

LAPORAN AKHIR

OPTIMASI POLA IRIGASI LAHAN PERTANIAN MENGUNAKAN ALGORITMA *GRADIENT DESCENT* UNTUK EFISIENSI DISTRIBUSI AIR DAN PENINGKATAN HASIL PANEN



Anggota Kelompok 6:

Farrelius Kevin	00000081783
Genadi Ikhsan Jaya	00000080773
Merhandes Gunawan	00000081070
Nadhila Citra	00000072495

**KELAS IF541 - F
SEMESTER GASAL 2024-2025**

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN INFORMATIKA
UNIVERSITAS MULTIMEDIA NUSANTARA
TANGERANG**

2024

BAB 1

PROBLEM DESCRIPTION

1.1 Latar Belakang

Irigasi merupakan salah satu komponen penting dalam mendukung produktivitas pertanian, terutama di negara agraris seperti Indonesia. Sistem irigasi modern menjadi solusi penting dalam mendukung produktivitas pertanian melalui distribusi air yang efisien dan terencana. Efisiensi distribusi air irigasi tidak hanya memengaruhi hasil panen tetapi juga keberlanjutan sumber daya air. Namun, tantangan utama yang dihadapi dalam penerapan sistem ini adalah menentukan pola irigasi yang paling optimal untuk berbagai kondisi lahan dan kebutuhan tanaman.[1], [2].

Metode pengambilan keputusan berbasis *ELECTRE* (*Elimination and Choice Translating Reality*) menawarkan pendekatan sistematis untuk memilih pola irigasi terbaik dengan mempertimbangkan berbagai kriteria, seperti jenis tanah, kebutuhan air tanaman, serta kondisi iklim. Namun, meskipun *ELECTRE* efektif dalam menentukan alternatif terbaik berdasarkan kriteria multi-dimensional, metode ini masih memerlukan langkah optimasi lanjutan untuk memastikan distribusi air yang paling efisien.

Di sisi lain, algoritma *Gradient Descent* memainkan peran kunci dalam mengoptimalkan distribusi air dengan cara meminimalkan kesalahan perencanaan irigasi. Algoritma ini bekerja secara iteratif dengan menyesuaikan parameter distribusi air agar mencapai konfigurasi optimal yang memaksimalkan efisiensi. Dengan menggabungkan kemampuan analisis multi-kriteria *ELECTRE* untuk pemilihan pola irigasi awal dan *Gradient Descent* untuk proses optimasi selanjutnya, pendekatan ini dapat meningkatkan efisiensi sistem irigasi modern secara signifikan. Hasil yang diharapkan adalah distribusi air yang lebih akurat, pengurangan pemborosan sumber daya, dan peningkatan hasil panen secara nyata.

Berbagai penelitian terdahulu telah dilakukan untuk mengoptimalkan distribusi air irigasi. Salah satunya adalah penggunaan algoritma Bat untuk mengoptimalkan sistem pengaturan air berbasis *Proportional-Integral-Derivative* (PID), yang terbukti mampu meningkatkan efisiensi sistem irigasi [1]. Selain itu, pendekatan program linier telah digunakan untuk menentukan pola tanam yang optimal, yang berdampak langsung pada kebutuhan dan distribusi air [2]. Metode optimasi lain seperti algoritma genetika menunjukkan potensi signifikan dalam

meningkatkan efisiensi air irigasi bawah permukaan. Hasilnya tidak hanya mengurangi pemborosan air tetapi juga meningkatkan produktivitas tanaman [3]. Di sisi lain, penelitian menggunakan jaringan syaraf tiruan (JST) untuk penentuan pola sistem irigasi menunjukkan bahwa metode ini efektif dalam menyesuaikan pola distribusi air berdasarkan kebutuhan tanaman, terutama untuk kondisi lahan tertentu seperti di Kabupaten Pesisir Selatan, Sumatra Barat [4]. Sementara itu, teknologi irigasi cerdas berbasis algoritma *Ant Colony Optimization* (ACO) telah diaplikasikan untuk sistem irigasi drip, yang memberikan efisiensi melalui analisis data kelembaban tanah dan pola distribusi air [5].

Berdasarkan gambaran permasalahan dari penelitian terdahulu terkait penerapan algoritma optimasi berbasis *Gradient Descent* dalam menentukan pola irigasi masih terbatas, maka penulis mencoba mencari solusi terbaik untuk pemilihan sistem irigasi modern menggunakan *Gradient descent*. Hal ini mendasari penulis untuk menulis laporan penelitian dengan judul “**Optimasi Pola Irigasi Lahan Pertanian Menggunakan Algoritma *Gradient Descent* Untuk Efisiensi Distribusi Air dan Peningkatan Hasil Panen**”

1.2 Rumusan Masalah

1. Bagaimana algoritma *Gradient Descent* dapat digunakan untuk mengoptimalkan sistem pola irigasi modern sehingga meningkatkan efisiensi distribusi air dan memaksimalkan hasil panen?
2. Bagaimana metode *ELECTRE* dapat membantu dalam menentukan pola irigasi modern yang fleksibel berdasarkan berbagai kriteria lahan pertanian, sebagai langkah pendukung optimasi?

1.3 Tujuan Penelitian

1. Menerapkan algoritma *Gradient Descent* untuk mengoptimalkan pola irigasi modern yang meningkatkan efisiensi distribusi air dan hasil panen.
2. Mengintegrasikan metode *ELECTRE* untuk mendukung proses pengambilan keputusan dalam menentukan pola irigasi berdasarkan kriteria lahan pertanian.

1.4 Batasan Masalah

Agar penelitian ini tetap terfokus dan dapat diselesaikan secara optimal, ditetapkan beberapa batasan masalah sebagai berikut:

1. Penelitian ini hanya memanfaatkan metode algoritma *Gradient Descent* dan bantuan *Multi-Criteria Decision Making* (MCDM) *ELECTRE* untuk optimasi pola sistem irigasi modern dan pemilihan solusi terbaik.
2. Penelitian ini terbatas pada pembahasan sistem irigasi modern, tidak mencakup sistem irigasi tradisional atau semi-modern.
3. Kriteria yang digunakan dalam pemilihan pola irigasi penelitian ini menggunakan: topografi, biaya implementasi, efisiensi air, jenis tanah, jenis tanaman, ketersediaan air
4. Data yang digunakan dalam penelitian ini disintesis berdasarkan referensi teoretis, literatur ilmiah, dan pengetahuan yang relevan, sehingga tidak menggunakan data primer atau lapangan.
5. Penelitian ini tidak dibatasi pada wilayah tertentu, sehingga hasilnya bersifat umum dan dapat diaplikasikan pada berbagai kondisi lahan pertanian.

1.5 Manfaat Penelitian

1. Memberikan solusi berbasis algoritma *Gradient Descent* untuk meningkatkan produktivitas pertanian melalui pola irigasi modern yang efisien.
2. Menawarkan metode pengambilan keputusan yang sistematis dan fleksibel melalui integrasi metode *ELECTRE* sebagai alat analisis kriteria lahan.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

BAB 2

THEORY - ALGORITHM - METHOD

2.1 Landasan Teori

2.1.1 Irigasi

Irigasi adalah proses penyediaan, pengaturan, dan pembuangan air untuk mendukung kegiatan pertanian. Jenis-jenis irigasi meliputi irigasi permukaan, irigasi rawa, irigasi air bawah tanah, irigasi pompa, dan irigasi tambak [6]. Tujuan irigasi adalah untuk meningkatkan produktivitas pertanian, mendukung ketahanan pangan nasional, dan meningkatkan kesejahteraan petani. Irigasi dibangun melalui jaringan yang menyuplai air ke lahan pertanian, yang disebut Daerah Irigasi. Jaringan irigasi terdiri dari saluran, bangunan utama, dan pelengkap yang diperlukan untuk mendistribusikan, mengatur, dan membuang air irigasi [6].

Terdapat tiga jenis jaringan irigasi: jaringan irigasi primer, sekunder, dan tersier. Jaringan irigasi primer mencakup saluran induk atau utama, bangunan bagi, saluran pembuangan, serta bangunan pelengkap lainnya. Jaringan irigasi sekunder terdiri dari saluran sekunder dan pembuangannya, bangunan bagi, bangunan sadap, dan bangunan pelengkap. Sementara itu, jaringan irigasi tersier berfungsi untuk mendistribusikan air ke petak tersier, termasuk saluran tersier, saluran kuarter, boks tersier, boks kuarter, dan bangunan pelengkap lainnya [6].

2.1.2 Irigasi Modern

Irigasi modern adalah teknik penyiraman tanaman yang menggunakan teknologi canggih untuk mengelola dan mendistribusikan air secara efisien, dengan tujuan meningkatkan hasil pertanian, meminimalkan pemborosan air, dan memastikan tanaman mendapatkan jumlah air yang tepat [7]. Sistem irigasi ini tidak hanya mengutamakan perbaikan aspek fisik, tetapi juga mencakup penguatan kelembagaan pengelolaan serta pengembangan sumber daya manusia. Pendekatan berbasis teknologi dan partisipasi masyarakat dalam pengelolaan irigasi dilakukan secara partisipatif melalui musyawarah, yang memungkinkan penetapan hak dan kewajiban secara terbuka dan tanpa diskriminasi. Tujuan utamanya mencapai keadilan dan kepuasan bagi semua pihak terkait, terutama petani, serta memberikan pelayanan yang lebih baik dalam mendukung pertanian dan kesejahteraan petani [8].

Teknologi menjadi unsur kunci dalam irigasi modern, yang bertujuan untuk meningkatkan efisiensi penggunaan air, mengurangi pemborosan, dan memastikan distribusi air yang optimal ke tanaman. Teknologi yang digunakan dalam irigasi modern mencakup sistem otomatis berbasis sensor, yang memungkinkan pemantauan kelembaban tanah secara *real-time* dan pengelolaan berbasis data. Dengan teknologi ini, sistem irigasi seperti irigasi tetes dan irigasi *sprinkler* dapat diimplementasikan secara lebih tepat dan efisien, disesuaikan dengan kebutuhan spesifik tanaman dan kondisi lingkungan setempat. Dengan menggunakan sistem yang canggih dan berbasis data, irigasi modern tidak hanya mengoptimalkan penggunaan air, tetapi juga meningkatkan produktivitas pertanian dan keberlanjutan sumber daya alam [8].

