besar jumlah populasi yang dibangkitkan maka semakin besar pula nilai rata-rata *fitness* yang dihasilkan. Hal ini dikarenakan jumlah populasi yang besar akan meningkatkan kemampuan eksplorasi algoritma genetika dan PSO sehingga memungkinkan untuk menghasilkan variasi individu yang beragam.

Nilai *fitness* semakin meningkat dari jumlah populasi 40 sampai 360, namun pada jumlah populasi 360 ke atas tidak terjadi peningkatan nilai secara signifikan. Hal ini menunjukkan bahwa ukuran populasi yang besar juga tidak menjamin dapat menghasilkan nilai *fitness* yang lebih baik, karena dengan populasi yang besar memungkinkan proses pembangkitan populasi awal memiliki variasi sebaran populasi yang jauh dari solusi optimal. Selain itu, jumlah populasi yang terlalu besar akan membuat proses komputasi menjadi lebih lama. Sedangkan jumlah

populasi yang terlalu sedikit mengakibatkan area eksplorasi algoritma semakin sempit, sehingga mengakibatkan variasi individu tidak beragam dan solusi yang dihasilkan tidak terlalu baik. Oleh karena itu rekomendasi jumlah populasi optimum dengan waktu komputasi yang efisien adalah 360 dengan rata-rata nilai *fitness* sebesar 0.3631.




**Gambar 5.3 Grafik pengujian jumlah populasi**

### 5.2.4 Hasil dan analisis pengujian jumlah iterasi yang optimal

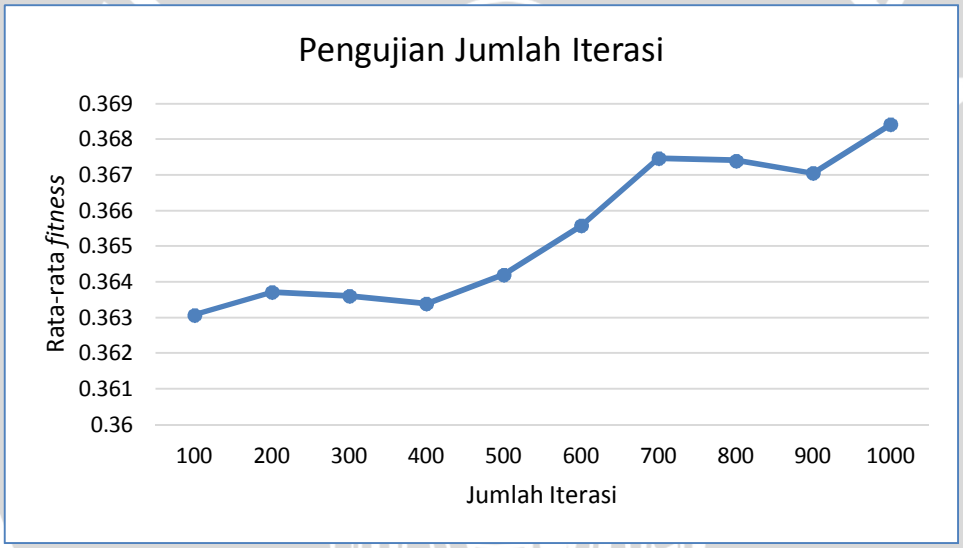
Pengujian jumlah iterasi digunakan untuk mengetahui jumlah iterasi yang tepat untuk menghasilkan parameter jaringan RBF yang paling optimal. Jumlah populasi yang akan diuji adalah kelipatan 100 yaitu iterasi 100 sampai dengan 1000, sedangkan untuk nilai parameter lain yang digunakan pada pengujian ini adalah sebagai berikut:

a. Jumlah *node* masukan = 4
b. Jumlah *node* tersembunyi = 3
c. Jumlah populasi = 360
d. Probabilitas tukar silang = 0.8
e. Probabilitas mutasi = 0.2
f. Nilai *delta* ($\delta$) tukar silang = 0.25
g. *Learning rate* ($\gamma 1$ dan $\gamma 2$) = 2

Untuk setiap satu variasi iterasi dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Hasil pengujian jumlah iterasi dapat dilihat pada Tabel 5.4.

**Tabel 5.4 Hasil pengujian jumlah iterasi**

| Jumlah iterasi | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke-*i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 100 | 0.3503 | 0.3588 | 0.3595 | 0.3687 | 0.3580 | 0.3744 | 0.3673 | 0.3688 | 0.3626 | 0.3625 | 0.3630791 |
| 200 | 0.3599 | 0.3577 | 0.3643 | 0.3738 | 0.3532 | 0.3646 | 0.3678 | 0.3668 | 0.3670 | 0.3622 | 0.3637207 |
| 300 | 0.3678 | 0.3503 | 0.3698 | 0.3440 | 0.3745 | 0.3710 | 0.3723 | 0.3647 | 0.3605 | 0.3613 | 0.3636142 |
| 400 | 0.3609 | 0.3658 | 0.3608 | 0.3681 | 0.3656 | 0.3673 | 0.3707 | 0.3482 | 0.3717 | 0.3549 | 0.3633923 |
| 500 | 0.3450 | 0.3708 | 0.3691 | 0.3620 | 0.3688 | 0.3678 | 0.3604 | 0.3617 | 0.3667 | 0.3697 | 0.3642093 |
| 600 | 0.3718 | 0.3655 | 0.3739 | 0.3714 | 0.3502 | 0.3698 | 0.3651 | 0.3719 | 0.3624 | 0.3538 | 0.3655876 |
| 700 | 0.3702 | 0.3749 | 0.3675 | 0.3728 | 0.3672 | 0.3624 | 0.3628 | 0.3655 | 0.3626 | 0.3688 | 0.367469 |
| 800 | 0.3589 | 0.3634 | 0.3669 | 0.3653 | 0.3629 | 0.3680 | 0.3650 | 0.3703 | 0.3755 | 0.3778 | 0.3674057 |
| 900 | 0.3644 | 0.3603 | 0.3614 | 0.3731 | 0.3599 | 0.3680 | 0.3660 | 0.3713 | 0.3791 | 0.3672 | 0.3670625 |
| 1000 | 0.3631 | 0.3696 | 0.3697 | 0.3687 | 0.3623 | 0.3608 | 0.3713 | 0.3640 | 0.3617 | 0.3931 | 0.368423 |




**Gambar 5.4 Grafik pengujian jumlah iterasi**

Berdasarkan grafik pada Gambar 5.4, dapat dilihat bahwa jumlah iterasi berpengaruh terhadap nilai *fitness*. Semakin besar jumlah iterasi sistem maka semakin besar pula nilai rata-rata *fitness* yang dihasilkan. Hal ini dikarenakan jumlah iterasi yang besar akan mengakibatkan proses evolusi baik pada GA maupun PSO semakin sering dilakukan. Hal tersebut akan berpengaruh terhadap individu yang dihasilkan dan berpengaruh juga terhadap variasi nilai *fitness*. Namun pada jumlah iterasi 700 tidak terjadi peningkatan nilai *fitness* secara signifikan dan dapat dikatakan bahwa pengujian mencapai konvergen. Jumlah iterasi yang terlalu besar juga dapat membuat waktu komputasi menjadi lebih

lama. Oleh karena itu rekomendasi jumlah iterasi optimum adalah 700 dengan rata-rata nilai *fitness* sebesar 0.3674.

### 5.2.5 Hasil dan analisis pengujian kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*)

Pengujian jumlah iterasi digunakan untuk mengetahui kombinasi *pc* dan *pm* terbaik untuk mendapatkan parameter jaringan RBF yang paling optimal. Nilai *pc* dan *pm* yang akan diuji adalah berkisar dari 0 sampai 1 dan keduanya jika dijumlahkan menghasilkan nilai 1. Untuk nilai parameter lain yang digunakan pada pengujian ini adalah sebagai berikut:

a. Jumlah *node* masukan = 4
b. Jumlah *node* tersembunyi = 3
c. Jumlah populasi = 360
d. Jumlah iterasi = 700
e. Nilai *delta* ($\delta$) tukar silang = 0.25
f. *Learning rate* ($\gamma 1$ dan $\gamma 2$) = 2

Untuk setiap satu kombinasi dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Hasil pengujian kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*) dapat dilihat pada Tabel 5.5.

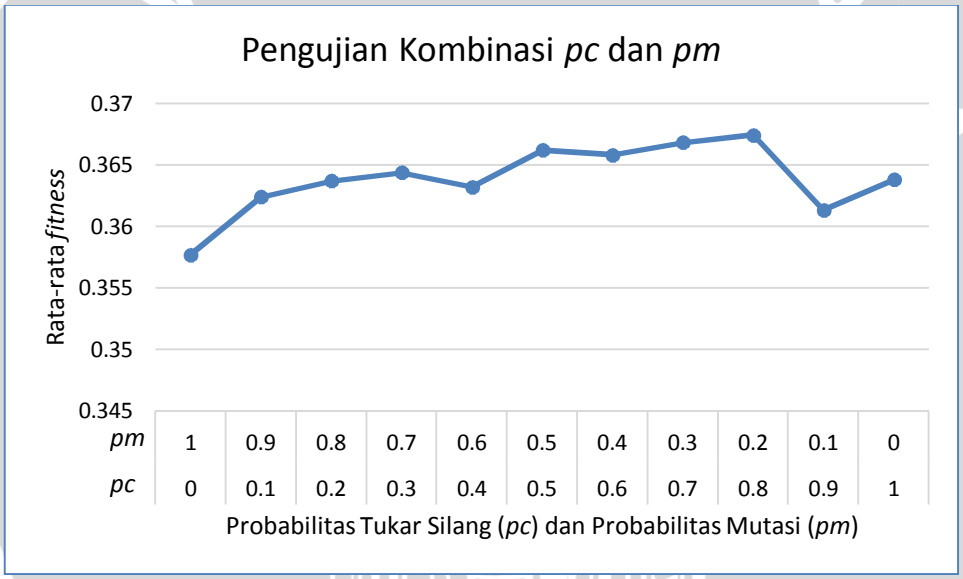
**Tabel 5.5 Hasil pengujian kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*)**

| Kombinasi | | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Percobaan ke- *i* | | | | | | | | | | |
| *pc* | *pm* | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 0 | 1 | 0.3579 | 0.3561 | 0.3584 | 0.3554 | 0.3564 | 0.3566 | 0.3571 | 0.3621 | 0.3563 | 0.3609 | 0.3577195 |
| 0.1 | 0.9 | 0.3620 | 0.3690 | 0.3574 | 0.3614 | 0.3591 | 0.3673 | 0.3643 | 0.3621 | 0.3600 | 0.3618 | 0.3624411 |
| 0.2 | 0.8 | 0.3642 | 0.3723 | 0.3789 | 0.3639 | 0.3624 | 0.3442 | 0.3623 | 0.3653 | 0.3614 | 0.3624 | 0.3637283 |
| 0.3 | 0.7 | 0.3654 | 0.3691 | 0.3705 | 0.3572 | 0.3753 | 0.3661 | 0.3743 | 0.3608 | 0.3620 | 0.3434 | 0.3644097 |
| 0.4 | 0.6 | 0.3690 | 0.3660 | 0.3608 | 0.3577 | 0.3619 | 0.3662 | 0.3545 | 0.3638 | 0.3682 | 0.3642 | 0.3632264 |
| 0.5 | 0.5 | 0.3626 | 0.3591 | 0.3530 | 0.3618 | 0.3607 | 0.3669 | 0.3650 | 0.3669 | 0.3917 | 0.3747 | 0.3662306 |
| 0.6 | 0.4 | 0.3594 | 0.3688 | 0.3651 | 0.3647 | 0.3667 | 0.3693 | 0.3616 | 0.3613 | 0.3637 | 0.3781 | 0.3658514 |
| 0.7 | 0.3 | 0.3971 | 0.3458 | 0.3720 | 0.3748 | 0.3677 | 0.3663 | 0.3617 | 0.3621 | 0.3691 | 0.3519 | 0.3668548 |
| 0.8 | 0.2 | 0.3702 | 0.3749 | 0.3675 | 0.3728 | 0.3672 | 0.3624 | 0.3628 | 0.3655 | 0.3626 | 0.3688 | 0.367469 |
| 0.9 | 0.1 | 0.3602 | 0.3613 | 0.3427 | 0.3652 | 0.3665 | 0.3633 | 0.3629 | 0.3591 | 0.3627 | 0.3699 | 0.3613627 |
| 1 | 0 | 0.3710 | 0.3591 | 0.3686 | 0.3668 | 0.3674 | 0.3668 | 0.3645 | 0.3689 | 0.3590 | 0.3464 | 0.3638408 |

Berdasarkan grafik pada Gambar 5.5, dapat dilihat bahwa nilai *fitness* semakin meningkat dari kombinasi *pc pm* bernilai [0,1] sampai dengan [0.8, 0.2]. Oleh karena itu, dapat dikatakan semakin besar nilai *pc* dan semakin kecil nilai *pm* akan

menyebabkan semakin besar pula nilai *fitness*. Hal ini dikarenakan nilai *pc* yang tinggi dan *pm* yang rendah memungkinkan algoritma genetika mempunyai level eksploitasi yang tinggi dan dapat belajar dari generasi sebelumnya secara efektif (Mahmudy, Marian, dan Luoang, 2013), sehingga peluang untuk mendapatkan individu selanjutnya dengan nilai *fitness* yang lebih baik dari individu awalnya akan semakin besar.

