BAB IV
HASIL DAN PEMBAHASAN

4.1 Data Penelitian

Data yang diperlukan dalam penelitian ini adalah data diri penderita gagal ginjal dari salah satu rumah sakit di Kota Bandung yang terdiri dari data jenis kelamin, usia, tinggi badan, berat badan dan pekerjaan. Data lengkap penderita gagal ginjal dapat dilihat pada Lampiran 4. Data lain yang dibutuhkan adalah data kandungan gizi dalam menu makanan yang didapatkan dari Tabel Komposisi Pangan Ide 2017 dan juga data dari instalasi gizi salah satu rumah sakit di Kota Bandung.

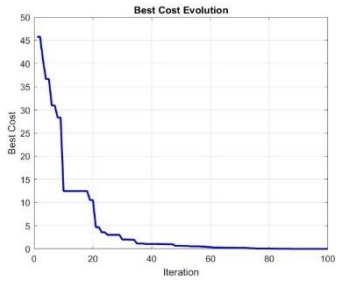
Terdapat 96 data menu makanan yang terdiri dari 4 kategori dengan masing-masing pembagian yaitu 6 menu makanan pokok sebagai sumber karbohidrat, 37 menu lauk pauk sebagai sumber protein, 26 menu sayur dan 27 jenis buah. Dalam tabel tersebut memuat nama menu, biaya, kandungan gizi yang terdiri dari energi, protein, lemak, dan karbohidrat. Untuk data lengkap mengenai kandungan gizi dan biaya dapat dilihat di Lampiran 5 hingga 8. Pada masalah optimasi menu makanan bagi penderita gagal ginjal ini akan diselesaikan menggunakan bantuan *software* MATLAB R2024b.

4.2 Validasi

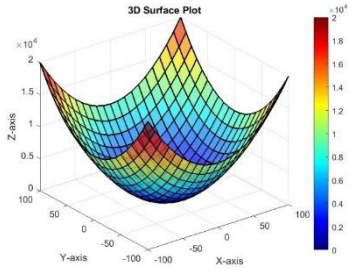
Untuk menguji bahwa model optimasi dan algoritma dapat digunakan untuk kasus penyusunan menu makan bagi penderita gagal ginjal, maka perlu dilakukan validasi terlebih dahulu. Validasi dilakukan untuk memverifikasi model optimasi, algoritma, dan program yang digunakan adalah benar atau tidak. Pada penelitian ini, validasi dilakukan sebanyak 2 kali. Validasi pertama adalah validasi algoritma *PSO* dilakukan dengan menggunakan beberapa fungsi *benchmark*. Fungsi *benchmark* yang dipilih digunakan untuk membandingkan hasil yang diperoleh dengan solusi optimal yang diketahui dan untuk memastikan keakuratan dan konsistensi algoritma yang digunakan. Dengan demikian, fungsi *benchmark* tidak hanya berfungsi sebagai alat ukur, namun juga sebagai standar evaluasi untuk menilai sejauh mana algoritma yang diuji dapat mencapai solusi yang mendekati

nilai optimal dalam ruang pencarian yang kompleks. Fungsi *benchmark* yang digunakan yaitu Fungsi *Sphere (Unimodal)*, Fungsi *Schwefel (Unimodal),* dan Fungsi *Rastrigin (Multimodal).*

Berdasarkan hasil pengujian, algoritma (*PSO*) mampu menyelesaikan fungsi *Sphere*, *Schwefel*, dan *Rastrigin* hingga mencapai nilai minimum yang diharapkan. Kemampuan algoritma *PSO* untuk menemukan solusi optimal pada ketiga fungsi uji tersebut menunjukkan bahwa algoritma ini berjalan dengan baik dan efektif dalam mengoptimalkan masalah optimasi, sehingga dapat dianggap andal untuk diterapkan pada berbagai kasus. Berikut adalah hasil validasi fungsi *Sphere*, *Schwefel*, dan *Rastrigin* menggunakan *PSO*.