2.1.3 Sistem Irigasi Modern

Berikut ini adalah jenis-jenis tipe dari sistem irigasi modern:

1. **Sistem Irigasi Permukaan (*Surface Irrigation System*)**

Sistem irigasi permukaan adalah metode tradisional di mana air dialirkan di permukaan tanah dan meresap ke dalam tanah. Metode ini biasanya digunakan pada lahan datar dan memiliki biaya yang rendah serta sederhana dalam pelaksanaan. Meskipun efisien untuk beberapa kondisi, sistem ini dapat menyebabkan pemborosan air jika tidak dikelola dengan baik.

2. **Sistem Irigasi Sprinkler (*Sprinkler Irrigation System*)**

Sistem irigasi sprinkler menyemprotkan air ke udara menggunakan nozel, menyerupai hujan alami. Metode ini sangat efektif untuk berbagai jenis tanaman dan dapat mencakup area yang luas. Kelebihan dari sistem ini adalah distribusi air yang merata dan kemampuannya untuk digunakan dalam berbagai kondisi tanah.

3. **Sistem Irigasi Tetes (*Drip Irrigation System*)**

Sistem irigasi tetes mengalirkan air perlahan langsung ke akar tanaman melalui pipa dengan lubang kecil. Metode ini sangat efisien dalam penggunaan air karena mengurangi pemborosan dan memastikan bahwa air diberikan tepat di area akar. Selain itu, sistem ini membantu mengurangi risiko penyakit tanaman karena kelembapan yang lebih rendah di permukaan tanah.

4. **Sistem Irigasi Subsurface (*Subsurface Irrigation System*)**

Sistem irigasi subsurface menggunakan pipa yang dipasang di bawah permukaan tanah untuk memberikan air langsung ke akar tanaman. Metode ini mengurangi penguapan dan menjaga kelembapan tanah, sehingga sangat cocok untuk tanaman yang sensitif terhadap kelembapan di permukaan. Dengan cara ini, tanaman mendapatkan pasokan air yang konsisten dan efisien.

5. **Sistem Irigasi Pivot Tengah (*Center Pivot Irrigation System*)**

Sistem irigasi pivot tengah adalah sistem bergerak yang secara otomatis menyiram area pertanian dengan lengan berputar. Metode ini menawarkan efisiensi tinggi dan cakupan area yang luas, sehingga sangat cocok untuk lahan pertanian besar. Dengan kontrol yang baik atas jumlah air yang diberikan, sistem ini membantu meningkatkan hasil panen secara signifikan.

6. **Sistem Irigasi Gerak Samping (*Lateral Move Irrigation System*)**

Sistem irigasi gerak samping bergerak secara horizontal untuk menyiram area pertanian. Metode ini memberikan distribusi air yang merata dan dapat disesuaikan dengan kebutuhan tanaman. Keunggulan dari sistem ini adalah fleksibilitas dalam pengaturan dan efisiensi dalam penggunaan air.

7. **Sistem Irigasi Hujan (*Rain Irrigation System*)**

Sistem irigasi hujan menggunakan alat untuk menyemprotkan air ke tanaman seperti hujan alami. Metode ini memungkinkan penyiraman yang seragam di seluruh area dan cocok untuk berbagai jenis tanaman. Dengan cara ini, kelembapan tanah dapat terjaga dengan baik, mendukung pertumbuhan tanaman yang optimal.

8. **Sistem Irigasi Kabut (*Fog Irrigation System*)**

Sistem irigasi kabut menggunakan kabut untuk menyiram tanaman dengan cara yang sangat halus. Metode ini sangat efisien dalam mengurangi penguapan dan ideal untuk tanaman yang sensitif terhadap kelembapan. Dengan aplikasi air yang lembut, tanaman dapat mendapatkan kelembapan yang diperlukan tanpa risiko kerusakan akibat penyiraman yang berlebihan.

2.1.4 Gradient Descent

Gradient Descent adalah algoritma optimasi yang digunakan untuk meminimalkan fungsi biaya $J(\theta)$ dengan memperbarui parameter θ secara iteratif menuju

arah negatif dari gradien. Pada setiap iterasi, gradien fungsi dihitung sebagai turunan parsial $\frac{\partial J(\theta)}{\partial \theta}$, lalu parameter diperbarui menggunakan formula:

$$\theta_{k+1} = \theta_k - \alpha \frac{\partial J(\theta)}{\partial \theta},$$

di mana α adalah *learning rate* yang mengontrol besarnya langkah pembaruan. Proses ini berlanjut hingga perbedaan nilai fungsi biaya antara iterasi berurutan lebih kecil dari ambang batas konvergensi ε atau iterasi maksimum tercapai. Dengan pendekatan ini, algoritma dapat menemukan nilai optimal θ^* yang meminimalkan $J(\theta)$ [9].

Algorithm 1: Pseudocode *Gradient Descent*

Input: $J(\theta), \alpha$
Output: θ^*
initialize (θ);
count $\leftarrow 0$;
while *count* $< k$ **or** $|J(\theta_{k+1}) - J(\theta_k)| > \varepsilon$ **do**
 do $\Delta J(\theta_k) \leftarrow -\alpha \partial / \partial \theta_k J(\theta_k)$;
 $\theta_{k+1} \leftarrow \theta_k + \alpha \Delta J(\theta_k)$;
 calculate $J(\theta_{k+1})$;
 $k \leftarrow k + 1$;
 $\theta^* \leftarrow \theta_{k+1}$;
end
return θ^* ;

2.1.5 *Elimination and Choice Translation Reality (ELECTRE)*

Electre didasarkan pada konsep perankingan melalui perbandingan berpasangan antar alternative pada criteria yang sesuai. Suatu alternative dikatakan mendominasi alternatif yang lainnya jika satu atau lebih kriterianya melebihi dan sama dengan criteria lain yang tersisa.

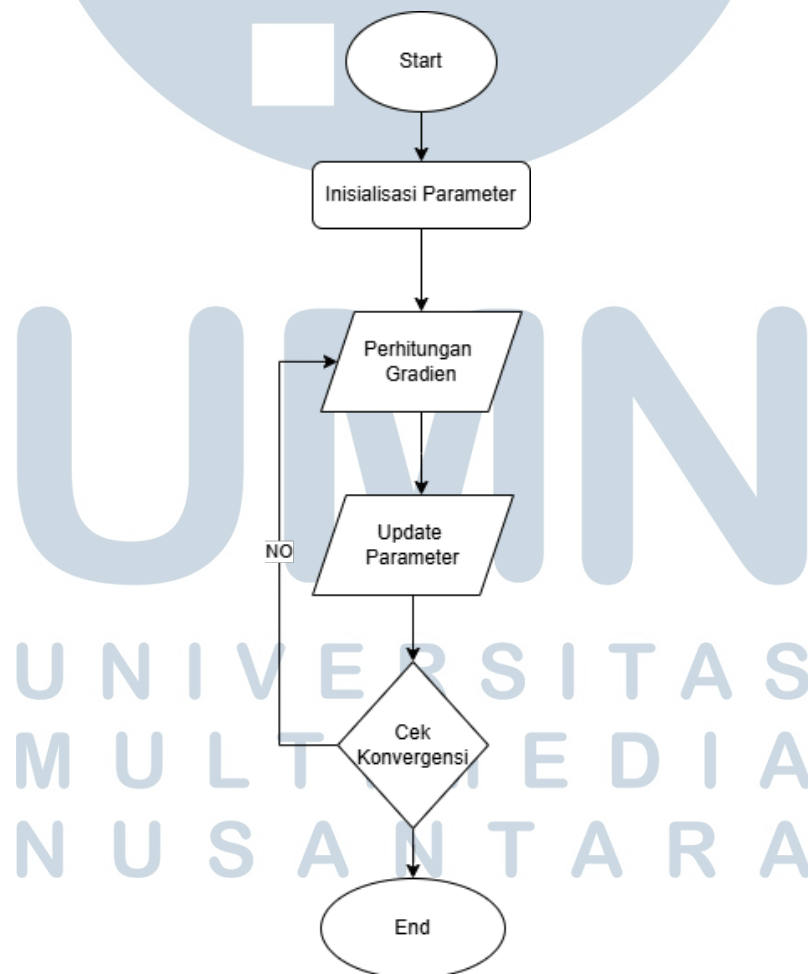
Metode *electre* termasuk metode analisis pengambilan keputusan multikriteria yang berasal dari Eropa pada tahun 1960an. *ELECTRE* didasarkan pada konsep outranking dengan menggunakan perbandingan berpasangan dari alternatif berdasarkan setiap kriteria yang sesuai.

Suatu alternatif mendominasi alternatif lain jika satu atau lebih kriteria melebihi dibandingkan dengan kriteria dari alternatif lain, dan sama dengan kriteria lain yang tersisa. Misalkan terdapat hubungan perankingan alternatif A_k dan

A_i . Jika alternatif ke- k tidak mendominasi alternatif ke- i secara kuantitatif, maka pengambil keputusan lebih baik mengambil risiko A_k daripada A_i .

Metode *ELECTRE* melakukan perbandingan berpasangan antara semua alternatif untuk setiap atribut secara terpisah dalam rangka untuk mengembangkan hubungan outranking antara alternatif. Metode ini pada pertama- pertama menghilangkan alternatif yang kurang diinginkan kemudian menggunakan *complimentary analysis* untuk memilih alternatif terbaik. Karena perbandingan berlangsung antara alternatif yang tersedia maka akan konsep *ELECTRE* dimana membandingkan alternatif dengan beberapa set referensi nilai untuk melihat nilai parameter yang diinginkan. Suatu alternatif mendominasi alternatif lain jika satu atau lebih kriteria melebihi dibandingkan dengan kriteria dari alternatif lain [10].