Namun pada saat kombinasi *pc pm* bernilai [0.9, 0.1] dan [1,0] tidak terjadi peningkatan nilai *fitness* secara signifikan. Hal ini menunjukkan bahwa nilai *pm* yang semakin kecil juga tidak menjamin dapat menghasilkan nilai *fitness* yang lebih baik, karena algoritma genetika tetap membutuhkan proses mutasi untuk dapat mengeksplorasi area lain dalam ruang pencarian dan tidak terjebak dalam area optimum lokal. Maka dari itu itu rekomendasi kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*) optimum adalah 0.8 dan 0.2 dengan rata-rata nilai *fitness* sebesar 0.3675.




**Gambar 5.5 Grafik pengujian kombinasi probabilitas tukar silang (*pc*) dan mutasi (*pm*)**

### 5.2.6 Hasil dan analisis pengujian nilai *delta* (*δ*) tukar silang

Pengujian nilai *delta* (*δ*) tukar silang dilakukan untuk mengetahui rentang nilai acak yang yang tepat pada metode *extended intermediate crossover* untuk menghasilkan parameter jaringan RBF yang paling optimal. Nilai *delta* (*δ*) tukar silang yang akan diuji adalah berkisar dari 0 sampai 0.85. Untuk nilai parameter lain yang digunakan pada pengujian ini adalah sebagai berikut:

a. Jumlah *node* masukan = 4
b. Jumlah *node* tersembunyi = 3

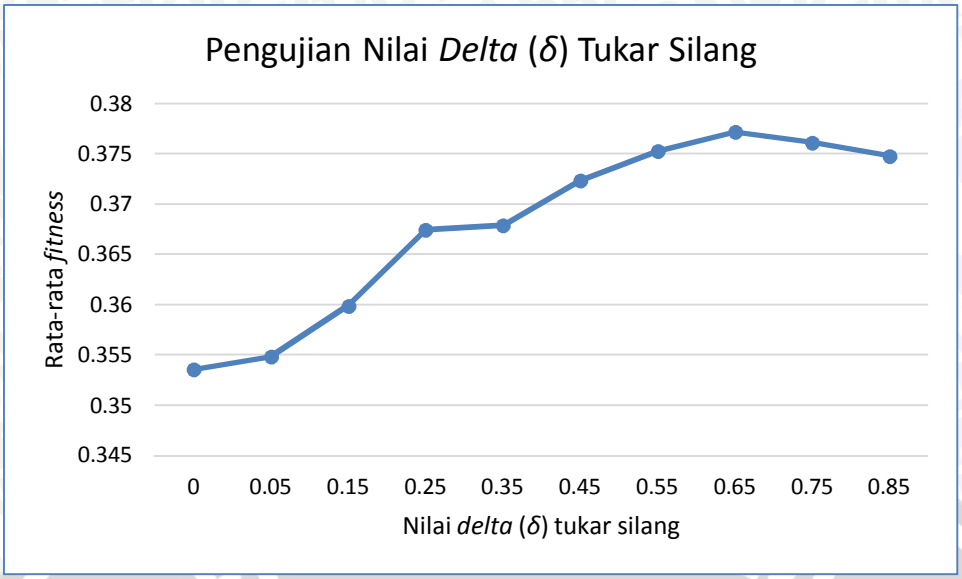
c. Jumlah populasi = 360
d. Jumlah iterasi = 700
e. Probabilitas tukar silang = 0.8
f. Probabilitas mutasi = 0.2
g. *Learning rate* (*γ1* dan *γ2*) = 2

Untuk setiap satu kombinasi dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Nilai *fitness* terbaik yang ditemukan pada setiap percobaan kemudian akan dihitung rata-ratanya untuk mengetahui kombinasi paling optimal. Hasil pengujian nilai *delta* (*δ*) tukar silang dapat dilihat pada Tabel 5.6.

**Tabel 5.6 Hasil pengujian nilai *delta* (*δ*) tukar silang**

| Nilai *delta* (*δ*) | Nilai *fitness* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 0 | 0.3548 | 0.3474 | 0.3601 | 0.3549 | 0.3703 | 0.3478 | 0.3465 | 0.3551 | 0.3585 | 0.3405 | 0.3535789 |
| 0.05 | 0.3588 | 0.3537 | 0.3655 | 0.3541 | 0.3616 | 0.3584 | 0.3442 | 0.3541 | 0.3507 | 0.3473 | 0.3548462 |
| 0.15 | 0.3615 | 0.3567 | 0.3622 | 0.3583 | 0.3649 | 0.3578 | 0.3595 | 0.3604 | 0.3529 | 0.3651 | 0.3599254 |
| 0.25 | 0.3702 | 0.3749 | 0.3675 | 0.3728 | 0.3672 | 0.3624 | 0.3628 | 0.3655 | 0.3626 | 0.3688 | 0.367469 |
| 0.35 | 0.3727 | 0.3721 | 0.3907 | 0.3635 | 0.3521 | 0.3687 | 0.3763 | 0.3729 | 0.3480 | 0.3621 | 0.367918 |
| 0.45 | 0.3914 | 0.3624 | 0.4009 | 0.3459 | 0.3731 | 0.3674 | 0.3691 | 0.3710 | 0.3767 | 0.3659 | 0.372376 |
| 0.55 | 0.3651 | 0.4038 | 0.3888 | 0.3823 | 0.3748 | 0.3766 | 0.3735 | 0.3721 | 0.3612 | 0.3546 | 0.3752847 |
| 0.65 | 0.3764 | 0.3745 | 0.3764 | 0.3723 | 0.3960 | 0.3758 | 0.3784 | 0.3720 | 0.3781 | 0.3723 | 0.3772195 |
| 0.75 | 0.3712 | 0.3761 | 0.4023 | 0.3765 | 0.3725 | 0.3756 | 0.3722 | 0.3740 | 0.3761 | 0.3652 | 0.376163 |
| 0.85 | 0.3659 | 0.3860 | 0.3687 | 0.3765 | 0.3758 | 0.3710 | 0.3856 | 0.3814 | 0.3646 | 0.3728 | 0.3748338 |

Berdasarkan grafik pada Gambar 5.7, dapat dilihat bahwa nilai *delta* (*δ*) tukar silang berpengaruh terhadap nilai *fitness*. Semakin besar nilai *delta* (*δ*) tukar silang maka semakin besar pula nilai rata-rata *fitness* yang dihasilkan. Hal ini dikarenakan semakin besar nilai *delta* (*δ*) tukar silang (*δ* >0) akan meningkatkan peluang *offspring* dapat dihasilkan di luar cakupan *parent*-nya (Muñoz, 2002) dan menyebabkan *offspring* yang dihasilkan lebih beragam. Namun pada nilai *delta* (*δ*) 0.65 ke atas terjadi penurunan nilai *fitness*, yang menunjukkan bahwa nilai *delta* yang besar juga tidak menjamin dapat menghasilkan nilai *fitness* yang lebih baik. Dengan lebih besarnya cakupan hasil *offspring* metode tukar silang, kemungkinan tukar silang menghasilkan *offspring* yang sangat tidak mirip dengan *parent* menjadi lebih besar dan proses semakin jauh dari solusi optimal.




**Gambar 5.6 Grafik pengujian nilai *delta* (*δ*) tukar silang**

### 5.2.7 Hasil dan analisis pengujian kombinasi *learning rate* (*γ1* dan *γ2*)

Pengujian kombinasi *learning rate* digunakan untuk mengetahui nilai *learning rate* (*γ1* dan *γ2*) pada metode PSO yang tepat untuk menghasilkan parameter jaringan RBF yang paling optimal. Nilai *learning rate* yang akan diuji adalah berkisar dari 1 sampai 2. Untuk nilai parameter lain yang digunakan pada pengujian ini adalah sebagai berikut:

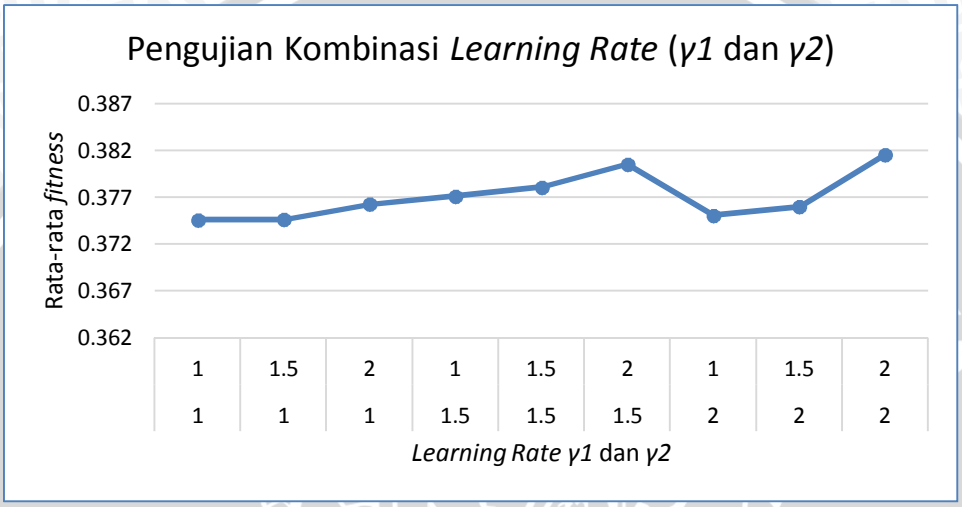
a. Jumlah *node* masukan = 4
b. Jumlah *node* tersembunyi = 3
c. Jumlah populasi = 360
d. Jumlah iterasi = 700
e. Probabilitas tukar silang = 0.8
f. Probabilitas mutasi = 0.2
g. Nilai *delta* (*δ*) tukar silang = 0.65

Untuk setiap satu kombinasi dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Nilai *fitness* terbaik yang ditemukan pada setiap percobaan kemudian akan dihitung rata-ratanya untuk mengetahui kombinasi paling optimal. Hasil pengujian kombinasi *learning rate* dapat dilihat pada Tabel 5.7.

**Tabel 5.7 Hasil pengujian kombinasi *learning rate* (*γ1* dan *γ2*)**

| Kombinasi | | Nilai *fitness* Percobaan ke- *i* | | | | | | | | | | Rata-rata *fitness* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\gamma_1$ | $\gamma_2$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1 | 1 | 0.3743 | 0.3738 | 0.3763 | 0.3675 | 0.3767 | 0.3653 | 0.3672 | 0.4034 | 0.3676 | 0.3738 | 0.3745727 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.5 | 0.3773 | 0.3785 | 0.3630 | 0.3687 | 0.3745 | 0.3594 | 0.3845 | 0.3606 | 0.4031 | 0.3765 | 0.3746193 |
| 1 | 2 | 0.3802 | 0.3747 | 0.3702 | 0.3755 | 0.3690 | 0.3818 | 0.3856 | 0.3764 | 0.3776 | 0.3715 | 0.376259 |
| 1.5 | 1 | 0.3850 | 0.3751 | 0.3618 | 0.3745 | 0.3759 | 0.3782 | 0.3732 | 0.3723 | 0.4076 | 0.3678 | 0.3771223 |
| 1.5 | 1.5 | 0.3713 | 0.3829 | 0.3690 | 0.3764 | 0.3933 | 0.3766 | 0.3705 | 0.3948 | 0.3721 | 0.3741 | 0.3780854 |
| 1.5 | 2 | 0.3717 | 0.4025 | 0.3856 | 0.3821 | 0.3767 | 0.3708 | 0.3999 | 0.3760 | 0.3725 | 0.3673 | 0.3805225 |
| 2 | 1 | 0.3724 | 0.3847 | 0.3654 | 0.3721 | 0.3795 | 0.3890 | 0.3765 | 0.3669 | 0.3734 | 0.3708 | 0.3750596 |
| 2 | 1.5 | 0.3685 | 0.3735 | 0.3855 | 0.3722 | 0.3667 | 0.3693 | 0.3649 | 0.3787 | 0.3796 | 0.4008 | 0.3759804 |
| 2 | 2 | 0.3789 | 0.4081 | 0.3769 | 0.3744 | 0.4007 | 0.3765 | 0.3681 | 0.3731 | 0.3813 | 0.3777 | 0.3815604 |




**Gambar 5.7 Grafik pengujian kombinasi *learning rate* (*γ1* dan *γ2*)**

Berdasarkan grafik pada Gambar 5.7, dapat dilihat bahwa kombinasi *learning rate* (*γ1* dan *γ2*) berpengaruh terhadap nilai *fitness*. Nilai *fitness* cenderung meningkat apabila salah satu nilai *learning rate* semakin besar mendekati nilai 2, kemudian menurun apabila salah satu nilai *learning rate* semakin kecil mendekati nilai 1. Nilai rata-rata *fitness* terbaik didapat pada saat kombinasi *learning rate* (*γ1* dan *γ2*) sama-sama bernilai 2. Hal ini sesuai dengan penelitian yang dilakukan oleh Ozcan dan Mohan, dimana mereka menganggap dua *learning rate* sebagai satu *learning rate γ* (*γ = γ1 + γ2)* dan menemukan nilai konstan terbaik untuk *learning rate* yaitu *γ1* + *γ2* = 4 dengan *γ1* = *γ2* = 2 (Ozcan dan Mohan, 1999).