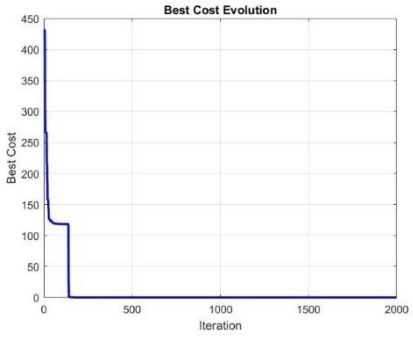
Gambar 4. 1 Subplot Fungsi *Sphere* dengan Algoritma *PSO*



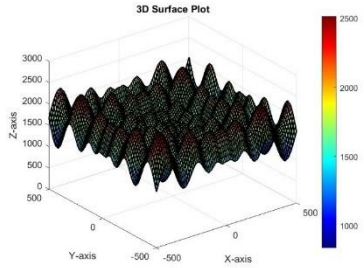
Gambar 4. 2 3D Plot Fungsi *Sphere* dengan Algoritma *PSO*

Validasi menggunakan fungsi *Sphere*, ditentukan batas $-10 \leq x_i \leq 10$, juga ditentukan beberapa parameter seperti ω= 0,6; $c_1 = c_2 = 2$; iterasi maksimal = 150; dimensi/$nVar$ = 10 dan $nPop$ = 50. Berdasarkan hasil komputasi, didapatkan *gbest* pada iterasi ke 150 dengan nilai *Global Best Cost* = 0,0064093 dengan *Global Best Position* : ($x_1 = 0{,}026803$, $x_2 = 0{,}011416$, $x_3 = 0{,}0091308$, $x_4 = -0{,}015401$, $x_5 = 0{,}0022051$, $x_6 = 0{,}030836$, $x_7 = 0{,}0094888$, $x_8 = 0{,}031177$, $x_9 = -0{,}039234, x_{10} = -0{,}041024$). Untuk

tabel iterasi penyelesaian fungsi dapat dilihat di Lampiran 9. Hasil ini menunjukkan bahwa algoritma *PSO* dapat mencari nilai minimum dari fungsi *Sphere*, yang hasilnya divalidasi oleh penelitian (Jamil & Yang, 2013).

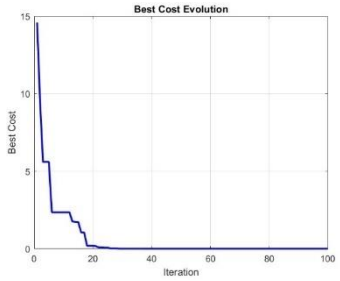


Gambar 4. 3 Subplot Fungsi *Schwefel* dengan Algoritma *PSO*

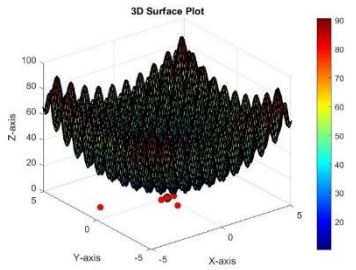


Gambar 4. 4 3D Plot Fungsi *Schwefel* dengan Algoritma *PSO*

Validasi menggunakan fungsi *Schwefel*, ditentukan batas $-500 \leq x_i \leq 500,$ juga ditentukan beberapa parameter seperti ω= 0,7; $c_1 = c_2 = 2$; iterasi maksimal = 2000; dimensi/$nVar$ = 4 dan $nPop$ = 60. Berdasarkan hasil komputasi, didapatkan nilai *gbest* pada iterasi ke 522 dengan nilai *Global Best Cost* = 0,00005091 dengan *Global Best Position* : ($x_1 = 420{,}9687,\ x_2 = 420{,}9687,\ x_3 = 420{,}9687,\ x_4 = 420{,}9687$). Untuk tabel iterasi penyelesaian fungsi dapat dilihat di Lampiran 10. Hasil ini menunjukkan bahwa algoritma *PSO* dapat mencari nilai minimum dari fungsi *Schwefel*, yang hasilnya divalidasi oleh penelitian (Dieterich & Hartke, 2012).