2.1.6 Algoritma Gradient Descent



Gambar 2.1. Flowchart *Gradient Descent*

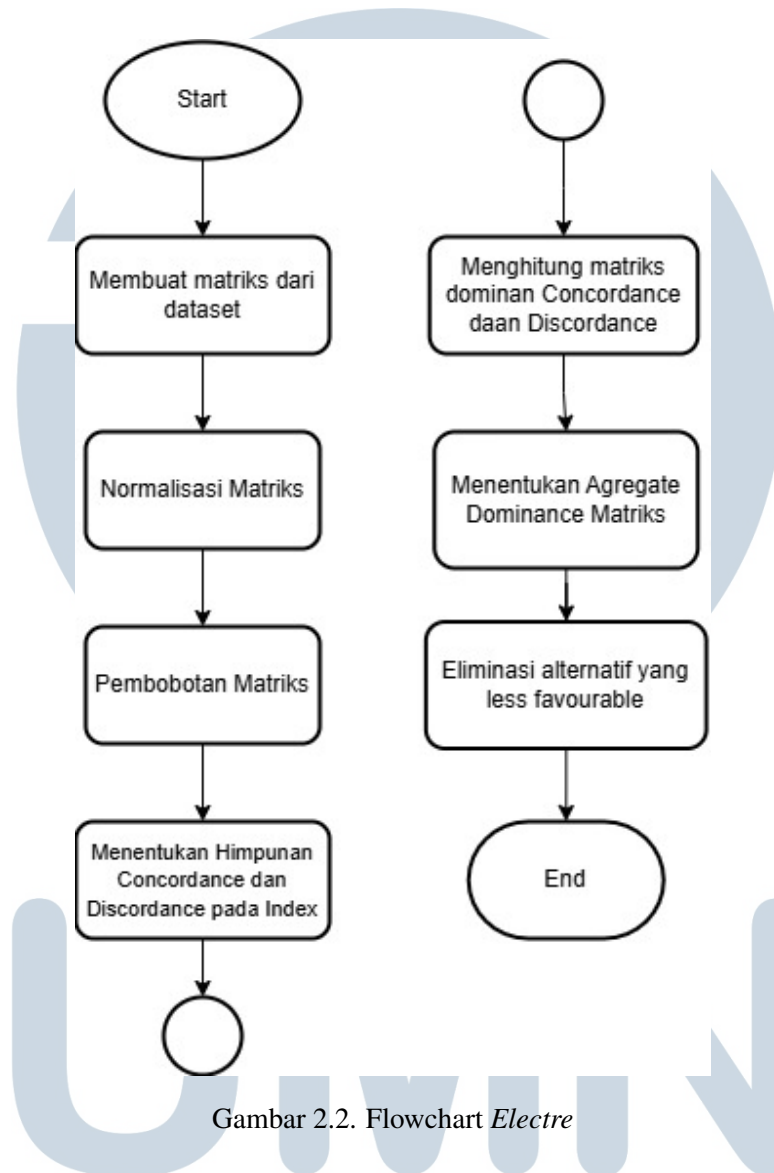
$$x_{n+1} = x_n - \alpha \nabla f(x_n) \quad (2.1)$$

Langkah-langkah yang dilakukan dalam penyelesaian masalah menggunakan metode *Gradient Descent* adalah sebagai berikut:

1. **Inisialisasi Parameter.** Menetapkan nilai awal untuk parameter seperti bobot w dan bias b , yang dapat diinisialisasi secara acak atau dengan nilai tertentu, misalnya nol. Pada tahap ini, juga ditentukan nilai *learning rate* (α) yang berfungsi mengatur seberapa besar perubahan parameter pada setiap iterasi.
2. **Perhitungan Gradien.** Menghitung turunan parsial dari fungsi kerugian (*loss function*) terhadap setiap parameter. Gradien menunjukkan arah perubahan fungsi kerugian terhadap parameter tersebut. Dengan kata lain, gradien memberikan informasi tentang arah mana parameter harus diperbarui agar fungsi kerugian berkurang. Fungsi kerugian ini biasanya berbeda tergantung pada masalah, seperti *Mean Squared Error* (MSE) untuk regresi atau *Cross-Entropy* untuk klasifikasi.
3. **Update Parameter.** dengan mengurangi nilai gradien yang dikalikan dengan *learning rate* dari parameter yang ada. Proses ini memastikan parameter bergerak ke arah minimum fungsi kerugian.
4. **Cek Konvergensi.** Langkah-langkah ini kemudian diulang dalam iterasi hingga tercapai kondisi konvergensi. Konvergensi terjadi ketika perubahan nilai fungsi kerugian menjadi sangat kecil, menandakan bahwa parameter telah mencapai nilai optimal, atau hingga jumlah iterasi maksimum yang ditentukan sebelumnya tercapai.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

2.1.7 Metode *Electre*



Gambar 2.2. Flowchart *Electre*

Langkah-langkah yang dilakukan dalam penyelesaian masalah menggunakan metode *ELECTRE* adalah sebagai berikut:

1. **Menentukan alternatif, kriteria, rating kecocokan dan tingkat kepentingan**, masing-masing kriteria diberikan 1 sampai 5. *Rating* kecocokan dan tingkat kepentingan diberi nilai dari 1 sampai 5 untuk menentukan nilai dari setiap alternatif pada setiap kriteria,
2. **Normalisasi Matriks Keputusan**. Menentukan Normalisasi matriks keputusan membutuhkan proses ke suatu skala yang dapat diperbandingkan dengan

semua rating alternatif yang ada. Sehingga, dapat mempermudah pemodifikasi data dan mengurangi kompleksitas. Dalam prosedur ini, setiap atribut diubah menjadi nilai yang *comparable*.

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^m x_{ij}^2}} \quad (2.2)$$

3. **Pembobotan matriks yang dinormalisasi.** Setelah dinormalisasi, setiap kolom dari matriks R dikalikan dengan bobot-bobot (w_j) yang ditentukan oleh pembuat keputusan. Sehingga terbentuk matriks kedua yaitu matriks V.

$$V = R \cdot W \quad (2.3)$$

4. **Pembentukan himpunan matriks *concordance* dan *discordance index*.** Untuk menentukan himpunan matriks *concordance* dan himpunan matriks *discordance* dalam sebuah kriteria dalam alternatif termasuk *discordance* yaitu untuk setiap pasang dari alternative k dan l ($k, l = 1, 2, 3, \dots, m$ dan $k \neq l$) kumpulan J kriteria dibagi menjadi dua himpunan bagian.

a. *Concordance*

$$C_{kl} = \{j \mid v_{kj} \geq v_{lj}\} \quad \text{untuk } j = 1, 2, 3, \dots, n \quad (2.4)$$

$$C_{ij} = w_1 + w_2 + w_4 + w_5 + \dots \quad (2.5)$$

Keterangan:

- C_{kl} : Himpunan elemen j yang memenuhi syarat $v_{kj} \geq v_{lj}$.
- v_{kj} : Nilai yang dibandingkan dengan v_{lj} untuk elemen tertentu j .
- $j = 1, 2, 3, \dots, n$: Indeks yang didefinisikan dalam rentang tertentu.

b. *Discordance*

$$D_{kl} = \{j \mid v_{kj} < v_{lj}\} \quad \text{untuk } j = 1, 2, 3, \dots, n \quad (2.6)$$

$$d_{kl} = \frac{\max \{|v_{kj} - v_{lj}|\}_{j \in D_{kl}}}{\max \{|v_{kj} - v_{lj}|\}_{j \notin D_{kl}}} \quad (2.7)$$

Keterangan:

- D_{kl} : Himpunan elemen j yang memenuhi syarat $v_{kj} < v_{ij}$.
- v_{kj} : Nilai yang dibandingkan dengan v_{ij} untuk elemen tertentu j .
- $j = 1, 2, 3, \dots, n$: Indeks yang didefinisikan dalam rentang tertentu.

5. **Pembentukan matriks concordance dan discordance.** Matriks *concordance* diperoleh dari himpunan *concordance* dengan menjumlahkan bobot yang termasuk dalam himpunan *concordance* sedangkan matriks *discordance* diperoleh dari elemen elemen dengan membagi maksimum selisih kriteria yang termasuk ke dalam himpunan bagian *Discordance* dengan maksimum selisih nilai seluruh kriteria yang ada.

a. *Concordance*

$$f_{kl} = \begin{cases} 1, & \text{jika } c_{kl} \geq \underline{c} \\ 0, & \text{jika } c_{kl} < \underline{c} \end{cases} \quad (2.8)$$

$$\underline{c} = \frac{\sum_{k=1}^n \sum_{l=1}^n c_{kl}}{m(m-1)} \quad (2.9)$$

b. *Discordance*

$$g_{kl} = \begin{cases} 1, & \text{jika } d_{kl} \geq \underline{d} \\ 0, & \text{jika } d_{kl} < \underline{d} \end{cases} \quad (2.10)$$

$$\underline{d} = \frac{\sum_{k=1}^n \sum_{l=1}^n d_{kl}}{m(m-1)} \quad (2.11)$$

6. **Pembentukan matriks dominan concordance dan discordance.** Nilai nilai dari matriks dominan *concordance* dan *discordance* atau matriks F diperoleh dengan membandingkan setiap nilai elemen matriks *concordance* dengan nilai *threshold* dan pada nilai-nilai dari matriks dominan *discordance* atau matriks G diperoleh dengan membandingkan setiap nilai elemen matriks *discordance* dengan nilai *threshold* dari matriks *discordance*.

a. *Concordance*

$$C_{kl} \geq \underline{c} \quad (2.12)$$

b. *Discordance*

$$d_{kl} \geq \underline{d} \quad (2.13)$$

7. **Pembentukan matriks dominance agreggate.** Matriks *Aggregate Dominance* matriks yaitu untuk mengetahui pilihan yang akan ditentukan. Semakin