Nilai *learning rate* yang sama berarti setiap partikel memberikan kepercayaan yang sama pada swarm dan partikel itu sendiri untuk mengontrol pergerakan partikel pada ruang pencarian (Askarzadeh, 2014). Kemudian nilai *learning rate* yang tinggi dapat memberikan posisi baru yang daerahnya relatif lebih jauh dalam ruang pencarian, yang sering mengarah ke eksplorasi yang lebih global sehingga peluang untuk mendapatkan posisi partikel selanjutnya dengan nilai *fitness* yang tinggi juga semakin besar (Parsopoulos dan Vrahatis, 2010). Oleh karena itu

rekomendasi kombinasi *learning rate* (*γ1* dan *γ2*) optimum adalah 2 dan 2 dengan rata-rata nilai *fitness* sebesar 0.3816.

## 5.3 Hasil dan analisis pengujian data

Pengujian data pada penelitian ini terdiri dari pengujian peramalan data uji dengan data latih dan pengujian data latih yang akan dijelaskan dalam sub bab 5.3.1 sampai dengan sub bab 5.3.2. Parameter yang digunakan pada pengujian ini adalah parameter optimal yang ditemukan pada pengujian di sub bab sebelumnya, yaitu:

a. Jumlah *node* masukan = 4
b. Jumlah *node* tersembunyi = 3
c. Jumlah populasi = 360
d. Jumlah iterasi = 700
e. Probabilitas tukar silang = 0.8
f. Probabilitas mutasi = 0.2
g. Nilai *delta* (*δ*) tukar silang = 0.65
h. *Learning rate* (*γ1* dan *γ2*) = 2

### 5.3.1 Hasil dan analisis pengujian peramalan data uji menggunakan data latih dengan parameter optimal

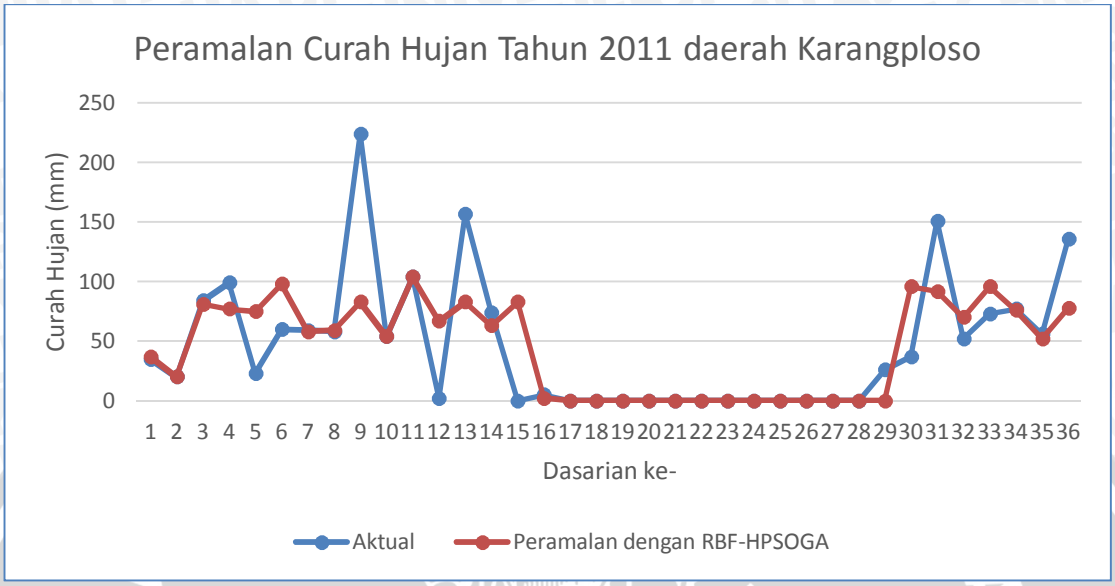
Pengujian peramalan data uji menggunakan data latih dilakukan untuk mengetahui apakah parameter yang dihasilkan pada pengujian sebelumnya dapat meramalkan data itu sendiri dengan baik atau tidak. Data uji yang akan digunakan adalah data curah hujan dasarian bulan Januari - Desember tahun 2011 dan 2012 daerah Karangploso Kabupaten Malang yang ada pada Lampiran. Untuk setiap satu variasi data uji dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Tolak ukur dalam pengujian ini adalah nilai rata-rata *Mean Absolute Error* (MAE) sehingga analisis hasil peramalan didapatkan dari pengujian dengan nilai MAE terbaik. Hasil pengujian peramalan data uji menggunakan data latih dapat dilihat pada Tabel 5.8.

**Tabel 5.8 Hasil pengujian peramalan data uji menggunakan data latih**

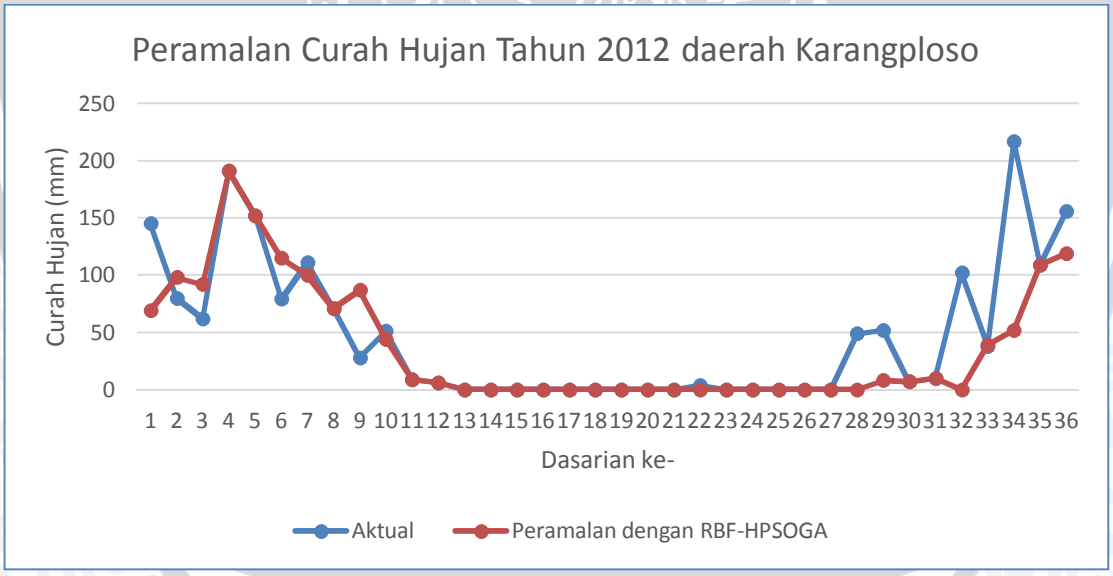
| Data tahun ke- | Nilai MAE | | | | | | | | | | Rata-rata MAE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 2011 | 21.44 | 22.33 | 20.47 | 20.92 | 20.67 | 22.67 | 22.44 | 27.19 | 21.50 | 22.44 | 22.21 |
| 2012 | 17.75 | 18.72 | 21.81 | 19.28 | 20.06 | 18.81 | 22.33 | 20.33 | 21.14 | 18.25 | 19.85 |

Berdasarkan Tabel 5.8, dapat dilihat bahwa nilai parameter yang didapat dari pengujian sebelumnya sudah mampu menghasilkan peramalan curah hujan yang cukup akurat apabila data ujinya disamakan dengan data latih, dengan nilai rata-rata MAE sebesar 22.21 untuk peramalan tahun 2011 dan 19.85 untuk peramalan

tahun 2012. Visualisasi hasil peramalan tahun 2011 dan 2012 dengan menggunakan arsitektur jaringan serta parameter RBF terbaik yang pernah ditemukan proses optimasi ditunjukkan pada Gambar 5.8 dan Gambar 5.9.




**Gambar 5.8 Hasil peramalan curah hujan bulan Januari - Desember tahun 2011 daerah Karangploso Kabupaten Malang**




**Gambar 5.9 Hasil peramalan curah hujan bulan Januari - Desember tahun 2012 daerah Karangploso Kabupaten Malang**

Berdasarkan Gambar 5.8 dapat dilihat bahwa hasil peramalan curah hujan tahun 2011 sudah cukup mendekati nilai aktualnya pada dasarian ke-1 sampai ke-4, sedangkan untuk dasarian di atas 4 mulai terjadi *error* yang cukup besar pada beberapa data peramalan. Untuk hasil peramalan curah hujan tahun 2012 sudah cukup mendekati nilai aktualnya pada dasarian ke-1 sampai ke-27, sedangkan untuk dasarian di atas 27 mulai terjadi *error* yang cukup besar pada beberapa data peramalan. Nilai MAPE yang dihasilkan untuk peramalan tahun 2011 adalah sebesar 26.096 (keakurasian peramalan 73.904%) dan 21.652 untuk peramalan tahun 2012 (keakurasian peramalan 78.348%).

### 5.3.2 Hasil dan analisis pengujian jumlah data latih yang optimal

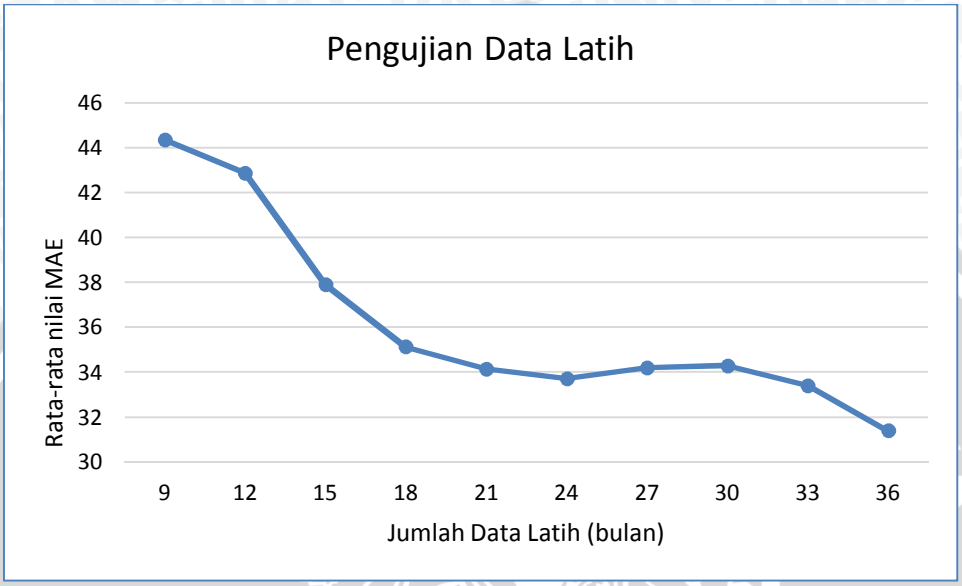
Pengujian jumlah data latih digunakan untuk mengetahui jumlah data latih yang diperlukan untuk mendapatkan hasil peramalan curah hujan yang paling akurat. Data uji yang akan digunakan adalah data curah hujan dasarian bulan Januari - Maret tahun 2013 daerah Karangploso Kabupaten Malang yang ada pada Lampiran. Sedangkan jumlah data latih yang akan diuji adalah kelipatan 3 bulan dimulai dari 9 sampai dengan 36 bulan sebelum bulan Januari 2013. Untuk setiap satu variasi data latih dilakukan sepuluh kali percobaan untuk mendapatkan hasil yang mewakili kemampuan algoritma secara utuh. Tolak ukur dalam pengujian ini adalah nilai rata-rata *Mean Absolute Error* (MAE) sehingga jumlah data latih paling optimal didapatkan dari pengujian dengan nilai rata-rata MAE terbaik. Hasil pengujian jumlah *node* masukan dapat dilihat pada Tabel 5.9.