Gambar 4. 5 Subplot Fungsi *Rastrigin* dengan Algoritma *PSO*



Gambar 4. 6 3D Plot Fungsi *Rastrigin* dengan Algoritma *PSO*

Validasi menggunakan fungsi *Rastrigin*, ditentukan batas $-5.12 \leq x_i \leq 5.12$, juga ditentukan beberapa parameter seperti ω= 0,5; $c_1 = c_2 = 2$; Iterasi Maksimal = 100; Dimensi/n = 3 dan $nPop$ = 50. Berdasarkan hasil komputasi, didapatkan nilai *gbest* pada iterasi ke 99 dengan nilai *Global Best Cost* = 0,00000000074687 dengan *Global Best Position* : ($x_1 = 0.00000025899$, $x_2 = 0.0000015853$, $x_3 = -0.0000010882$). Untuk tabel iterasi penyelesaian fungsi dapat dilihat di Lampiran 11. Hasil ini menunjukkan bahwa algoritma *PSO* dapat mencari nilai minimum dari fungsi *Rastrigin*, yang hasilnya divalidasi oleh penelitian (Dieterich & Hartke, 2012).

Selanjutnya untuk program validasi dikembangkan dan disesuaikan dengan studi kasus penelitian ini mengenai optimasi menu makanan. Untuk validasi ke 2, dilakukan dengan membandingkan hasil dari program optimasi *PSO* dengan perhitungan manual yang telah dilakukan di bab sebelumnya. Berdasarkan hasil validasi, perhitungan program dan perhitungan manual memiliki solusi yang sama. Sehingga dapat disimpulkan bahwa program ini berjalan dengan baik untuk menyelesaikan permasalahan optimasi menu makanan. Hasil validasi perhitungan

secara manual untuk kasus pada bab 3 menggunakan program *PSO* dapat dilihat di Lampiran 12.

4.3 Tahap Implementasi

Pada sub bab ini akan dibahas mengenai langkah-langkah penyelesaian masalah optimasi biaya dalam penyusunan menu makanan bagi penderita gagal ginjal dengan menggunakan algoritma *Particle Swarm Optimization.* Berikut langkah-langkah kerja algoritma:

1. Memasukan data penderita gagal ginjal seperti jenis kelamin, usia, tinggi badan, berat badan dan pekerjaan. Kemudian hitung *TEE* atau *Total energy Expenditure* (Kebutuhan Energi Sehari).
2. Inisialisasi parameter *PSO*
    - ω = 0,9
    - c_1 = 2
    - c_2 = 2
    - R_1 = 0,2
    - R_2 = 0,4
    - Iterasi Maksimal = 100
3. Proses inisialisasi populasi partikel awal.

    Untuk pembangkitan populasi awal, akan ditentukan banyaknya populasi/partikel sebanyak $nPop = 10$. Kemudian akan ditentukan jumlah dimensi/nVar pada setiap partikel sebanyak 12 dimensi. Dimensi ini ditentukan berdasarkan banyaknya menu yang disajikan dalam 1 hari untuk makan pagi, siang dan malam yaitu sebanyak 12 menu. Terdiri dari 4 kategori yaitu 3 menu makanan pokok, 3 menu lauk-pauk, 3 menu sayur, dan 3 menu buah-buahan. Sehingga dalam 1 partikel mewakili menu yang akan dikonsumsi selama 1 hari.

    Berdasarkan data, akan ditentukan masing-masing rentang dalam pengacakan angka bulat untuk setiap kategori menu.

    - Jumlah Variasi Kategori Makanan Pokok = 6, maka rentangnya adalah 1-6
    - Jumlah Variasi Kategori Lauk-Pauk = 37, maka rentangnya adalah 1-37

- Jumlah Variasi Kategori Sayur = 26, maka rentangnya adalah 1-26
- Jumlah Variasi Kategori Buah = 26, maka rentangnya adalah 1-27

Angka yang didapatkan dalam proses pengacakan angka bulat sesuai rentang ini akan mewakili indeks menu.

4. Proses menghitung *fitness* partikel.

   Nilai *fitness* dapat diperoleh dengan terlebih dulu melakukan perhitungan:

   a. Berat masing-masing menu yang didapatkan dari partikel ke-n.
   b. Kandungan gizi berupa energi, protein, lemak dan karbohidrat per menu partikel ke-n.
   c. Total biaya yang harus dikeluarkan partikel ke-n.
   d. Variasi menu makanan partikel ke-n.
   e. Penalti gizi berupa selisih energi dan gizi yang harus dipenuhi dengan energi dan gizi yang didapatkan dari partikel ke-n.