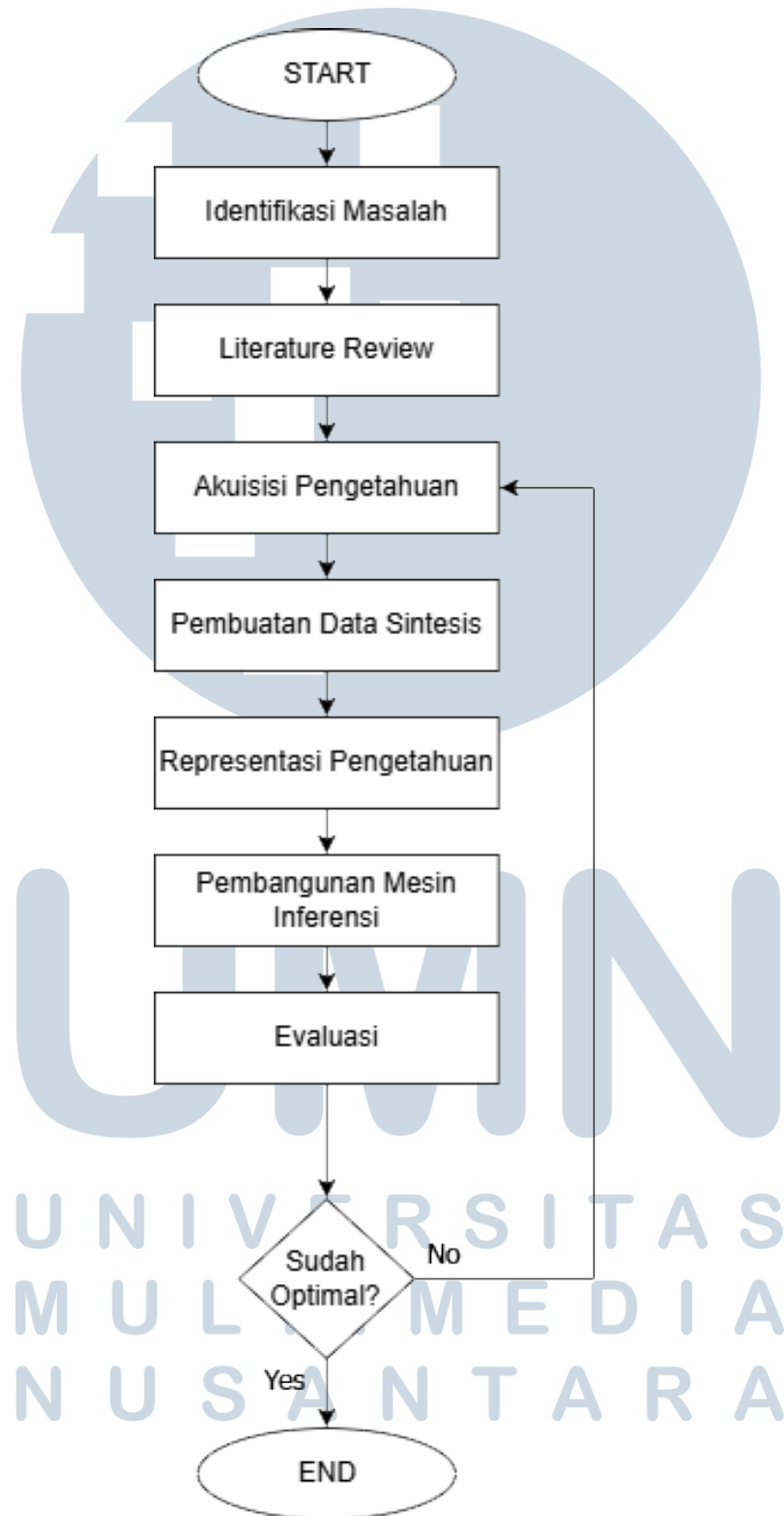
banyak nilai $e_{kl} = 1$ maka alternatif A_k merupakan alternatif yang lebih baik.

$$e_{kl} = f_{kl} \times g_{kl} \quad (2.14)$$

8. **Eliminasi alternatif yang *less favourable*.** Matriks E atau matriks *dominance aggregate* memberikan urutan pilihan dari setiap alternatif, yaitu bila $e_{kl} = 1$ maka alternatif A_k merupakan alternatif yang lebih baik daripada A_l . Sehingga, baris dalam matriks E yang memiliki jumlah $e_{kl} = 1$ paling sedikit dapat dieliminasi sehingga didapat pemilihan terbaik yang ditentukan akan tetapi jika nilai saling mendominasi maka dilakukan dengan cara perbandingan nilai *weight normalized* atau matriks v .
9. **Uji matriks *Aggregate Dominance*.** Matriks Uji matriks *Aggregate Dominance* Matriks dilakukan apabila ada nilai 1 yang sama, maka itu perlu diuji dengan nilai *Weight Normalized* untuk mengetahui hasil dari dua atau lebih. Jika tidak ada yang sama, maka tidak perlu diuji dengan *Weight Normalized* karena hasilnya telah didapatkan dari matriks terakhir atau *aggregate dominance matriks*.



2.2 Metode Penelitian



Gambar 2.3. Flowchart Metode Penelitian

2.2.1 Identifikasi Masalah

Pada tahap awal penelitian, dimulai dengan mencari masalah yang ada. Didapati masalah utama yang dihadapi adalah bagaimana mengoptimalkan sistem pola irigasi untuk meningkatkan efisiensi distribusi air dan hasil panen. Penelitian ini menggunakan pendekatan *Gradient Descent* untuk optimasi dan metode *ELECTRE* sebagai alat pengambilan keputusan multikriteria.

2.2.2 Literature Review

Studi literatur dilakukan dengan mencari dan mengkaji jurnal-jurnal serta teori-teori yang relevan dengan topik penelitian. Kajian literatur bertujuan untuk memahami pendekatan-pendekatan yang telah digunakan dalam pengelolaan sistem irigasi serta metode yang terbukti efektif dalam meningkatkan efisiensi distribusi air. Selain itu, studi ini juga mencakup eksplorasi algoritma yang relevan dengan penelitian, yaitu menggunakan *Gradient Descent* dan *Multiple Criteria Decision Making* (MCDM) menggunakan metode *ELECTRE*. *Gradient Descent* akan dianalisis untuk memahami penerapannya dalam optimasi bobot kriteria, sedangkan *ELECTRE* akan dikaji sebagai metode pengambilan keputusan multi-kriteria untuk menilai dan menentukan alternatif terbaik dalam sistem irigasi.

2.2.3 Akuisisi Pengetahuan

Proses akuisisi pengetahuan melibatkan pengumpulan data dan informasi yang relevan untuk memahami faktor-faktor yang memengaruhi distribusi air dan hasil panen. Pengetahuan yang diperoleh mencakup elemen-elemen kunci seperti kondisi tanah, kebutuhan air tanaman, dan parameter lingkungan lainnya. Informasi ini digunakan untuk merumuskan model awal serta mendukung proses pengambilan keputusan berbasis data. Kriteria yang digunakan dalam pemilihan pola irigasi meliputi topografi, biaya implementasi, efisiensi air, jenis tanah, jenis tanaman, dan ketersediaan air. Jika sistem optimasi belum mencapai hasil yang optimal, proses ini akan mengalami *looping*, yaitu pengulangan proses pengumpulan dan pengolahan data hingga diperoleh pengetahuan yang lebih akurat dan memadai.

2.2.4 Pembuatan Data Sintesis

Data yang dikumpulkan diolah untuk menjadi bentuk yang siap digunakan dalam algoritma. Karena data asli tidak tersedia, data yang digunakan disusun secara sintesis dengan membuat sendiri berdasarkan asumsi-asumsi yang relevan dan literatur pendukung. Setelah data sintesis dibuat, langkah-langkah seperti normalisasi data, pemilihan parameter kunci, dan penyusunan dataset dilakukan untuk memastikan data siap digunakan dalam simulasi optimasi.

2.2.5 Representasi Pengetahuan

Pengetahuan yang diperoleh dari data dan literatur dirumuskan dalam bentuk model matematis yang menggambarkan hubungan antarvariabel kunci, seperti efisiensi penggunaan air, hasil panen, dan biaya operasional. Model ini mencakup fungsi tujuan yang dirancang untuk meminimalkan pemborosan air sekaligus memaksimalkan hasil panen. Fungsi tujuan tersebut mempertimbangkan berbagai faktor, seperti efisiensi air sebagai variabel pengurangan pemborosan, serta batasan anggaran dan hasil minimum sebagai kendala optimasi. Model matematis ini menjadi landasan dalam merancang algoritma optimasi yang diimplementasikan menggunakan bahasa pemrograman Python untuk memastikan solusi optimal dapat ditemukan dengan pendekatan yang sistematis dan terukur.

2.2.6 Pembangunan Mesin Inferensi

1. Tahap Optimasi Bobot Kriteria dengan Algoritma *Gradient Descent*

1. Inisialisasi Parameter Awal

- Tetapkan nilai awal parameter seperti bobot X_n secara acak atau berbasis data *historis*. Parameter ini akan menjadi dasar optimasi distribusi air.
- Tetapkan nilai *learning rate* α untuk mengendalikan kecepatan pembaruan parameter selama proses iterasi.

2. Perhitungan Gradien

- Hitung gradien parsial dari fungsi tujuan $f(X_n)$ terhadap setiap parameter, dengan memanfaatkan fungsi objektif yang mencerminkan efisiensi

penggunaan air dan hasil panen (misalnya, *Mean Squared Error* atau fungsi lain yang relevan).

- Gradien ini memberikan arah perubahan yang optimal untuk meminimalkan fungsi tujuan.