**Tabel 5.9 Hasil pengujian jumlah data latih**

| Jumlah Data Latih (bulan) | Nilai MAE | | | | | | | | | | Rata-rata Nilai MAE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percobaan ke- *i* | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 9 | 54.89 | 34.00 | 37.44 | 56.11 | 57.89 | 35.22 | 38.78 | 53.33 | 38.56 | 37.44 | 44.366667 |
| 12 | 32.33 | 43.00 | 45.11 | 33.78 | 45.89 | 47.33 | 44.00 | 49.22 | 42.44 | 45.67 | 42.877778 |
| 15 | 33.00 | 35.11 | 33.11 | 52.33 | 40.44 | 32.56 | 33.78 | 35.56 | 42.89 | 40.33 | 37.911111 |
| 18 | 32.33 | 31.89 | 34.33 | 32.11 | 36.22 | 32.67 | 38.33 | 42.56 | 36.33 | 34.56 | 35.133333 |
| 21 | 33.00 | 31.89 | 34.56 | 35.89 | 31.78 | 37.44 | 32.67 | 33.11 | 37.22 | 34.00 | 34.155556 |
| 24 | 34.22 | 42.33 | 33.11 | 31.67 | 33.33 | 32.89 | 34.11 | 32.00 | 31.56 | 32.00 | 33.722222 |
| 27 | 40.22 | 30.67 | 32.44 | 32.78 | 40.56 | 36.22 | 33.89 | 30.67 | 32.00 | 32.67 | 34.211111 |
| 30 | 29.56 | 34.33 | 31.11 | 35.22 | 48.00 | 31.33 | 32.56 | 32.89 | 31.22 | 36.67 | 34.288889 |
| 33 | 34.11 | 35.11 | 32.33 | 30.22 | 34.56 | 34.78 | 34.00 | 33.56 | 30.89 | 34.56 | 33.411111 |
| 36 | 30.89 | 31.89 | 29.78 | 29.89 | 32.89 | 32.89 | 31.00 | 31.78 | 32.78 | 30.22 | 31.4 |

Berdasarkan grafik pada Gambar 5.10, dapat dilihat bahwa jumlah data latih berpengaruh terhadap nilai MAE. Nilai rata-rata *MAE* cenderung meningkat dari jumlah data latih 9 bulan sebelum sampai 36 bulan sebelumnya. Hal ini dikarenakan jumlah data latih yang besar akan secara umum memberikan

representasi dari problem yang ingin dipecahkan dengan lebih baik serta meningkatkan kemungkinan jaringan syaraf tiruan dapat menghasilkan keluaran yang diharapkan (Priddy dan Keller, 2005). Oleh karena itu jumlah data latih optimum adalah 36 bulan sebelum dengan rata-rata nilai MAE sebesar 35.1888. Visualisasi hasil peramalan dengan jumlah data latih paling optimal dapat dilihat pada Lampiran.




**Gambar 5.10 Grafik pengujian jumlah data latih**

## 5.4 Analisis global keseluruhan pengujian

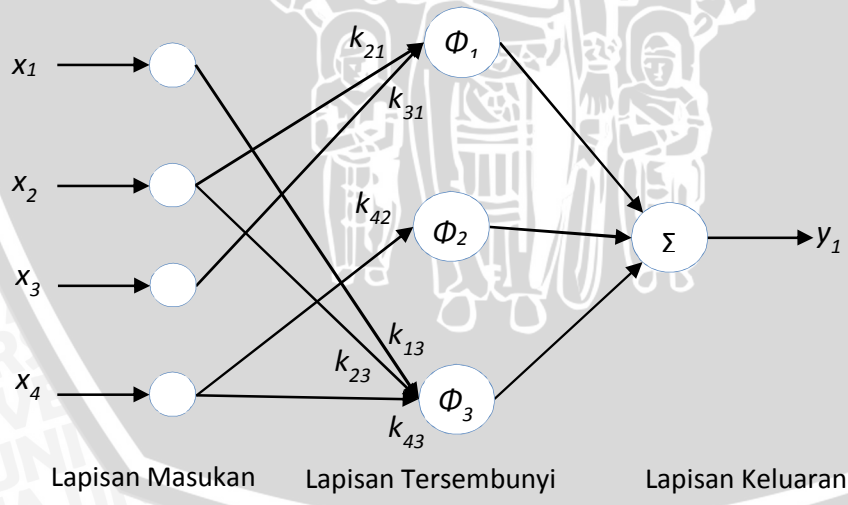
Dari keseluruhan hasil pengujian yang dilakukan pada sistem ditemukan parameter GA, PSO, dan RBF yang optimal yaitu:

a. Jumlah *node* masukan = 4
b. Jumlah *node* tersembunyi = 3
c. Jumlah populasi = 360
d. Jumlah iterasi = 700
e. Probabilitas tukar silang = 0.8
f. Probabilitas mutasi = 0.2
g. Nilai *delta* ($\alpha$) tukar silang = 0.65
h. *Learning rate* ($\gamma 1$ dan $\gamma 2$) = 2
i. Jumlah data latih = 36 bulan

Parameter yang optimal ini kemudian digunakan untuk peramalan curah hujan dengan data latih yaitu data curah hujan daerah Karangploso Kabupaten Malang tahun 2010 sampai 2012 dan data uji yaitu data curah hujan daerah Karangploso Kabupaten Malang tahun 2013. Dari banyak percobaan didapatkan kromosom terbaik sebagai solusi yang dapat memberikan hasil peramalan curah hujan paling akurat, yaitu:

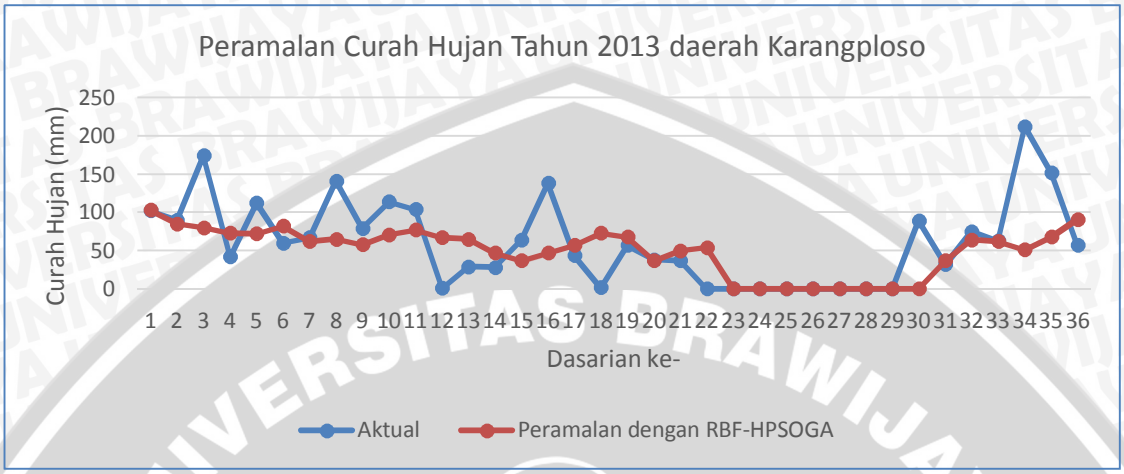
- Koneksi dari *node* masukan ke-1 ke *node* tersembunyi ke-1 ($k_{11}$)= 0
- Koneksi dari *node* masukan ke-1 ke *node* tersembunyi ke-2 ($k_{12}$) = 0
- Koneksi dari *node* masukan ke-1 ke *node* tersembunyi ke-3 ($k_{13}$) = 1
- Koneksi dari *node* masukan ke-2 ke *node* tersembunyi ke-1 ($k_{21}$) = 1
- Koneksi dari *node* masukan ke-2 ke *node* tersembunyi ke-2 ($k_{22}$) = 0
- Koneksi dari *node* masukan ke-2 ke *node* tersembunyi ke-3 ($k_{23}$) = 1
- Koneksi dari *node* masukan ke-3 ke *node* tersembunyi ke-1 ($k_{31}$)= 1
- Koneksi dari *node* masukan ke-3 ke *node* tersembunyi ke-2 ($k_{32}$) = 0
- Koneksi dari *node* masukan ke-3 ke *node* tersembunyi ke-3 ($k_{33}$) = 0
- Koneksi dari *node* masukan ke-4 ke *node* tersembunyi ke-1 ($k_{41}$) = 0
- Koneksi dari *node* masukan ke-4 ke *node* tersembunyi ke-2 ($k_{42}$) = 1
- Koneksi dari *node* masukan ke-4 ke *node* tersembunyi ke-3 ($k_{43}$) = 1
- Vektor titik pusat *node* tersembunyi ke-1 ($cn_1$) = (8.8, -6.22, -1.97, -9.95)
- Vektor titik pusat *node* tersembunyi ke-2 ($cn_2$) = (10, -4.05, -9.99, 1.04)
- Vektor titik pusat *node* tersembunyi ke-3 ($cn_3$) = (0.41, 9.86, -10, -9.99)
- Radius *node* tersembunyi ke-1 ($r_1$) = -5.38
- Radius *node* tersembunyi ke-2 ($r_2$) = 9.99
- Radius *node* tersembunyi ke-3 ($r_3$) = 4.42
- Bobot *node* tersembunyi ke-1 ke *node* keluaran ($\omega_{11}$) = -9.99
- Bobot *node* tersembunyi ke-2 ke *node* keluaran ($\omega_{12}$) = 5.44
- Bobot *node* tersembunyi ke-3 ke *node* keluaran ($\omega_{13}$) = -4.79

Dari penjabaran di atas, maka dapat digambarkan arsitektur jaringan RBF solusi terbaik untuk peramalan curah hujan daerah Karangploso Kabupaten Malang tahun 2013 adalah seperti Gambar 5.11.




**Gambar 5.11 Arsitektur jaringan RBF solusi terbaik untuk peramalan curah hujan daerah Karangploso Malang tahun 2013**

Visualisasi hasil peramalan curah hujan tahun 2013 dengan menggunakan arsitektur jaringan serta parameter RBF terbaik yang pernah ditemukan proses optimasi ditunjukkan pada Gambar 5.12 sedangkan untuk tahun lainnya dapat dilihat pada Lampiran.




**Gambar 5.12 Hasil peramalan curah hujan daerah Karangploso Malang tahun 2013**

Performa model peramalan dengan solusi terbaik untuk nilai evaluasi *Mean Absolute Error* (MAE) adalah sebesar 32, nilai *Root Mean Square Error* (RMSE) sebesar 48.98, nilai *Correlation Coefficient* (CC) sebesar 0.5, dan nilai *Mean Absolute Percentage Error* (MAPE) sebesar 37.325. Melihat nilai MAE dan RMSE yang didapat bisa dikatakan performa model peramalan masih belum baik karena tingkat *error* yang masih termasuk besar apabila dibandingkan dengan rentang data curah hujan yang berkisar dari 0 sampai 295 serta dari jumlah data uji yang hanya sebanyak 36 data. Untuk nilai CC sudah cukup baik karena 0.5 cukup dekat dengan nilai 1.00 yang menunjukkan hubungan antar variabel dalam model peramalan cukup kuat dengan sifat hubungan yang langsung (berbanding lurus). Sedangkan untuk nilai MAPE masih belum terlalu baik karena nilainya masih tinggi dan menyebabkan keakurasian sistem menjadi rendah yaitu 62.67% (dihitung dari 100 – nilai MAPE).

Dilihat dari Gambar 5.12, hasil peramalan curah hujan daerah Karangploso tahun 2013 dengan menggunakan RBF-HPSOGA ini dapat dikatakan masih belum terlalu baik karena masih ada beberapa hasil peramalan yang jauh dengan nilai aktualnya terutama pada data curah hujan aktual yang penurunan atau peningkatannya cukup drastis dari sebelumnya. Hal ini dikarenakan pola data curah hujan tahun 2013 cukup berbeda dengan data curah hujan tiga tahun sebelumnya yang dijadikan data latih, sehingga sistem gagal untuk meramalkan pola data yang baru. Visualisasi hasil peramalan curah hujan pada tahun lain menggunakan solusi terbaik dapat dilihat pada Lampiran.