   Nilai *fitness* dapat dihitung menggunakan Persamaan 2.4.

5. Proses menghitung nilai *pBest* dan *gBest*.

   Nilai *pBest* akan dipilih berdasarkan nilai *fitness* terbaik dari partikel ke-n. Sedangkan nilai *gBest* akan dipilih berdasarkan nilai *fitness* terbaik dari seluruh populasi.

6. Proses menghitung kecepatan partikel untuk iterasi selanjutnya.

   Kecepatan awal akan ditentukan dengan nilai 0. Dengan menggunakan persamaan 3.1 akan dihitung kecepatan baru pada setiap partikel.

7. Proses menghitung posisi partikel untuk iterasi selanjutnya.

   Posisi partikel baru diperoleh dengan menggunakan persamaan 3.2. Perhitungan ini dilakukan untuk setiap partikel, sehingga membentuk posisi partikel baru yang dapat digunakan untuk iterasi selanjutnya. Namun, perlu diperhatikan nilai untuk setiap posisi partikel baru. Agar memudahkan komputasi dan perhitungan, posisi baru akan mengalami pembulatan dan penyesuaian sesuai dengan rentang yang telah ditentukan. Proses ini akan berulang hingga mencapai jumlah iterasi maksimal.

4.4 Hasil Implementasi

Setelah seluruh tahapan dalam algoritma *PSO* dengan iterasi yang ditentukan selesai, maka akan diperoleh partikel terbaik. Partikel ini memiliki nilai *fitness* tertinggi sehingga menu yang dihasilkan akan memenuhi kebutuhan gizi penderita gagal ginjal dengan biaya yang murah dan menu yang bervariasi dalam sehari. Partikel tersebut adalah solusi optimal untuk masalah optimasi menu pada penderita gagal ginjal.

Berikut adalah hasil implementasi pada penderita gagal ginjal ID 001:

===== Informasi Pasien =====

- ID Pasien: 001
- Usia: 61 tahun
- Jenis Kelamin: L
- Berat Badan: 85 kg
- Tinggi Badan: 165 cm
- Kebutuhan Energi Sehari/TEE: 2132.91 kkal
- Protein (12.5% dari TEE): 66.6534 gram
- Lemak (22.5% dari TEE): 53.3228 gram
- Karbohidrat (65% dari TEE): 346.5979 gram

========== OUPUT ==========

- ====== Partikel Terbaik =====
- Global Best Particle: 9
- Global Best Position: 1 12 10 12 3 22 18 27 4 19 5 11
- Global Best Cost: 1.0829
- Total Harga: 54113.6191
- Total Energi: 1975.8333 kkal
- Total Protein: 75.2459 gram
- Total Lemak: 30.0644 gram
- Total Karbohidrat: 267.0972 gram
- Variasi Makanan: 12

Berdasarkan data hasil implementasi penyelesaian masalah optimasi menu makanan penderita gagal ginjal menggunakan *PSO*, diperoleh hasil sebagai berikut:

a. Rincian susunan menu yang disarankan untuk pasien ID 001 dapat dilihat di Lampiran 13.
b. Selisih kebutuhan gizi dan gizi yang didapatkan adalah:
    - Selisih energi = 157,0767 kkal, artinya kebutuhan energi terpenuhi sebanyak 92%.
    - Selisih protein = 8,5925 gram, artinya kebutuhan protein terpenuhi sebanyak 112%.
    - Selisih lemak = 23,2584 gram, artinya kebutuhan lemak terpenuhi sebanyak 56,3%
    - Selisih karbohidrat = 79,5007 gram, artinya kebutuhan protein terpenuhi sebanyak 77%
c. Variasi makanan bernilai 12, artinya tidak terdapat duplikat menu.
d. Total biaya yang dibutuhkan sebesar Rp54.113.6191, sehingga biaya untuk 1 kali makan adalah Rp18.037,873.