3. Pembaruan Parameter

- Perbarui nilai parameter X_n menggunakan rumus berikut:

$$X_{n+1} = X_n - \alpha \nabla f(X_n)$$

di mana $\nabla f(X_n)$ adalah gradien fungsi tujuan pada iterasi ke- n .

- Proses ini diulang hingga parameter X_n mencapai nilai konvergen yang stabil.

4. Cek Konvergensi

- Tentukan apakah perubahan nilai fungsi tujuan antar iterasi telah cukup kecil (di bawah toleransi tertentu). Jika sudah, hasil parameter optimal X^* disimpan untuk langkah berikutnya.

2. Tahap Evaluasi Alternatif dengan MCDM ELECTRE

Setelah bobot optimal diperoleh melalui *Gradient Descent*, hasil ini diproses lebih lanjut menggunakan metode *Multi-Criteria Decision Making (MCDM)*, yaitu *ELECTRE*. Metode *ELECTRE* digunakan untuk mengevaluasi berbagai alternatif pola distribusi air berdasarkan kriteria-kriteria yang telah ditentukan, sehingga memungkinkan pemilihan pola distribusi yang paling efisien dan efektif sesuai dengan kebutuhan.

1. Menentukan Alternatif dan Kriteria

- Definisikan alternatif pola distribusi air yang dihasilkan dari optimasi *Gradient Descent*.
- Tentukan kriteria evaluasi (misalnya, efisiensi air, peningkatan hasil panen, biaya operasional).
- Berikan bobot penting untuk setiap kriteria berdasarkan preferensi pembuat keputusan.

2. Normalisasi Matriks Keputusan

- Normalisasi nilai kinerja alternatif r_{ij} menggunakan rumus berikut:

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^m x_{ij}^2}}$$

3. Pembobotan Matriks

- Hasil normalisasi dikalikan dengan bobot w_j setiap kriteria untuk menghasilkan matriks nilai akhir V :

$$V = R \cdot W$$

4. Pembentukan Matriks *Concordance* dan *Discordance*

- Hitung matriks *concordance* C dan *discordance* D untuk membandingkan alternatif berdasarkan bobot kriteria dan selisih nilai kinerja masing-masing alternatif.

5. Dominasi *Concordance* dan *Discordance*

- Tentukan matriks dominan F dan G dengan membandingkan nilai *concordance* dan *discordance* terhadap ambang batas yang ditentukan.

6. Penentuan Matriks Agregat Dominance

- Hitung matriks *agregat dominance* E untuk mengetahui alternatif dengan performa terbaik:

$$e_{ij} = f_{ij} \cdot g_{ij}$$

7. Eliminasi Alternatif yang Tidak Efisien

- Hapus alternatif dengan nilai agregat dominasi e_{ij} rendah hingga hanya tersisa alternatif terbaik.

8. Perangkingan Alternatif

- Lakukan perangkingan terhadap alternatif yang tersisa berdasarkan nilai agregat dominasi e_{ij} tertinggi.
- Alternatif dengan nilai e_{ij} tertinggi dianggap sebagai sistem irigasi yang paling optimal sesuai dengan kriteria yang telah ditentukan.

2.2.7 Evaluasi

Tahap evaluasi bertujuan untuk menilai hasil optimasi dan memastikan bahwa solusi yang dihasilkan sesuai dengan kriteria yang telah ditetapkan. Setelah Algoritma *Gradient Descent* menghasilkan pola distribusi air, hasilnya divalidasi berdasarkan kriteria seperti efisiensi distribusi air, peningkatan hasil panen, topografi, biaya implementasi, jenis tanah, jenis tanaman, dan ketersediaan air.

Kemudian, metode *ELECTRE* digunakan untuk mengevaluasi dan meranking alternatif-alternatif yang dihasilkan. Langkah-langkah *ELECTRE* meliputi normalisasi data, pemberian bobot pada setiap kriteria, serta perhitungan matriks *concordance* dan *discordance* untuk menentukan indeks dominasi global. Alternatif dengan nilai dominasi tertinggi dianggap sebagai solusi terbaik. Jika hasil evaluasi menunjukkan bahwa solusi belum optimal, proses optimasi diulang dengan penyesuaian parameter atau data input. Dengan demikian, penelitian ini diakhiri dengan alternatif terbaik yang memenuhi semua kriteria dipilih sebagai pola distribusi air yang akan diterapkan.



BAB 3

SOLUTION

3.1 Studi Kasus Penerapan Metode *ELECTRE* dan Algoritma *Gradient Descent*

Perusahaan agrikultur sedang mempertimbangkan penerapan salah satu dari 8 strategi irigasi untuk meningkatkan efisiensi dan produktivitas lahan mereka. Berikut adalah karakteristik lahan yang dimiliki perusahaan:

- **Topografi:** Mayoritas datar, dengan beberapa area berbukit.
- **Jenis Tanaman:** Terdiri dari padi, sayuran, buah-buahan, dan tanaman hortikultura.
- **Ketersediaan Air:** Cukup di musim penghujan, tetapi terbatas di musim kemarau.
- **Jenis Tanah:** Campuran antara tanah berat (lempung) dan tanah ringan (berpasir).
- **Biaya Implementasi:** Perusahaan memiliki batas anggaran moderat untuk implementasi.
- **Efisiensi Air:** Sangat penting, karena keberlanjutan menjadi salah satu tujuan utama perusahaan.

Manajemen perusahaan harus memilih salah satu dari 8 strategi irigasi berikut:

- **Strategi A:** *Surface Irrigation System*
- **Strategi B:** *Sprinkler Irrigation System*
- **Strategi C:** *Drip Irrigation System*
- **Strategi D:** *Subsurface Irrigation System*
- **Strategi E:** *Center Pivot Irrigation*
- **Strategi F:** *Lateral Move Irrigation System*

- **Strategi G:** *Rain Irrigation System*
- **Strategi H:** *Fog Irrigation System*

Bobot untuk Masing-Masing Kriteria

Kriteria	Bobot
Topografi	20%
Biaya Implementasi	30%
Efisiensi Air	20%
Jenis Tanah	10%
Jenis Tanaman	10%
Ketersediaan Air	10%

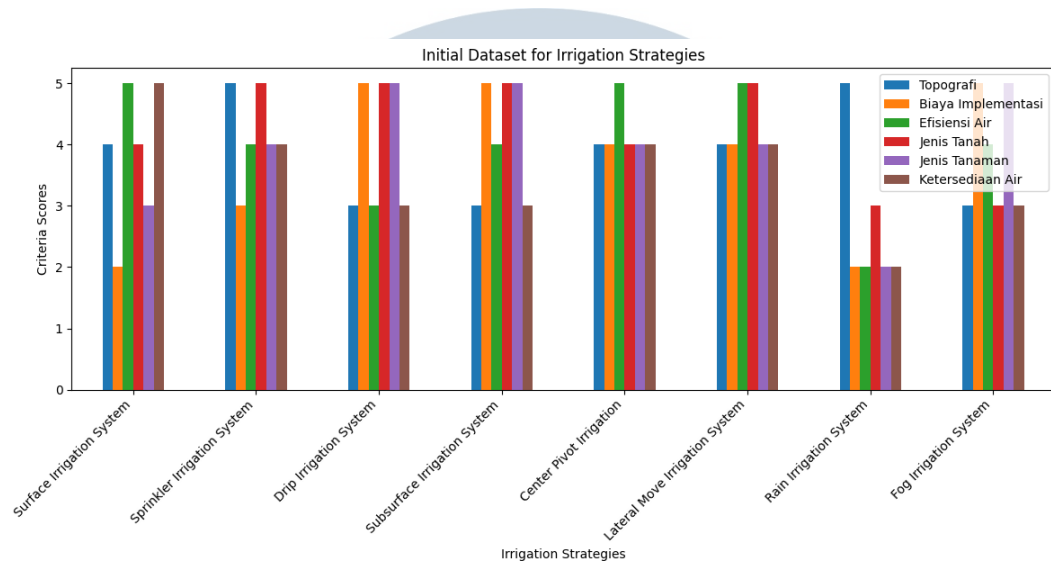
3.2 Hasil Analisis

3.2.1 Pembuatan Matriks Dari Dataset

Kriteria	Topografi (benefit)	Biaya Implementasi (cost)	Efisiensi Air (benefit)	Jenis Tanah (benefit)	Jenis Tanaman (benefit)	Ketersediaan Air (benefit)
<i>Surface Irrigation System</i>	4	2	5	4	3	5
<i>Sprinkler Irrigation System</i>	5	3	4	5	4	4
<i>Drip Irrigation System</i>	3	5	3	5	5	3
<i>Subsurface Irrigation System</i>	3	5	4	5	5	3
<i>Center Pivot Irrigation</i>	4	4	5	4	4	4
<i>Lateral Move Irrigation System</i>	4	4	5	5	4	4
<i>Rain Irrigation System</i>	5	2	2	3	2	2
<i>Fog Irrigation System</i>	3	5	4	3	5	3