# BAB 6 KESIMPULAN DAN SARAN

Pada bagian ini memuat kesimpulan yang diperoleh dari implementasi dan pengujian sistem serta saran untuk pengembangan penelitian terkait lebih lanjut.

## 6.1 Kesimpulan

Berdasarkan hasil pengujian maka diperoleh kesimpulan sebagai berikut:

1. Dalam mengimplementasikan metode *Radial Basis Function Neural Network* menggunakan optimasi *Hybrid Particle Swarm Optimization* dan *Genetic Algorithm* untuk peramalan curah hujan, yang harus dilakukan pertama adalah melakukan normalisasi dan tabulasi data latih sesuai dengan jumlah *node* masukan. Setelah itu sistem akan melakukan optimasi yang diawali dengan membangkitkan kromosom sebanyak jumlah populasi yang diinginkan. Masing-masing kromosom merepresentasikan arsitektur, jumlah dari *node* tersembunyi, titik pusat *node* tersembunyi, radius *node* tersembunyi, dan bobot *node* tersembunyi ke *node* keluaran jaringan RBF yang dibangkitkan secara acak. Kemudian akan dihitung nilai *fitness* untuk masing-masing kromosom dan dilakukan perangkingan berdasarkan nilai *fitness*. Dari 4N kromosom yang tersedia (jumlah total populasi yang dibangkitkan), optimasi dengan metode GA dilakukan pada 2N kromosom dengan nilai *fitness* tertinggi, sedangkan optimasi dengan metode PSO dilakukan untuk 2N kromosom dengan nilai *fitness* tertinggi sebagai kecepatan partikel serta 2N kromosom dengan nilai *fitness* terendah sebagai posisi partikel. Hasil *offspring* yang diperoleh dari optimasi metode GA dan PSO kemudian digabungkan dengan kromosom *parent* dan diseleksi menggunakan metode *elitism*. Individu terbaik hasil optimasi ini selanjutnya digunakan sebagai parameter metode RBF untuk melakukan peramalan curah hujan.

2. Parameter paling optimal untuk meramalkan curah hujan yang ditemukan dari hasil pengujian adalah jumlah *node* masukan jaringan RBF sebesar 4, jumlah *node* tersembunyi jaringan RBF sebesar 3, jumlah populasi sebesar 360, jumlah iterasi sebanyak 700, probabilitas tukar silang (*pc*) 0.8, probabilitas mutasi (*pm*) 0.2, nilai *delta* ($\delta$) tukar silang 0.65, nilai *learning rate* ($\gamma 1$ dan $\gamma 2$) yaitu 2, dan jumlah data latih sebesar 36 bulan sebelumnya.

3. Nilai evaluasi dari model peramalan curah hujan menggunakan RBF-HSOGA terbaik yang pernah didapatkan adalah yaitu dengan nilai *Mean Absolute Error* (MAE) sebesar 32, nilai *Root Mean Square Error* (RMSE) sebesar 48.98, nilai *Correlation Coefficient* (CC) sebesar 0.5, dan nilai *Mean Absolute Percentage Error* (MAPE) sebesar 37.325. Keakurasian peramalan dari model peramalan curah hujan terbaik yang pernah didapatkan adalah sebesar 62.675% pada data uji tahun 2013 dengan data latih 36 bulan sebelumnya.

## 6.2 Saran

Saran yang diberikan untuk pengembangan penelitian selanjutnya adalah sebagai berikut:

1. Dapat menambahkan *local search* di dalam metode optimasi PSO dengan batasan iterasi tertentu agar PSO diberikan kesempatan untuk belajar dari iterasi PSO yang sebelumnya secara efektif sehingga bisa meningkatkan keakurasian hasil peramalan curah hujan.
2. Dapat melakukan optimasi metode GA pada kromosom yang berbeda, dimana awalnya metode GA dilakukan pada 2N kromosom dengan nilai *fitness* tertinggi menjadi dilakukan pada 2N kromosom dengan nilai *fitness* terendah. Harapannya adalah dapat meningkatkan keakurasian hasil peramalan curah hujan karena bisa saja perlakuan yang sebelumnya menyebabkan pencarian solusi terjebak pada lokal optimum.

# DAFTAR PUSTAKA


Afantitis, Antreas, Georgia Melagraki, Kalliiopi Makridima, Alex Alexandridis, Haralambos Sarimveis, and Olga Iglessi-Markopoulou. 2005. "Prediction of High Weight Polymers Glass Transition Temperature Using RBF Neural Networks." *Journal of Molecular Structure: THEOCHEM* 716: 193–98.

Askarzadeh, Alireza. 2014. "Comparison of Particle Swarm Optimization and Other Metaheuristics on Electricity Demand Estimation: A Case Study of Iran." *Energy* 72: 484–91.

Autry, Chad W., Thomas J. Goldsby, John E. Bell, and Arthur V. Hill. 2013. *Managing the Global Supply Chain (Collection)*. FT Press.

Badan Meteorologi Klimatologi dan Geofisika. 2015. "Prakiraan Musim Kemarau 2015 Di Indonesia." http://dataweb.bmkg.go.id/cews/pikam/pdf/PMK2015_BMKG_2.pdf.

Badan Penelitian dan Pengembangan Pertanian Kementerian Pertanian. 2015. "Kalender Tanam Terpadu Versi 2.2 Kab. Malang Prov. Jawa Timur Musim Hujan (MH) Oktober 2015 - Maret 2016." http://katam.litbang.pertanian.go.id/katam_terpadu/2015/mh/3/35/3507/3507.pdf.

Berretti, Stefano, Sabu M. Thampi, and Praveen Ranjan Srivastava. 2015. *Intelligent Systems Technologies and Applications Vol. 1*. Advances in Intelligent Systems and Computing 384. Springer International Publishing.

Berry, Michael JA, and Gordon S. Linoff. 2004. *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management*. John Wiley & Sons.

Billings, Steve A., and Guang L. Zheng. 1995. "Radial Basis Function Network Configuration Using Genetic Algorithms." *Neural Networks* 8 (6): 877–90.

Bi, W., G.C. Dandy, and H.R. Maier. 2014. "Improved Genetic Algorithm Optimization of Water Distribution System Design by Incorporating Domain Knowledge." *Environmental Modelling & Software*, 1–12.

Blum, Adam. 1992. *Neural Networks in C++: An Object-Oriented Framework for Building Connectionist Systems*. John Wiley & Sons, Inc.

BMKG Staklim Karangploso Malang. 2015. "Analisis Dinamika Atmosfer Dan Laut Dasarian III Maret 2015 Update 2 April 2015." http://karangploso.jatim.bmkg.go.id/index.php/analisis-kondisi-dinamika-atmosfer-laut-dasarian/158-analisis-kondisi-dinamika-atmosfer-laut-dasarian-tahun-2015/399-analisis-dinamika-atmosfer-dan-laut-dasarian-iii-maret-2015-update-2-april-2015#axzz3X8h9y4fg&gsc.tab=0.

Boger, Zvi, and Hugo Guterman. 1997. "Knowledge Extraction from Artificial Neural Network Models." *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on* 4: 3030–35.

Bonanno, Francesco, Giacomo Capizzi, G. Graditi, C. Napoli, and Giuseppe Marco Tina. 2012. "Bonanno, Francesco, Giacomo Capizzi, G. Graditi, C. Napoli, and Giuseppe Marco Tina. 'A Radial Basis Function Neural Network Based Approach for the Electrical Characteristics Estimation of a Photovoltaic Module.' 97 (2012): 956-961." *Applied Energy* 97: 956–61.

BPS Jatim. 2014. "Provinsi Jawa Timur Dalam Angka 2014." http://jatim.bps.go.id/en/?hal=publikasi_detil&id=57.

Brooks, Chris, and Sotiris Tsolacos. 2010. *Real Estate Modelling and Forecasting*. Cambridge University Press.

Burdsall, Ben, and Christophe Giraud-Carrier. 1998. "GA-RBF: A Self-Optimising RBF Network." *Artificial Neural Nets and Genetic Algorithms: Proceedings of the International Conference in Norwich, U.K., 1997*, 346–49.

Chande, Swati V., and Madhavi Sinha. 2008. "Genetic Algorithm: A Versatile Optimization Tool." *BVICAM'S International Journal of Information Technology*.

Chen, Chiu-Hung, Tung-Kuan Liu, Jyh-Horng Chou, Chung-Hung Tasi, and Hsiu Wang. 2015. "Optimization of Teacher Volunteer Transferring Problems Using Greedy Genetic Algorithms." *Expert Systems with Applications* 42: 668–78.

Chen, Hui-Ling, Da-You Liu, Bo Yang, Jie Liu, Gang Wang, and Su-jing Wang. 2011. "An Adaptive Fuzzy K-Nearest Neighbor Method Based on Parallel Particle Swarm Optimization for Bankruptcy Prediction." *Advances in Knowledge Discovery and Data Mining*, 249–64.

Cura, Tunchan. 2009. "Particle Swarm Optimization Approach to Portfolio Optimization." *Nonlinear Analysis: Real World Applications* 10 (4): 2396–2406.

Dianingtyas, Tiara. 2014. "Akurasi KATAM Masih Rendah." *Sinar Tani*. September 2. http://tabloidsinartani.com/content/read/akurasi-katam-masih-rendah.

Diao, Yinliang, Weinong Sun, Sai Wing Leung, and Kwok Hung Chan. 2015. "Prediction of Magnetic Field Emissions by Current Source Reconstruction Using Radial Basis Function Network." *Electronics Letters* 51 (16): 1243–45.

Duan, Haibin, Qinan Luo, Yuhui Shi, and Guanjun Ma. 2013. "Hybrid Particle Swarm Optimization and Genetic Algorithm for Multi-UAV Formation Reconfiguration." *IEEE Computational Intelligence Magazine*.

Ekasari, Nuraini. 2015. "Mau Tanam? Lihat Katam Versi Baru." *Sinar Tani*. April 2. http://tabloidsinartani.com/content/read/mau-tanam-lihat-katam-versi-baru/.

Gálvez, Akemi, and Andrés Iglesias. 2013. "A New Iterative Mutually Coupled Hybrid GA–PSO Approach for Curve Fitting in Manufacturing." *Applied Soft Computing* 13: 1491–1504.

Giovanis, Eleftherios. 2012. "Study of Discrete Choice Models and Adaptive Neuro-Fuzzy Inference System in the Prediction of Economic Crisis Periods in USA." *Economic Analysis & Policy* 42 (1): 79–95.

Han, Jiawei, Michelin Kamber, and Jian Pei. 2011. *Data Mining: Concepts and Techniques: Concepts and Techniques*. Elsevier.

Ingragustari. 2005a. "Prediksi Curah Hujan Dengan Menggunakan ANFIS." *Lokakarya Nasional Forum Prakiraan, Evaluasi Dan Validasi BMG*.

———. 2005b. "Prediksi Curah Hujan Dengan Menggunakan Transformasi Wavelet." *Prosiding Lokakarya Nasional Forum Prakiraan, Evaluasi Dan Validasi BMG*.

Jackson, Sherri. 2011. *Research Methods and Statistics: A Critical Thinking Approach*. Cengage Learning.

Jain, Yogendra Kumar, and Santosh Kumar Bhandare. 2011. "Min Max Normalization Based Data Perturbation Method for Privacy Protection." *International Journal of Computer & Communication Technology* 2 (8): 45–50.

Lee, Kwang Hyung. 2006. *First Course on Fuzzy Theory and Applications*. Vol. 27. Springer Science & Business Media.

Mahmudy, Wayan F., Romeo M. Marian, and Lee HS Luoang. 2013. "Modeling and Optimization of Part Type Selection and Loading Problem in Flexible Manufacturing System Using Real Coded Genetic Algorithms." *Nternational Journal of Electrical, Electronic Science and Engineering* 7: 181–90.

Mitchell, Melanie. 1998. *An Introduction to Genetic Algorithms*. MIT Press.

Mohammadi, Reza, S.M.T Fatemi Ghomi, and Farzad Zeinali. 2014. "A New Hybrid Evolutionary Based RBF Networks Method for Forecasting Time Series: A Case Study of Forecasting Emergency Supply Demand Time Series." *Engineering Applications of Artificial Intelligence* 36: 204–14.

Morecroft, John. 2007. *Strategic Modelling and Business Dynamics: A Feedback Systems Approach*. John Wiley & Sons.

Muñoz, Lluís A. Belanche. 2002. "Evolutionary Optimization of Heterogeneous Problems." *Parallel Problem Solving from Nature—PPSN VII* 7: 475–84.