Berdasarkan hasil komputasi, menu yang tersusun memiliki selisih yang rendah dengan kebutuhan sebenarnya. Meskipun begitu selain memperhatikan gizi yang didapat, perlu diperhatikan pula biaya yang dibutuhkan dan variasi menu makanan. Sehingga dapat disimpulkan bahwa algoritma *PSO* berhasil menyelesaikan masalah optimasi menu makanan. Untuk implementasi pada pasien yang lain, hasil komputasi dapat dilihat secara rinci di Lampiran 14 hingga 25.

4.5 Analisis Parameter *PSO*

Pada bagian ini akan dibahas pengaruh perubahan nilai parameter *PSO* terhadap nilai *fitness* atau solusi optimal yang didapatkan. Komputasi akan dilakukan dengan menggunakan data dari pasien ID 001.

1. Pengaruh iterasi maksimal terhadap nilai *fitness*.

    Variasi iterasi maksimal yang digunakan adalah 50, 100, dan 150 iterasi, dengan masing-masing dilakukan 10 kali percobaan untuk

mendapatkan rata-rata nilai *fitness*. Berdasarkan hasil analisis pada Tabel 4.1, rata-rata nilai *fitness* untuk 50 iterasi adalah 154,5, meningkat menjadi 195,1 pada 100 iterasi. Namun, ketika iterasi maksimal dinaikkan menjadi 150, rata-rata nilai *fitness* justru menurun menjadi 180,2. Hal ini menunjukkan bahwa peningkatan iterasi maksimal hingga titik tertentu dapat meningkatkan performa algoritma dalam mencapai solusi yang lebih optimal. Namun, setelah melewati jumlah iterasi tertentu, performa dapat menurun, kemungkinan disebabkan oleh kecenderungan algoritma untuk melakukan eksplorasi yang berlebihan sehingga mengurangi efisiensi eksploitasi solusi optimal.

2. Pengaruh beban inersia terhadap nilai *Fitness*.

   Variasi beban inersia yang digunakan adalah 0,4, 0,6, dan 0,8, dengan masing-masing dilakukan 10 kali percobaan untuk mendapatkan rata-rata nilai *fitness*. Berdasarkan hasil analisis pada Tabel 4.2, rata-rata nilai *fitness* yang diperoleh untuk beban inersia 0,4 adalah 115,9, menurun menjadi 111,7 pada beban inersia 0,6, dan meningkat signifikan menjadi 188,4 ketika beban inersia ditingkatkan menjadi 0,8. Hasil ini menunjukkan bahwa nilai beban inersia memiliki pengaruh yang signifikan terhadap kinerja algoritma dalam mencapai nilai *fitness* optimal. Dengan meningkatnya nilai beban inersia algoritma dapat menjelajahi ruang pencarian dengan lebih efisien. Namun, perlu dilakukan analisis lebih lanjut untuk memastikan konsistensi hasil.

3. Pengaruh jumlah populasi terhadap nilai *fitness*.

   Variasi jumlah partikel yang digunakan adalah 10, 15, dan 20, dengan masing-masing dilakukan 10 kali percobaan untuk mendapatkan rata-rata nilai *fitness*. Hasil analisis pada Tabel 4.3 menunjukkan bahwa rata-rata nilai *fitness* yang diperoleh untuk jumlah partikel 10 adalah 217,8, menurun menjadi 198,7 saat jumlah partikel ditingkatkan menjadi 15, dan kembali meningkat menjadi 213,8 ketika jumlah partikel dinaikkan menjadi 20. Hasil ini mengindikasikan bahwa pemilihan jumlah partikel berpengaruh terhadap kinerja algoritma *PSO* dalam mencapai nilai *fitness* optimal. Pada jumlah partikel 10, algoritma cenderung lebih fokus dan cepat

dalam menemukan solusi yang baik. Namun, peningkatan jumlah partikel hingga 15 kemungkinan menyebabkan eksplorasi ruang pencarian menjadi kurang efisien, sehingga nilai *fitness* menurun. Sementara itu, pada jumlah partikel 20, algoritma menunjukkan perbaikan performa, meskipun belum melampaui nilai *fitness* pada jumlah partikel 10. Penelitian lebih lanjut dapat dilakukan untuk memahami pola ini dengan lebih mendalam.