Tabel 3.1. Tabel Dataset Sistem Irigasi Modern Berdasarkan Kriteria

3.2.2 Visualisasi Dataset



Gambar 3.1. Visualisasi Dataset

3.2.3 Hasil Optimasi Bobot Menggunakan Gradient Descent

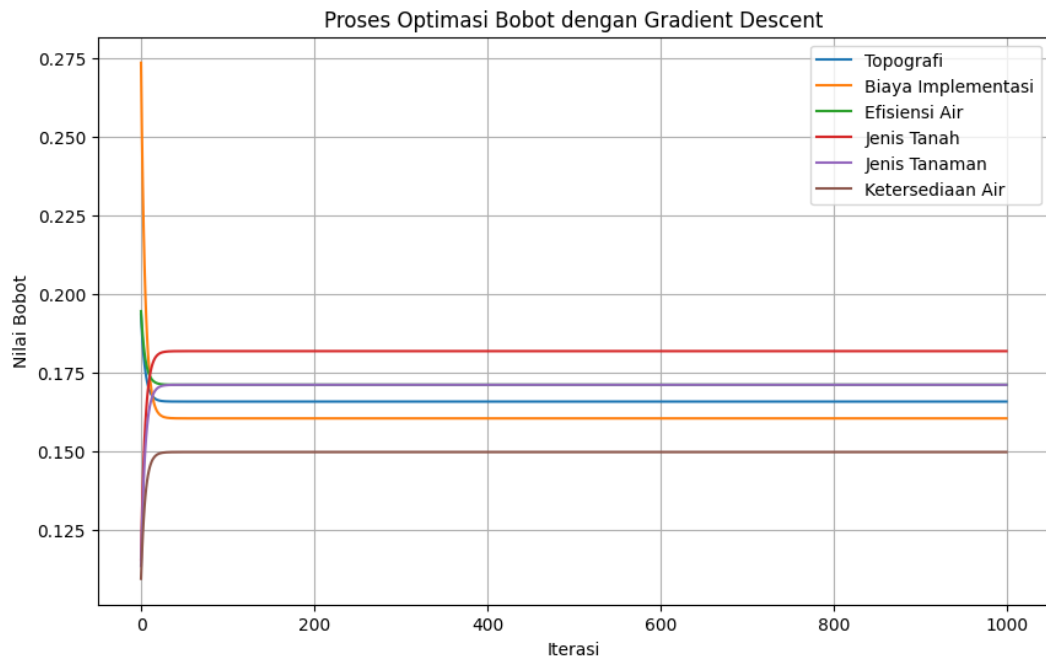
Mencapai iterasi maksimum (1000)

Optimal Weights:

Topografi: 0.1658
 Biaya Implementasi: 0.1604
 Efisiensi Air: 0.1711
 Jenis Tanah: 0.1818
 Jenis Tanaman: 0.1711
 Ketersediaan Air: 0.1497

Gambar 3.2. Hasil Optimasi Bobot

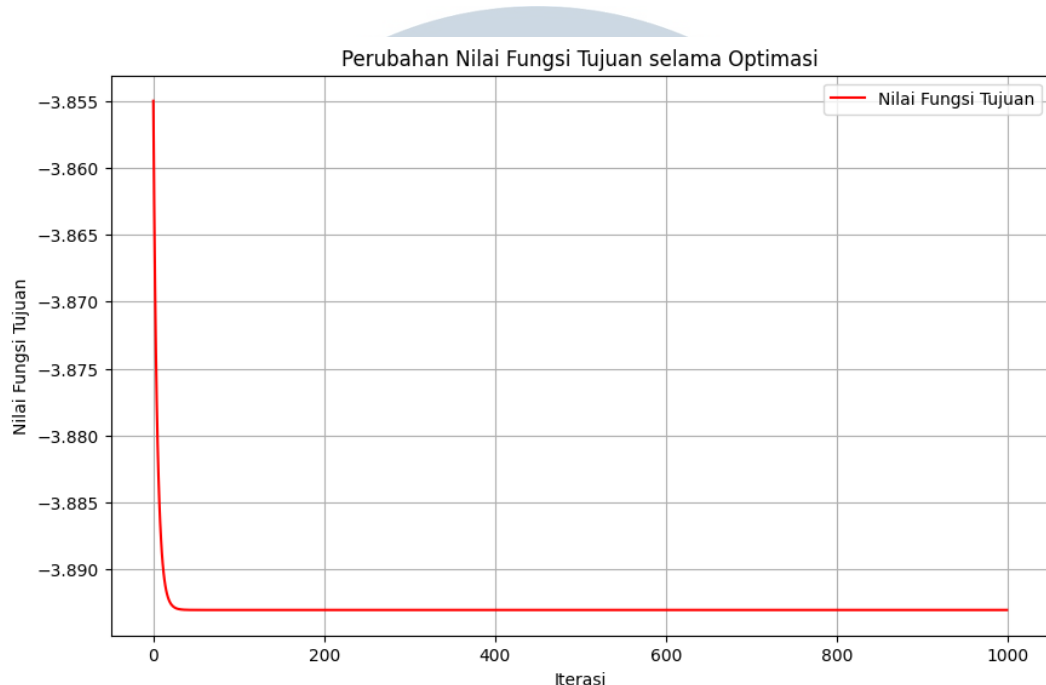
UNIVERSITAS
 MULTIMEDIA
 NUSANTARA



Gambar 3.3. Visualisasi Proses Optimasi Bobot

Setelah melakukan optimasi bobot menggunakan *Gradient Descent*, diperoleh bobot untuk masing-masing kriteria. Dari hasil tersebut, diketahui bahwa kriteria Jenis Tanah memiliki nilai bobot tertinggi sebesar 0.1818. Hal ini menunjukkan bahwa Jenis Tanah adalah kriteria yang paling berpengaruh dalam optimasi ini, atau dapat dikatakan sebagai kriteria yang paling "optimal" dalam memengaruhi model atau fungsi tujuan. Kriteria berikutnya yang memiliki bobot tinggi adalah Efisiensi Air dan Jenis Tanaman, masing-masing dengan nilai 0.1711, disusul oleh Topografi dengan bobot 0.1658, dan Biaya Implementasi sebesar 0.1604. Di sisi lain, kriteria dengan nilai bobot terendah adalah Ketersediaan Air, dengan bobot sebesar 0.1497. Ini menunjukkan bahwa pengaruh Ketersediaan Air terhadap model lebih kecil dibandingkan dengan kriteria lainnya. Dengan demikian, prioritas utama dalam konteks optimasi ini adalah Jenis Tanah, sedangkan Ketersediaan Air memberikan pengaruh paling kecil terhadap model.

3.2.4 Visualisasi Perubahan Nilai Fungsi Tujuan Selama Optimasi



Gambar 3.4. Grafik Perubahan Nilai Fungsi Tujuan

Dari visual grafik diatas dapat dilihat bahwa Fungsi Tujuan yang semakin negatif dalam masalah optimasi adalah karena ingin meminimalkan fungsi tujuan (*loss function*) untuk memperoleh hasil yang lebih baik. Fungsi tujuan semakin negatif menandakan bahwa model sedang menuju solusi yang optimal atau lebih baik. Ketika Fungsi Tujuan tersebut mulai berhenti menurun dan menjadi datar (*flat*) setelah beberapa iterasi, ini memberikan beberapa penanda penting mengenai proses optimasi yang terjadi dengan algoritma *Gradient Descent*. Ketika kurva tersebut mulai datar, menunjukkan bahwa algoritma telah mencapai Titik Konvergensi.

Titik Konvergensi berarti model sudah sangat dekat dengan nilai minimum atau solusi optimal dari fungsi tujuan. Kurva yang tadi menurun kemudian mulai flat hingga iterasi terakhir menunjukkan bahwa algoritma telah menemukan solusi yang stabil dan tidak ada perubahan signifikan lagi. Proses optimasi telah selesai, dan tidak ada perbaikan yang lebih besar yang dapat diperoleh tanpa mengubah parameter lain seperti learning rate.

3.2.5 Matriks Keputusan

	Topografi	Biaya Implementasi	Efisiensi Air	Jenis Tanah	Jenis Tanaman	Ketersediaan Air
Surface Irrigation System	4	2	5	4	3	5
Sprinkler Irrigation System	5	3	4	5	4	4
Drip Irrigation System	3	5	3	5	5	3
Subsurface Irrigation System	3	5	4	5	5	3
Center Pivot Irrigation	4	4	5	4	4	4
Lateral Move Irrigation System	4	4	5	5	4	4
Rain Irrigation System	5	2	2	3	2	2
Fog Irrigation System	3	5	4	3	5	3

Gambar 3.5. Menampilkan Matriks Keputusan

Matriks keputusan ini membandingkan berbagai sistem irigasi berdasarkan beberapa kriteria, termasuk topografi, biaya implementasi, efisiensi air, jenis tanah, jenis tanaman, dan ketersediaan air. Setiap kriteria diberi nilai (skala 1–5) yang mencerminkan seberapa baik sistem tersebut memenuhi kebutuhan tertentu. Misalnya, *Drip Irrigation System* memiliki skor tinggi pada efisiensi air (5) dan kompatibilitas dengan jenis tanah dan tanaman (5), tetapi skor rendah pada ketersediaan air (3). Sebaliknya, *Surface Irrigation System* unggul dalam ketersediaan air (5), namun rendah pada biaya implementasi (2). Matriks ini membantu dalam memilih sistem irigasi yang paling sesuai dengan kebutuhan spesifik lahan, mempertimbangkan efisiensi dan keterbatasan masing-masing metode.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

3.2.6 Normalisasi Matriks Keputusan

	Topografi	Biaya Implementasi	Efisiensi Air	Jenis Tanah	Jenis Tanaman	Ketersediaan Air
Surface Irrigation System	0.357771	0.179605	0.428746	0.326599	0.257248	0.490290
Sprinkler Irrigation System	0.447214	0.269408	0.342997	0.408248	0.342997	0.392232
Drip Irrigation System	0.268328	0.449013	0.257248	0.408248	0.428746	0.294174
Subsurface Irrigation System	0.268328	0.449013	0.342997	0.408248	0.428746	0.294174
Center Pivot Irrigation	0.357771	0.359211	0.428746	0.326599	0.342997	0.392232
Lateral Move Irrigation System	0.357771	0.359211	0.428746	0.408248	0.342997	0.392232
Rain Irrigation System	0.447214	0.179605	0.171499	0.244949	0.171499	0.196116
Fog Irrigation System	0.268328	0.449013	0.342997	0.244949	0.428746	0.294174

Gambar 3.6. Hasil Normalisasi Matriks Keputusan

Dalam tahap ini, dilakukan proses untuk menormalisasi nilai dari setiap alternatif sistem irigasi berdasarkan enam kriteria: topografi, biaya implementasi, efisiensi air, jenis tanah, jenis tanaman, dan ketersediaan air. Normalisasi ini bertujuan untuk menyamakan skala dari semua kriteria sehingga dapat dibandingkan secara objektif meskipun memiliki unit atau skala yang berbeda-beda. Proses normalisasi dilakukan dengan membagi setiap nilai dalam kriteria dengan akar kuadrat dari jumlah kuadrat semua nilai dalam kolom kriteria tersebut.