Nurcahyo, Septian, Fhira Nhita, and Adiwijaya. 2014. "Rainfall Prediction in Kemayoran Jakarta Using Hybrid Genetic Algorithm (GA) and Partially Connected Feedforward Neural Network (PCFNN)." *2014 2nd International Conference on Information and Communication Technology (ICoICT)*, 166–71.

Nuryadi. 2005. "Validasi Model Prakiraan Jangka Panjang Menggunakan Model ARIMA." *Lokakarya Nasional Forum Prakiraan, Evaluasi Dan Validasi BMG*.

Ozcan, Ender, and Chilukuri K. Mohan. 1999. "Particle Swarm Optimization: Surfing the Waves." *Proceedings of the 1999 Congress on Evolutionary Computation IEEE* 3: 1939–1934.

Parsopoulos, Konstantinos E., and Michael N. Vrahatis. 2010. *Particle Swarm Optimization and Intelligence: Advances and Applications: Advances and Applications*. IGI Global.

Phillips, Keith R., and Jose Joaquin Lopez. 2007. "An Evaluation of Real-Time Forecasting Performance Across 10 Western U.S. States."

Priddy, Kevin L., and Paul E. Keller. 2005. *Artificial Neural Networks: An Introduction*. Vol. 68. SPIE Press.

Roqib, Muhammad. 2015. "Sawah Di Bengawan Solo Panen Dini." http://www.koran-sindo.com/read/985544/151/sawah-di-bengawan-solo-panen-dini-1428289435.

Rubin, Allen. 2012. *Statistics for Evidence-Based Practice and Evaluation*. Cengage Learning.

Sermpinis, Georgios, Konstantinos Theofilatos, Andreas Karathanasopoulos, Efstratios F. Georgopoulos, and Christian Dunis. 2013. "Forecasting Foreign Exchange Rates with Adaptive Neural Networks Using Radial-Basis Functions and Particle Swarm Optimization." *European Journal of Operational Research* 225: 528–40.

Siddhartha, Naveen Sharma, and Varun. 2012. "A Particle Swarm Optimization Algorithm for Optimization of Thermal Performance of a Smooth Flat Plate Solar Air Heater." *Energy* 38 (1): 406–13.

Sideratos, George, and Nikos D. Hatziargyriou. 2007. "An Advanced Statistical Method for Wind Power Forecasting." *Power Systems, IEEE Transactions on* 22 (1): 258–65.

Sukstrienwong, Anon. 2013. "Multi-Criteria Genetic Algorithms for Solving Pig Food Problems." *International Journal on Computer Science and Engineering (IJCSE)* 3 (1): 225–35.

Utomo, Yunanto Wiji. 2014. "BMKG Akui Prakiraan Cuacanya Masih Kurang Akurat." *Kompas*. January 30. http://sains.kompas.com/read/2014/01/30/1628275/BMKG.Akui.Prakiraan.Cuacanya.Masih.Kurang.Akurat.

Vijay, Athira K., and M. Mathurakani. 2014. "Image Denoising Using Dual Tree Complex Wavelet Transform." *IJRET: International Journal of Research in Engineering and Technology* 3 (March): 60–64.

Winkler, Joab, Neil Lawrence, and Mahesan Niranjan. 2005. *Deterministic and Statistical Methods in Machine Learning: First International Workshop, Sheffield, UK, September 7-10, 2004. Revised Lectures*. Vol. 3635. Lecture Notes in Artificial Intelligence. Springer-Verlag Berlin Heidelberg.

Wu, Jiansheng, Jin Long, and Mingzhe Liu. 2015. "Evolving RBF Neural Networks for Rainfall Prediction Using Hybrid Particle Swarm Optimization and Genetic Algorithm." *Neurocomputing* 148: 136–42.

Yoshida, Hirotaka, Kenichi Kawata, Yoshikazu Fukuyama, Shinichi Takayama, and Yosuke Nakanishi. 2000. "A Particle Swarm Optimization for Reactive

Power and Voltage Control Considering Voltage Security Assessment." *Power Systems, IEEE Transactions on* 15 (4): 1232–39.

Yu, Shiwei, Ke Wang, and Yi-Ming Wei. 2015. "A Hybrid Self-Adaptive Particle Swarm Optimization–Genetic Algorithm–Radial Basis Function Model for Annual Electricity Demand Prediction." *Energy Conversion and Management* 91: 176–85.

Yu, Shiwei, Yi-Ming Wei, and Ke Wang. 2012. "A PSO–GA Optimal Model to Estimate Primary Energy Demand of China." *Energy Policy* 42: 329–40.

Yu, Wei, Baizhan Li, Hongyuan Jia, Ming Zhang, and Di Wang. 2015. "Application of Multi-Objective Genetic Algorithm to Optimize Energy Efficiency and Thermal Comfort in Building Design." *Energy and Buildings* 88: 135–43.

## LAMPIRAN A DATA HUJAN DASARIAN TAHUN 2009-2014 STASIUN KLIMATOLOGI KARANGPLOSO MALANG

BADAN METEOROLOGI KLIMATOLOGI DAN GEOFISIKA
STASIUN KLIMATOLOGI KARANGPLOSO
JL. ZENTANA 33 KARANGPLOSO MALANG, Telp. 461595
Telp : (0341) 464827, 461595 ; Fax : (0341) 464827 ;
Email : zentana33@yahoo.com , Website : karangploso.jatim.bmkg.go.id

### DATA HUJAN DASARIAN TAHUN 2009 - 2014
### STASIUN KLIMATOLOGI KARANGPLOSO MALANG

| Tahun | Jan | | | Feb | | | Mar | | | Apr | | | May | | | Jun | | | Jul | | | Aug | | | Sep | | | Oct | | | Nov | | | Dec | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| 2009 | 89 | 58 | 112 | 120 | 36 | 279 | 52 | 11 | 19 | 44 | 16 | 7 | 26 | 43 | 32 | 3 | 67 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 1 | 23 | 11 | 0 | 121 | 79 | 67 | 6 | 151 |
| 2010 | 123 | 59 | 165 | 73 | 110 | 36 | 222 | 54 | 76 | 217 | 152 | 157 | 138 | 48 | 156 | 12 | 15 | 3 | 43 | 35 | 16 | 7 | 38 | 89 | 11 | 126 | 51 | 67 | 30 | 45 | 295 | 16 | 155 | 190 | 50 | 21 |
| 2011 | 35 | 20 | 84 | 99 | 23 | 60 | 59 | 58 | 224 | 54 | 104 | 2 | 157 | 74 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 26 | 37 | 151 | 52 | 73 | 77 | 56 | 136 |
| 2012 | 145 | 80 | 62 | 191 | 152 | 79 | 111 | 71 | 28 | 51 | 9 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 49 | 52 | 7 | 10 | 102 | 38 | 217 | 109 | 156 |
| 2013 | 102 | 90 | 174 | 42 | 112 | 60 | 67 | 141 | 79 | 114 | 104 | 1 | 29 | 28 | 64 | 138 | 44 | 2 | 57 | 38 | 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 89 | 32 | 75 | 63 | 212 | 152 | 57 |
| 2014 | 89 | 55 | 161 | 38 | 121 | 35 | 82 | 95 | 6 | 112 | 38 | 143 | 25 | 11 | 4 | 4 | 5 | 36 | 3 | 4 | 0 | 19 | 21 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 60 | 46 | 35 | 109 | 160 | 69 |

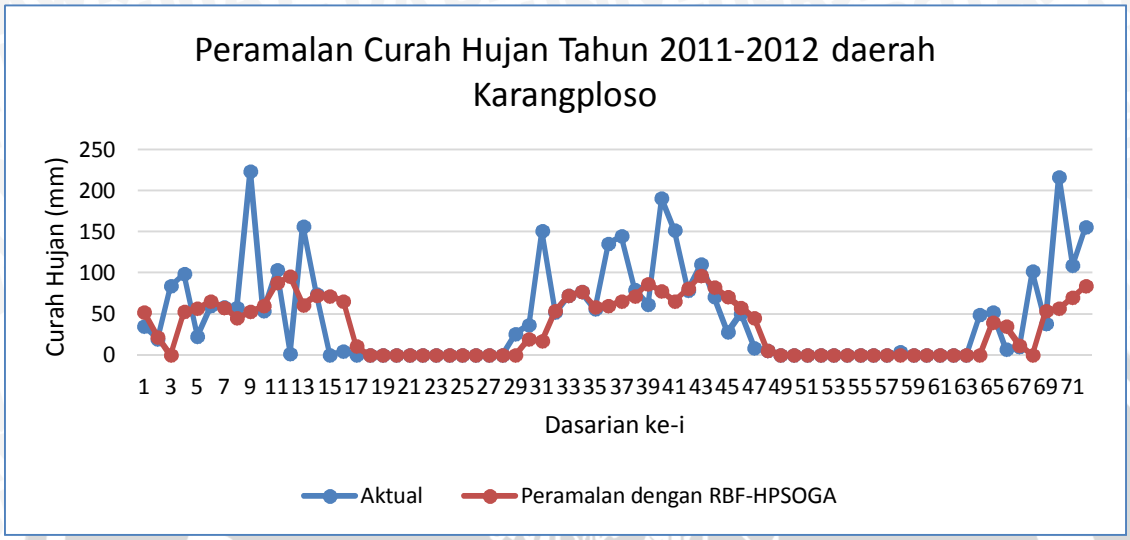
**Keterangan :**
0 = Tidak ada hujan .
TTU = Terjadi hujan tetapi tidak terukur

Malang,
Kepala Seksi Observasi dan Informasi
Stasiun Klimatologi Karangploso Malang

Aminudin Al Romi , SP
NIP. 19690604 199003 1 001

# LAMPIRAN B VISUALISASI HASIL PERAMALAN CURAH HUJAN PENGUJIAN JUMLAH DATA LATIH OPTIMAL

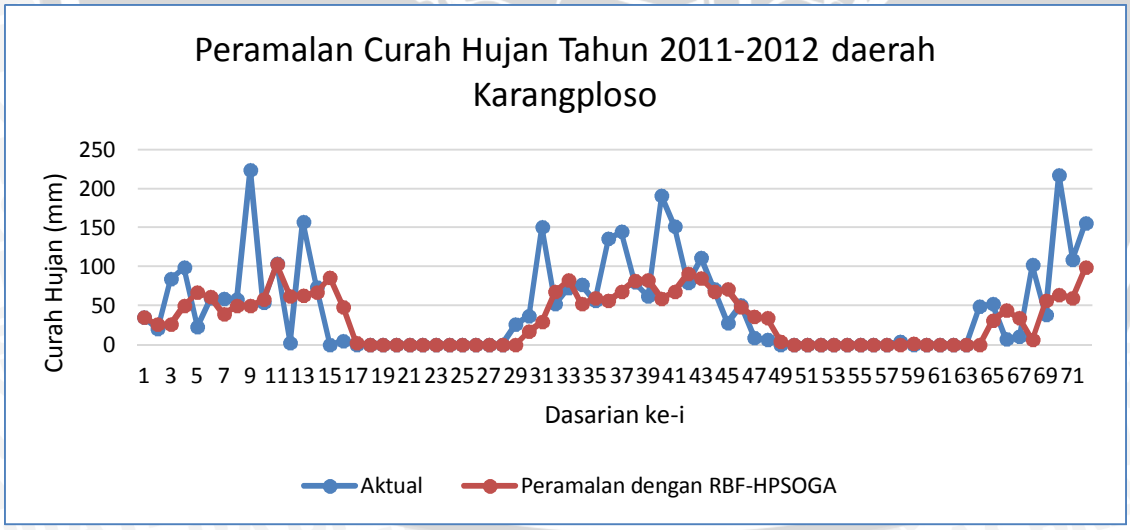
1. Percobaan ke-1




Nilai evaluasi peramalan:

- MAE : 30.8888888888889
- RMSE : 42.0211586915185
- CC : 0.243426582051645
- MAPE : 31.4505858508891

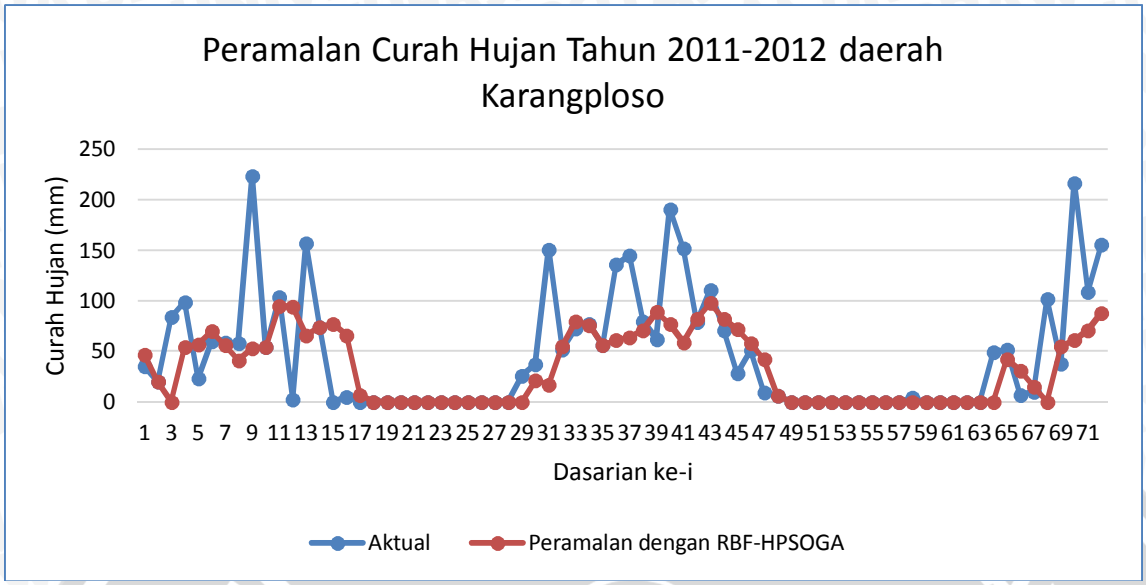
2. Percobaan ke-2




Nilai evaluasi peramalan:

- MAE : 31.8888888888889
- RMSE : 47.7644684304604
- CC : -0.0808772509789947
- MAPE : 29.2462669207438

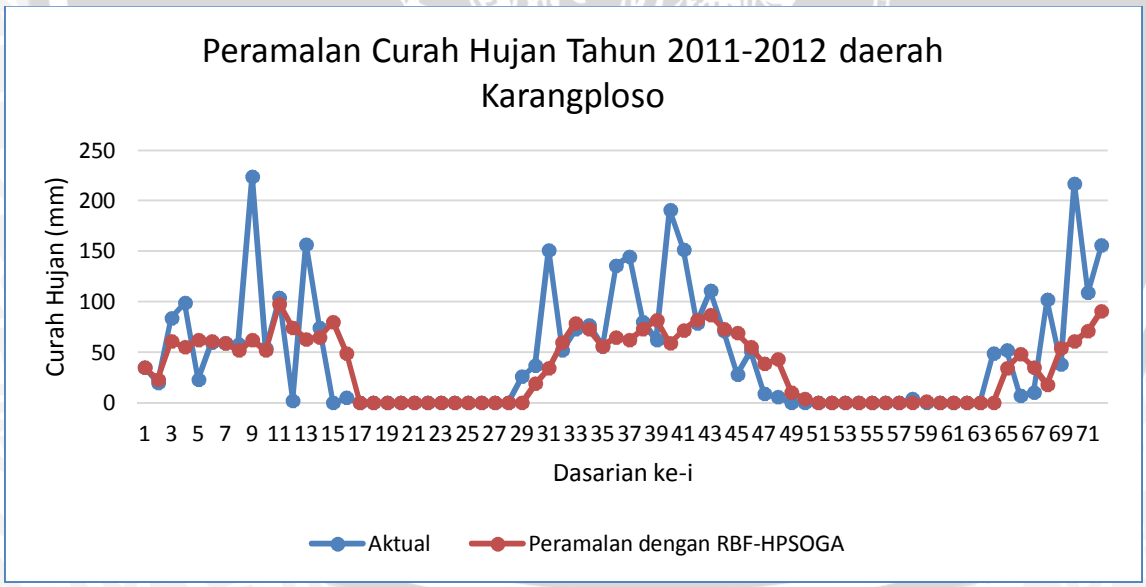
3. Percobaan ke-3




Nilai evaluasi peramalan:

- MAE : 29.7777777777778
- RMSE : 41.6119640915393
- CC : 0.29336540624285
- MAPE : 29.6307474791799

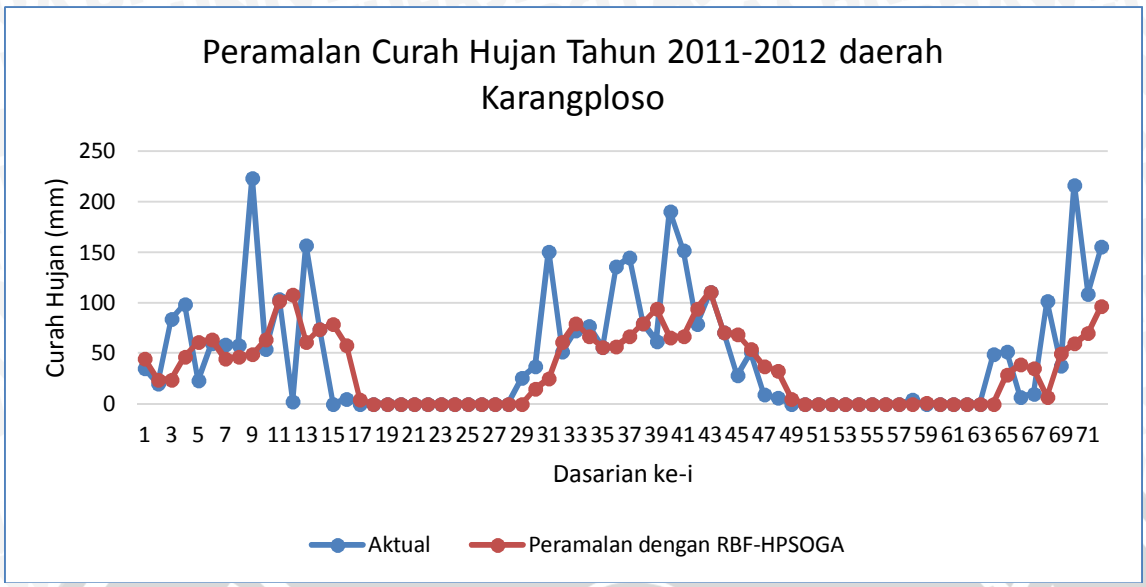
4. Percobaan ke-4




Nilai evaluasi peramalan:

- MAE : 29.8888888888889
- RMSE : 44.3157860010278
- CC : 0.0886888530165135
- MAPE : 29.053168519752

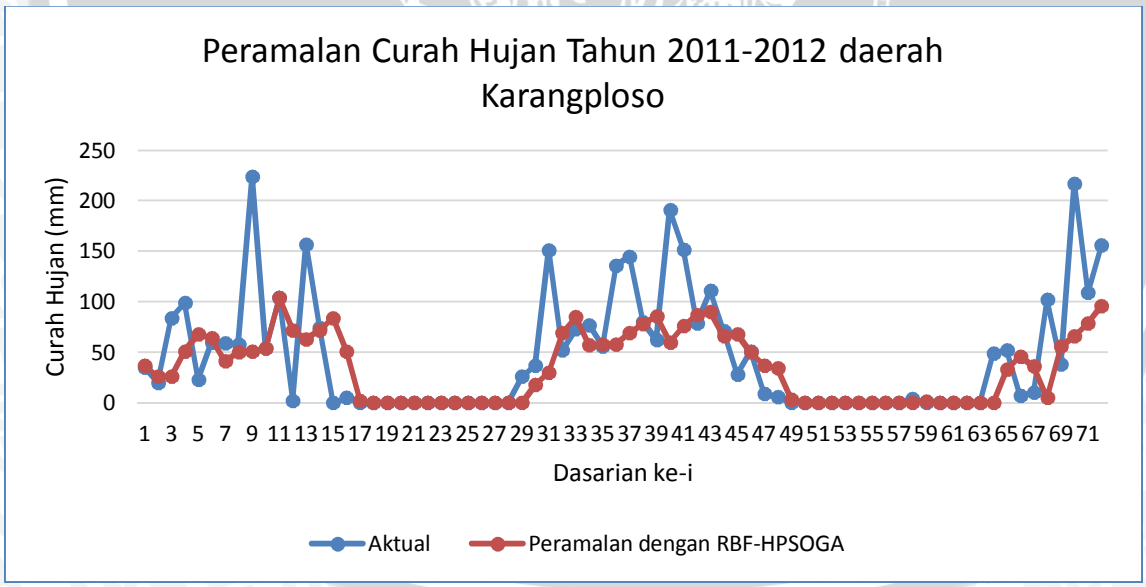
5. Percobaan ke-5




Nilai evaluasi peramalan:

- MAE : 32.8888888888889
- RMSE : 44.226688774992
- CC : 0.113009791958865
- MAPE : 32.8340836057622

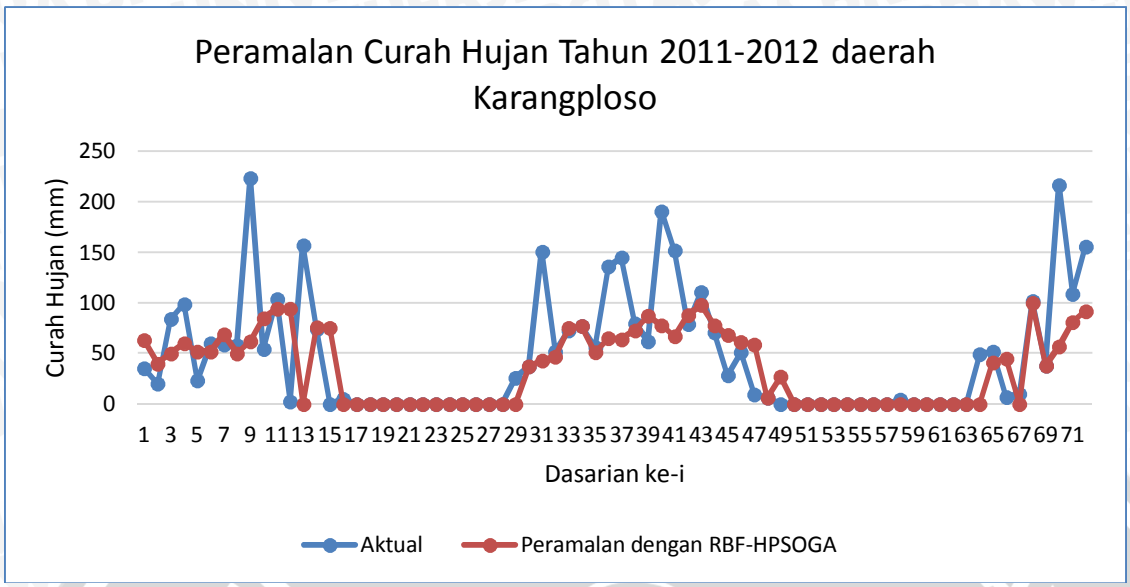
6. Percobaan ke-6




Nilai evaluasi peramalan:

- MAE : 32.8888888888889
- RMSE : 47.029541543351
- CC : -0.00062635571675908
- MAPE : 31.8847257256999

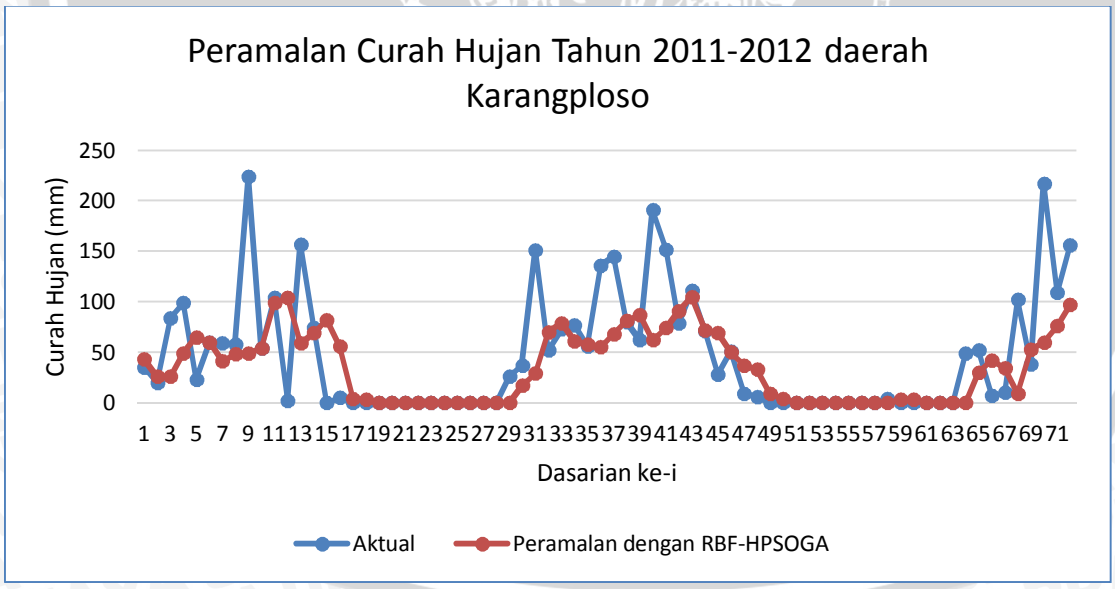
7. Percobaan ke-7




Nilai evaluasi peramalan:

- MAE : 31
- RMSE : 42.2965719651132
- CC : 0.290318983985066
- MAPE : 31.1829803743593

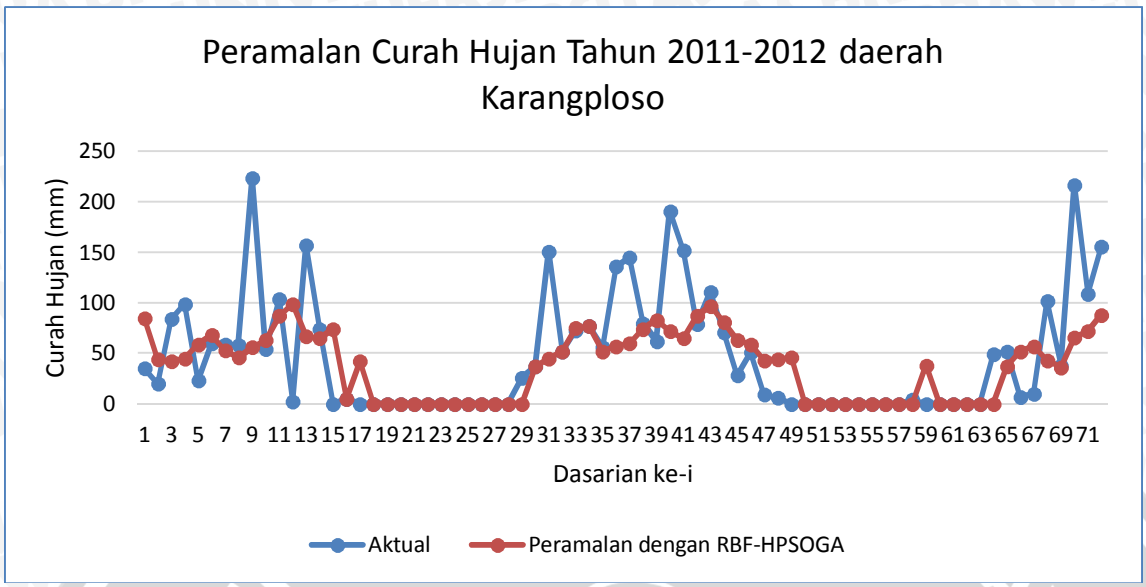
8. Percobaan ke-8




Nilai evaluasi peramalan:

- MAE : 31.7777777777778
- RMSE : 46.2048578494609
- CC : -0.0451834332935068
- MAPE : 31.2881117708931

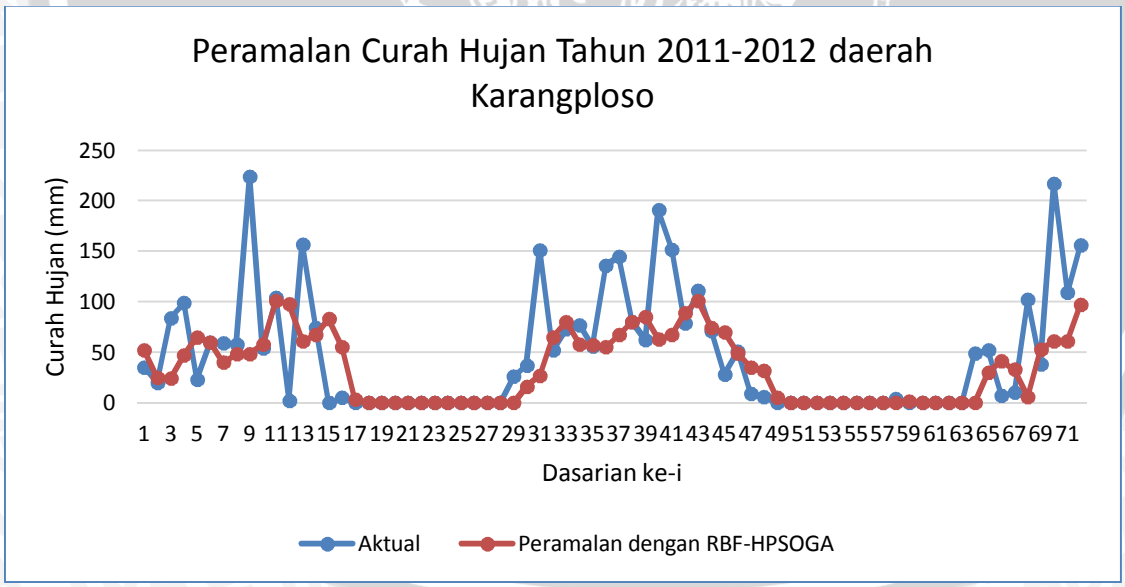
9. Percobaan ke-9




Nilai evaluasi peramalan:

- MAE : 32.7777777777778
- RMSE : 44.9060748179526
- CC : 0.113469638636007
- MAPE : 32.7438551939513

10. Percobaan ke-10

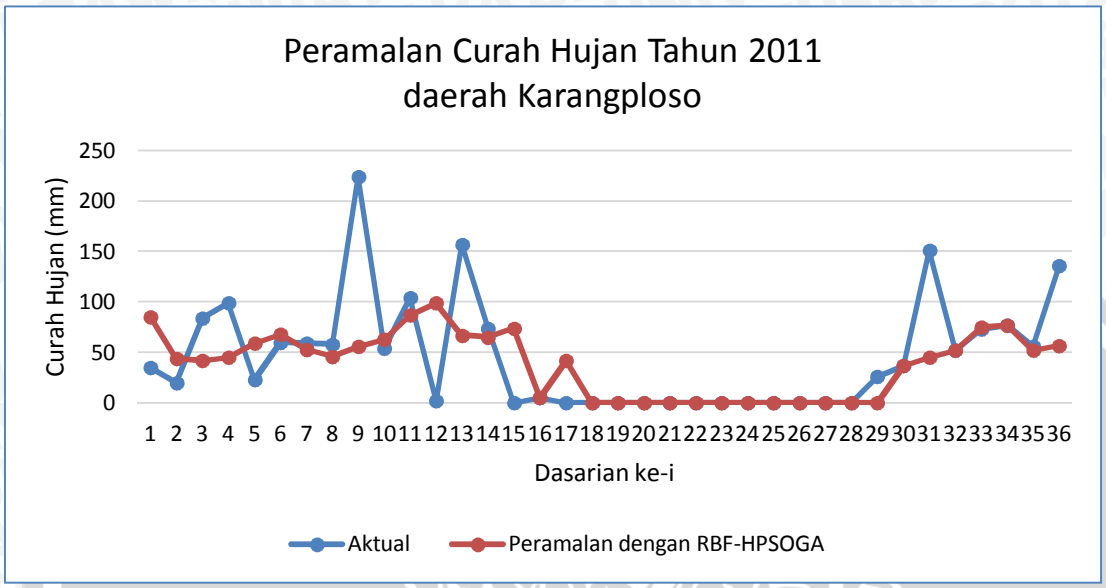



Nilai evaluasi peramalan:

- MAE : 30.2222222222222
- RMSE : 45.8911756223351
- CC : 0.00905423610078206
- MAPE : 28.570628137746

# LAMPIRAN C VISUALISASI HASIL PERAMALAN CURAH HUJAN PADA TAHUN LAIN DENGAN SOLUSI TERBAIK

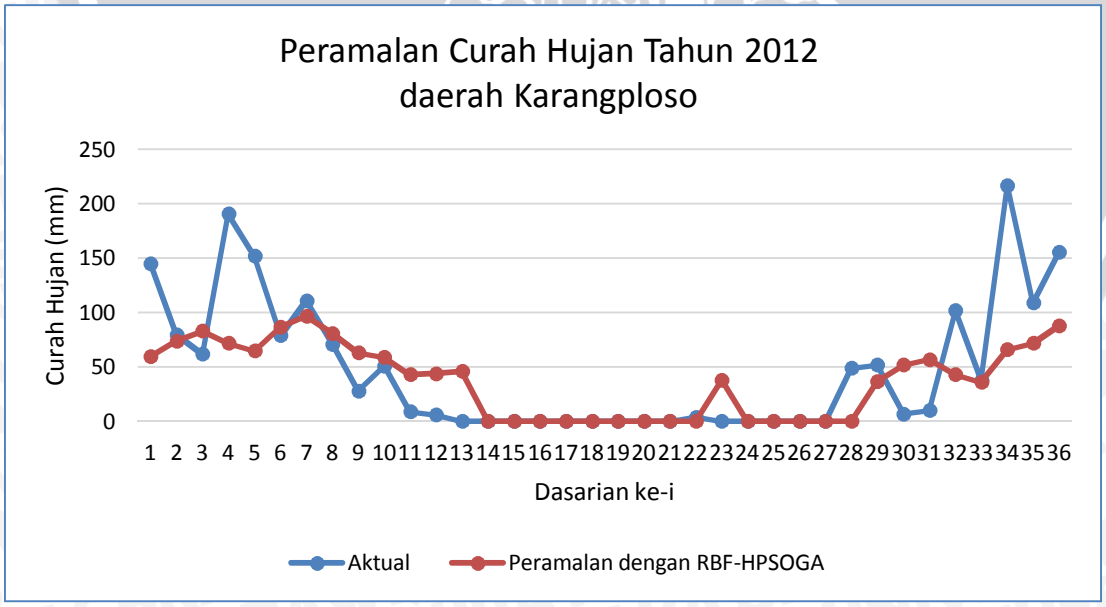
1. Tahun 2011




Nilai evaluasi peramalan:

- MAE : 26.5277777777778
- RMSE : 47.4028245759446
- CC : 0.515967262921901
- MAPE : 32.3446233025503

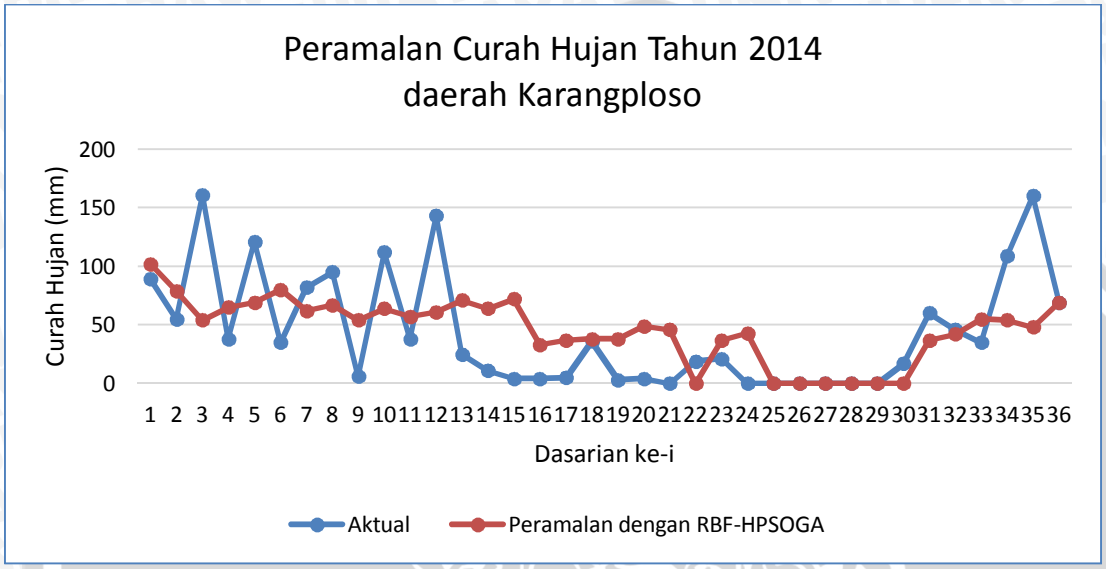
2. Tahun 2012



Nilai evaluasi peramalan:

- MAE : 28.5
- RMSE : 46.0663772301568
- CC : 0.697963598303052
- MAPE : 39.197538065275

3. Tahun 2014




Nilai evaluasi peramalan:

- MAE : 32.7222222222222
- RMSE : 43.0058135604944
- CC : 0.475505392333049
- MAPE : 56.9213452752337