Tabel 4. 1 Hasil Percobaan Pengaruh Iterasi Maksimal Terhadap *Fitness*

Iterasi Maksimal	Percobaan	*Fitness*
50	1	230,1
	2	93,0
	3	426,2
	4	114,2
	5	131,2
	6	77,1
	7	92,3
	8	97,1
	9	197,9
	10	85,9
Rata-rata		**154,5**
100	1	102,8
	2	115,9
	3	131,3
	4	361,0
	5	68,3
	6	114,4
	7	188,5
	8	200,0
	9	478,8
	10	189,8
Rata-rata		**195,1**
200	1	224,5
	2	200,4
	3	148,4
	4	163,5
	5	371,0
	6	153,5
	7	107,5
	8	186,3
	9	134,1
	10	113,3
Rata-rata		**180,2**

Tabel 4. 2 Hasil Percobaan Pengaruh Beban Inersia Terhadap *Fitness*

Beban Inersia	Percobaan	*Fitness*
0,4	1	72,3
	2	163,8
	3	144,4
	4	111,8
	5	96,6
	6	90,3
	7	115,4
	8	156,3
	9	127,8
	10	80,5
Rata-rata		**115,9**
0,6	1	90,1
	2	71,2
	3	147,2
	4	294,8
	5	93,7
	6	94,1
	7	86,2
	8	89,3
	9	49,0
	10	100,9
Rata-rata		**111,7**
0,8	1	505,2
	2	130,2
	3	113,7
	4	98,7
	5	148,0
	6	223,8
	7	138,0
	8	120,0
	9	152,2
	10	253,7
Rata-rata		**188,4**

Tabel 4. 3 Hasil Percobaan Jumlah Partikel Iterasi Terhadap *Fitness*

Jumlah Partikel	Percobaan	*Fitness*
10	1	74,8
	2	125,6
	3	197,9
	4	223,5
	5	84,6
	6	519,1
	7	158,8
	8	249,8
	9	412,6
	10	131,4
Rata-rata		**217,8**
15	1	162,9
	2	288,4
	3	176,1
	4	130,3
	5	118,0
	6	134,2
	7	246,7
	8	277,6
	9	158,7
	10	294,5
Rata-rata		**198,7**
20	1	182,6
	2	220,8
	3	177,8
	4	75,9
	5	162,0
	6	177,8
	7	362,3
	8	239,4
	9	313,5
	10	226,4
Rata-rata		**213,8**

4.6 Analisis Perbandingan dengan Menu yang Sudah Ada

Dalam penelitian ini dilakukan perbandingan nilai *fitness* menu makanan yang sudah ada dengan hasil yang diperoleh menggunakan metode komputasi berbasis algoritma *Particle Swarm Optimization* (*PSO*). Perbandingan ini bertujuan untuk melihat keunggulan dari algoritma *PSO* dalam hal efisiensi waktu, kualitas hasil, dan variasi menu yang dihasilkan. Tabel 4.5 menyajikan hasil perbandingan berdasarkan penalti gizi, biaya menu, jumlah variasi, serta nilai *fitness*. Berikut adalah hasil perbandingan metode manual dan komputasi *PSO*:

Tabel 4. 4 Rincian Gizi dan Biaya Menu yang Sudah Ada

Menu	Biaya	Energi	Protein	Lemak	Karbo
Nasi	1604,2	284,4	4,7	0,5	62,9
Bistik Ayam	1256,0	37,5	4,7	2,0	0,0
Sayur Caisin	1200,0	25,0	2,1	0,5	4,3
Nasi	1604,2	455,0	7,6	0,8	100,6
Ayam goreng	752,0	60,0	9,2	2,2	0,2
Sayur asem waluh	8375,0	266,0	9,1	4,6	51,4
Nasi	1604,2	398,1	6,6	0,7	88,0
Ikan Saus Tiram	1780,3	52,5	6,8	2,6	2,6
Sup Wortel Oyong	5565,0	239,8	6,2	0,6	53,6
Total	**23740,8**	**1818,3**	**57,1**	**14,4**	**363,6**