Hasil normalisasi menunjukkan perbandingan relatif antara sistem irigasi dalam setiap kriteria, misalnya *Surface Irrigation System* memiliki nilai yang relatif tinggi pada efisiensi air (0,428746) dan ketersediaan air (0,490290), sementara *Rain Irrigation System* memiliki nilai yang lebih rendah pada kriteria tersebut (0,171499 dan 0,196116). Dengan normalisasi ini, setiap kriteria memiliki nilai terstandarisasi antara 0 dan 1, sehingga memudahkan untuk melanjutkan ke tahap perhitungan berikutnya dalam proses pengambilan keputusan multi-kriteria.

UNIVERSITAS
MULTIMEDIA
NUSANTARA

3.2.7 Matriks Ternormalisasi Bobot

	Topografi	Biaya Implementasi	Efisiensi Air	Jenis Tanah	Jenis Tanaman	Ketersediaan Air
Surface Irrigation System	0.059318	0.028809	0.073359	0.059376	0.044015	0.073396
Sprinkler Irrigation System	0.074148	0.043213	0.058687	0.074220	0.058687	0.058717
Drip Irrigation System	0.044489	0.072022	0.044015	0.074220	0.073359	0.044038
Subsurface Irrigation System	0.044489	0.072022	0.058687	0.074220	0.073359	0.044038
Center Pivot Irrigation	0.059318	0.057617	0.073359	0.059376	0.058687	0.058717
Lateral Move Irrigation System	0.059318	0.057617	0.073359	0.074220	0.058687	0.058717
Rain Irrigation System	0.074148	0.028809	0.029343	0.044532	0.029343	0.029359
Fog Irrigation System	0.044489	0.072022	0.058687	0.044532	0.073359	0.044038

Gambar 3.7. Hasil Matriks Ternormalisasi Bobot

Setelah menormalisasi matriks keputusan, proses selanjutnya adalah mengalikan setiap kolom dari matriks tersebut dengan bobot-bobot yang telah dioptimalkan menggunakan algoritma *Gradient Descent*. Proses ini menghasilkan matriks baru yang disebut matriks ternormalisasi bobot. Matriks ini memperlihatkan evaluasi sistem irigasi berdasarkan beberapa kriteria, seperti efisiensi air, ketersediaan air, jenis tanah, jenis tanaman, dan biaya implementasi. Gambar diatas merupakan hasil matriks ternormalisasi bobot yang dapat dijelaskan bahwa, *Surface Irrigation System* memiliki skor tinggi pada efisiensi air (0,073359) dan ketersediaan air (0,073396), menunjukkan kesesuaiannya untuk daerah dengan ketersediaan air yang baik, meskipun biaya implementasinya rendah (0,028809). *Sprinkler Irrigation System* unggul dalam kriteria jenis tanah (0,074220) dan jenis tanaman (0,058687), sehingga cocok untuk berbagai kondisi tanah dan tanaman, meskipun memiliki biaya implementasi yang relatif tinggi (0,043213). *Drip Irrigation System* dan *Subsurface Irrigation System* menonjol dalam efisiensi air (0,044015 dan 0,058687), namun memiliki biaya implementasi yang tinggi (0,072022), sehingga lebih cocok untuk daerah yang memerlukan penghematan air. *Center Pivot Irrigation* dan *Lateral Move Irrigation System* menunjukkan skor yang seimbang dalam efisiensi air (0,073359) dan jenis tanaman (0,058687), meskipun membutuhkan biaya implementasi yang cukup besar (0,057617). Sementara itu, *Rain Irrigation System* memiliki skor terendah pada efisiensi air (0,029343) dan ketersediaan air (0,029395), menunjukkan keterbatasannya untuk daerah dengan kebutuhan irigasi yang intensif.

3.2.8 Matriks *Concordance*

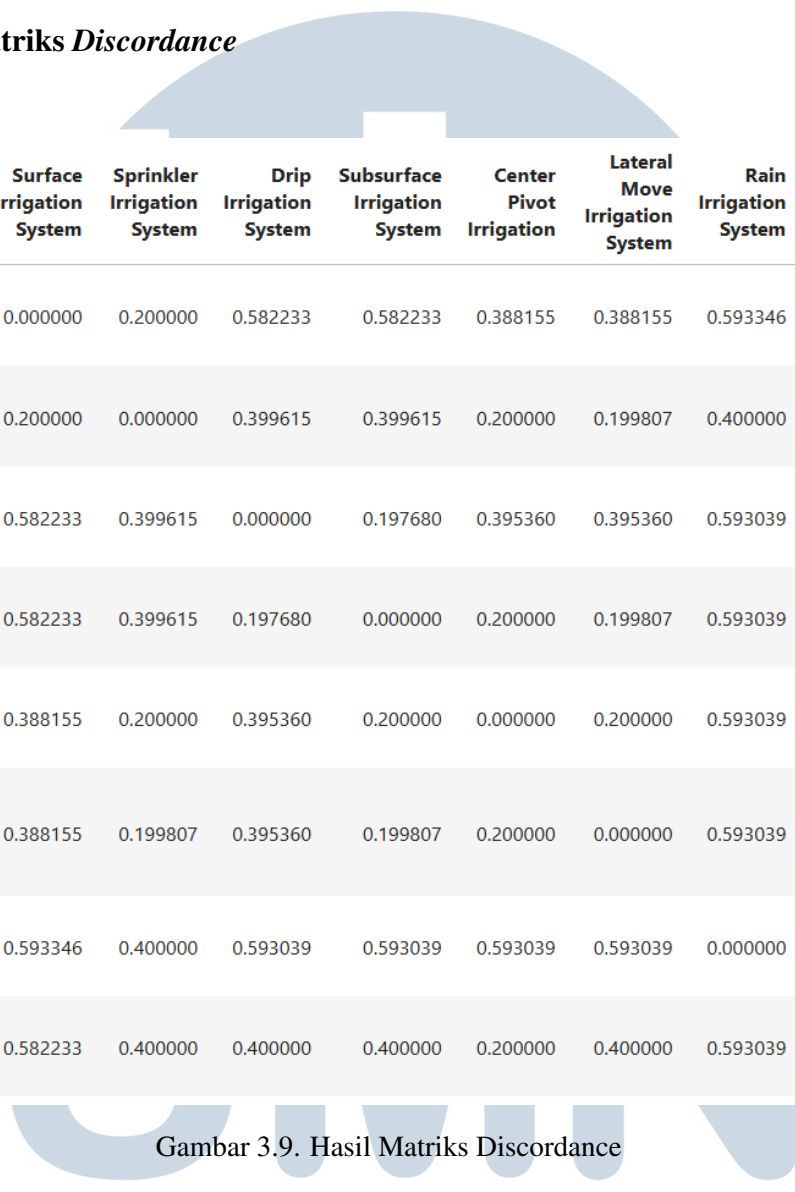
	Surface Irrigation System	Sprinkler Irrigation System	Drip Irrigation System	Subsurface Irrigation System	Center Pivot Irrigation	Lateral Move Irrigation System	Rain Irrigation System	Fog Irrigation System
Surface Irrigation System	0.0000	0.3208	0.4866	0.4866	0.6684	0.4866	0.8341	0.6684
Sprinkler Irrigation System	0.6791	0.0000	0.6684	0.6684	0.6684	0.6684	0.9999	0.6684
Drip Irrigation System	0.5133	0.5133	0.0000	0.8288	0.5133	0.5133	0.8341	0.8288
Subsurface Irrigation System	0.5133	0.6844	0.9999	0.0000	0.5133	0.5133	0.8341	0.9999
Center Pivot Irrigation	0.8502	0.6523	0.4866	0.4866	0.0000	0.8181	0.8341	0.6684
Lateral Move Irrigation System	0.8502	0.8341	0.6684	0.6684	0.9999	0.0000	0.8341	0.6684
Rain Irrigation System	0.3262	0.1658	0.1658	0.1658	0.1658	0.1658	0.0000	0.3476
Fog Irrigation System	0.3315	0.5026	0.8181	0.8181	0.3315	0.3315	0.8341	0.0000

Gambar 3.8. Hasil Matriks *Concordance*

Gambar diatas merupakan hasil dari matriks *concordance*. Di dalam matriks *corcondance* ini, nilai-nilai yang ditampilkan menunjukkan sejauh mana suatu alternatif sistem irigasi lebih baik dibandingkan alternatif lainnya berdasarkan sejumlah kriteria yang memiliki bobot tertentu. Nilai *concordance* dihitung dengan menjumlahkan bobot dari kriteria di mana suatu alternatif memiliki skor yang lebih baik dibandingkan alternatif lainnya. Misalnya, *Sprinkler Irrigation System* memiliki nilai *concordance* sebesar 0,6791 ketika dibandingkan dengan *Surface Irrigation System*, yang berarti dalam 67,91 persen kriteria, sistem ini lebih baik. Nilai tinggi pada matriks ini menunjukkan dominasi alternatif tertentu terhadap alternatif lainnya, seperti *Lateral Move Irrigation System* yang memiliki nilai *concordance* tinggi terhadap banyak sistem, menunjukkan performa yang unggul. Sebaliknya, *Rain Irrigation System* memiliki nilai *concordance* rendah dalam banyak perbandingan,

yang menunjukkan bahwa sistem ini kalah dalam banyak kriteria.