Tabel 4. 5 Rincian Gizi dan Biaya Menu Hasil Komputasi *PSO*

Menu	Biaya	Energi	Protein	Lemak	Karbo
Mie Basah	7755,68	284,38	1,94	10,66	45,24
Cumi Goreng	608,49	37,50	5,75	1,43	0,00
Sayur Asem Waluh	10468,75	166,25	5,69	2,84	32,13
Bihun	1961,21	455,00	6,15	0,13	107,34
Ayam Goreng Kentukcy	656,80	60,00	6,25	3,66	0,07
Sup Wortel Oyong	11130,00	274,00	7,13	0,73	61,27
Bihun	1716,06	398,13	5,38	0,11	93,93
Abon Sapi	4212,21	78,75	8,79	8,79	0,00
Capcay	20063,75	45,50	3,00	0,55	9,39
Total	**58572,95**	**1799,50**	**50,07**	**28,91**	**349,37**

Tabel 4. 6 Hasil Perbandingan dengan Menu yang Sudah Ada

	Menu Lama	**Menu**
Penalti Gizi	815,2	827,3
Biaya	Rp23.740,82	Rp58.572,95
Variasi	7	9
Fitness	19,3	21,1

Dari hasil perbandingan tersebut, dapat disimpulkan bahwa menu yang dihasilkan dengan komputasi *PSO* memberikan nilai *fitness* yang lebih tinggi dibandingkan dengan menu yang sudah ada. Hasil ini menunjukkan bahwa menu yang dihasilkan dengan komputasi *PSO* lebih optimal dan sesuai kriteria. Selain itu, menu yang dihasilkan dengan komputasi *PSO* juga menghasilkan menu dengan variasi yang lebih banyak (9 jenis makanan dibandingkan 7 pada metode manual), sehingga lebih beragam. Namun, menu yang dihasilkan dengan komputasi *PSO* menghasilkan total biaya menu yang lebih tinggi dibandingkan dengan menu yang sudah ada. Keunggulan signifikan dari *PSO* terletak pada efisiensi waktu, di mana waktu perhitungan hanya $\pm$20 detik. Hal ini menunjukkan bahwa *PSO* lebih efisien untuk menyelesaikan masalah optimasi menu dengan jumlah data yang kompleks.

4.7 Analisis Pemenuhan Kebutuhan Gizi

Pada subab ini, akan dilakukan analisis mengenai efektivitas algoritma *Particle Swarm Optimization (PSO)* dalam menyusun menu makanan yang dapat memenuhi kebutuhan gizi yang ditargetkan. Dalam konteks optimasi menu makanan, pemenuhan kebutuhan gizi menjadi faktor utama yang harus dipertimbangkan, selain aspek harga dan variasi menu. Melalui penggunaan *PSO*, diharapkan dapat ditemukan solusi yang tidak hanya memenuhi batasan gizi, tetapi juga mempertimbangkan faktor biaya yang efisien serta variasi yang cukup untuk menjaga keberagaman dalam pola makan. Analisis ini akan mengkaji seberapa baik algoritma *PSO* mampu mencapai keseimbangan antara pencapaian gizi yang optimal dan pengendalian harga serta variasi dalam menu makanan yang dihasilkan.

Berikut adalah hasil perhitungan kebutuhan gizi untuk pasien ID 001 hingga ID 013.

Berdasarkan hasil perhitungan menggunakan algoritma *Particle Swarm Optimization (PSO),* rata-rata pemenuhan kebutuhan gizi mencapai 86,47%, yang menunjukkan bahwa algoritma mampu mendekati target kebutuhan gizi dengan cukup baik meskipun belum sepenuhnya optimal. Dari segi biaya, rata-rata pengeluaran per hari untuk memenuhi kebutuhan gizi tersebut adalah sebesar Rp 40.666,92, dengan rata-rata biaya untuk satu kali makan sebesar Rp 13.555,64. Selain itu, variasi menu yang dihasilkan memiliki rata-rata nilai 11,92, yang mencerminkan adanya keberagaman menu dalam hasil optimasi. Data ini menunjukkan bahwa algoritma *PSO* tidak hanya memperhatikan aspek gizi, tetapi juga mempertimbangkan efisiensi biaya dan variasi menu, yang merupakan elemen penting dalam perancangan menu makanan yang optimal. Untuk hasil perhitungan secara rinci, dapat dilihat di Lampiran 26.