3.2.9 Matriks *Discordance*



	Surface Irrigation System	Sprinkler Irrigation System	Drip Irrigation System	Subsurface Irrigation System	Center Pivot Irrigation	Lateral Move Irrigation System	Rain Irrigation System	Fog Irrigation System
Surface Irrigation System	0.000000	0.200000	0.582233	0.582233	0.388155	0.388155	0.593346	0.582233
Sprinkler Irrigation System	0.200000	0.000000	0.399615	0.399615	0.200000	0.199807	0.400000	0.400000
Drip Irrigation System	0.582233	0.399615	0.000000	0.197680	0.395360	0.395360	0.593039	0.400000
Subsurface Irrigation System	0.582233	0.399615	0.197680	0.000000	0.200000	0.199807	0.593039	0.400000
Center Pivot Irrigation	0.388155	0.200000	0.395360	0.200000	0.000000	0.200000	0.593039	0.200000
Lateral Move Irrigation System	0.388155	0.199807	0.395360	0.199807	0.200000	0.000000	0.593039	0.400000
Rain Irrigation System	0.593346	0.400000	0.593039	0.593039	0.593039	0.593039	0.000000	0.593039
Fog Irrigation System	0.582233	0.400000	0.400000	0.400000	0.200000	0.400000	0.593039	0.000000

Gambar 3.9. Hasil Matriks *Discordance*

Setelah melakukan proses matriks *concordance*, selanjutnya adalah melakukan perhitungan matriks *discordance*. Gambar di atas merupakan hasil dari perhitungan matriks *discordance*. Matriks *discordance* menunjukkan sejauh mana suatu alternatif sistem irigasi berbeda atau memiliki ketidaksesuaian dibandingkan alternatif lainnya berdasarkan sejumlah kriteria tertentu. Nilai *discordance* dihitung dengan membandingkan selisih skor dari kriteria di mana suatu alternatif memiliki performa lebih buruk dibandingkan alternatif lainnya, kemudian dibagi dengan selisih maksimum pada semua kriteria.

Sebagai contoh, *Surface Irrigation System* memiliki nilai *discordance* sebesar 0,582233 ketika dibandingkan dengan *Drip Irrigation System*, yang menunjukkan tingkat ketidaksesuaian yang signifikan di beberapa kriteria antara kedua alternatif tersebut. Nilai *discordance* yang tinggi, seperti antara *Rain Irrigation System* dan *Center Pivot Irrigation* sebesar 0,593039, menunjukkan perbedaan yang besar dalam banyak kriteria, yang berarti *Rain Irrigation System* kurang sesuai dibandingkan dengan *Center Pivot Irrigation* dalam konteks tersebut.

Sebaliknya, nilai *discordance* yang rendah, seperti antara *Sprinkler Irrigation System* dan *Lateral Move Irrigation System* sebesar 0,199807, menunjukkan bahwa kedua alternatif ini memiliki tingkat perbedaan yang kecil dalam performa berdasarkan kriteria tertentu. Matriks *discordance* ini membantu dalam menganalisis kelemahan relatif dari setiap alternatif dibandingkan dengan yang lain, sehingga memberikan gambaran tentang tingkat ketidaksesuaian antar alternatif dalam pemilihan sistem irigasi.



3.2.10 Matriks Dominan *Concordance* dan *Discordance*

```
Concordance Threshold:
0.5369625

Discordance Threshold:
0.35214980479727365

Matriks Dominan Concordance:
[[0 0 0 0 1 0 1 1]
 [1 0 1 1 1 1 1 1]
 [0 0 0 1 0 0 1 1]
 [0 1 1 0 0 0 1 1]
 [1 1 0 0 0 1 1 1]
 [1 1 1 1 1 0 1 1]
 [0 0 0 0 0 0 0 0]
 [0 0 1 1 0 0 1 0]]

Matriks Dominan Discordance:
[[1 1 0 0 0 0 0 0]
 [1 1 0 0 1 1 0 0]
 [0 0 1 1 0 0 0 0]
 [0 0 1 1 1 1 0 0]
 [0 1 0 1 1 1 0 1]
 [0 1 0 1 1 1 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 1]]
```

Gambar 3.10. Hasil Matriks Dominan *Concordance* dan *Discordance*

Matriks diatas menunjukkan adanya pembentukan matriks dominan *concordance* dan *discordance* berdasarkan nilai ambang batas (*threshold*). Ambang batas *concordance* adalah 0.5369625, sedangkan ambang batas *discordance* adalah 0.35214980479727365. Matriks dominan *concordance* berisi nilai biner (0 dan 1), di mana "1" menunjukkan bahwa nilai *concordance* antara dua alternatif melebihi ambang batas, sedangkan "0" berarti sebaliknya. Begitu pula, matriks dominan *discordance* menampilkan nilai biner yang menunjukkan apakah nilai *discordance* antara dua alternatif melebihi atau tidak melewati ambang batas. Kombinasi kedua matriks ini digunakan untuk mengidentifikasi hubungan preferensi kuat di antara alternatif-alternatif yang dibandingkan.

3.2.11 Matriks Dominan Agregat

	Surface Irrigation System	Sprinkler Irrigation System	Drip Irrigation System	Subsurface Irrigation System	Center Pivot Irrigation	Lateral Move Irrigation System	Rain Irrigation System	Fog Irrigation System
Surface Irrigation System	0	0	0	0	0	0	0	0
Sprinkler Irrigation System	1	0	0	0	1	1	0	0
Drip Irrigation System	0	0	0	1	0	0	0	0
Subsurface Irrigation System	0	0	1	0	0	0	0	0
Center Pivot Irrigation	0	1	0	0	0	1	0	1
Lateral Move Irrigation System	0	1	0	1	1	0	0	0
Rain Irrigation System	0	0	0	0	0	0	0	0
Fog Irrigation System	0	0	0	0	0	0	0	0

Gambar 3.11. Hasil Matriks Dominan Agregat

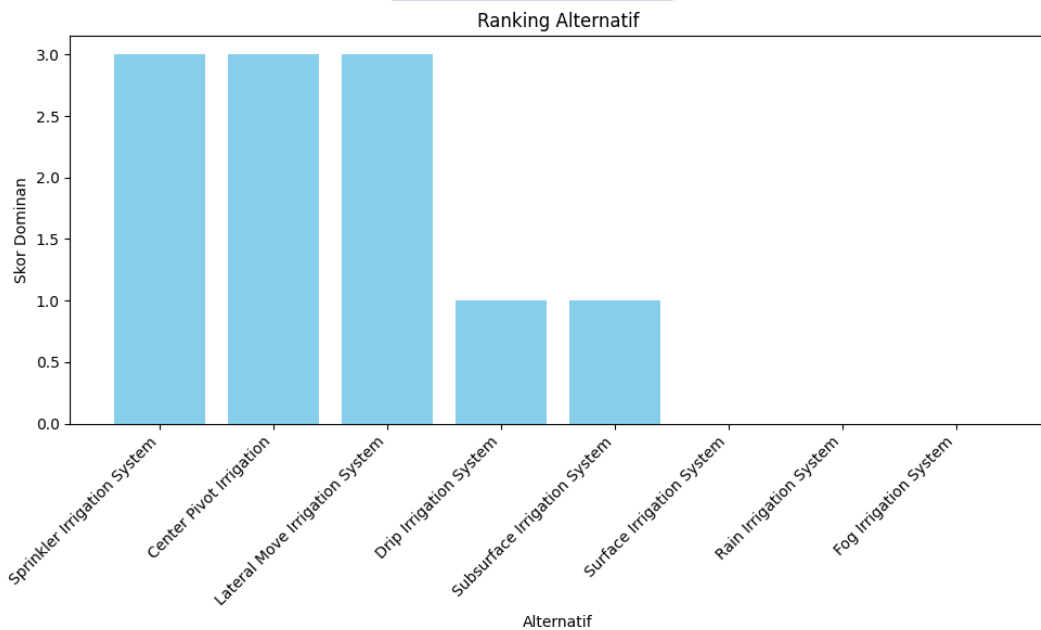
Gambar di atas merupakan hasil dari matriks dominan agregat. Dalam matriks ini, nilai 1 menunjukkan bahwa suatu alternatif sistem irigasi mendominasi alternatif lainnya berdasarkan kriteria tertentu, sedangkan nilai 0 menunjukkan bahwa tidak ada dominasi antara kedua alternatif. Sebagai contoh, *Sprinkler Irrigation System* mendominasi *Surface Irrigation System* dan *Rain Irrigation System*, terlihat dari nilai 1 pada baris *Sprinkler Irrigation System* di kolom tersebut. Selain itu, *Center Pivot Irrigation* mendominasi *Drip Irrigation System* dan *Lateral Move Irrigation System*, sebagaimana terlihat pada barisnya. Sebaliknya, *Fog Irrigation System* tidak mendominasi alternatif lainnya, karena memiliki nilai 0 pada seluruh kolom. Hal ini menunjukkan bahwa sistem ini tidak memiliki keunggulan dominan dibandingkan alternatif lainnya berdasarkan kriteria yang dianalisis.

3.2.12 Hasil Perankingan Alternatif Menggunakan *ELECTRE*

Final Ranking:

	Alternatif	Skor Dominan
0	Sprinkler Irrigation System	3
1	Center Pivot Irrigation	3
2	Lateral Move Irrigation System	3
3	Drip Irrigation System	1
4	Subsurface Irrigation System	1
5	Surface Irrigation System	0
6	Rain Irrigation System	0
7	Fog Irrigation System	0

Gambar 3.12. Hasil Ranking Alternatif



Gambar 3.13. Visual Ranking Alternatif

Berdasarkan hasil analisis menggunakan metode *ELECTRE*, ditemukan bahwa Sistem Irigasi Terbaik yang sangat direkomendasikan adalah *Sprinkler Irrigation System*, *Center Pivot Irrigation*, dan *Lateral Move Irrigation System*, de-

ngan skor dominan tertinggi sebesar 3. Hal ini menunjukkan bahwa ketiga sistem tersebut unggul di hampir semua kriteria yang dievaluasi dan memberikan performa optimal untuk memenuhi kebutuhan irigasi secara keseluruhan.

Sementara itu, *Drip Irrigation System* dan *Subsurface Irrigation System* memiliki skor dominan sebesar 1, yang mengindikasikan bahwa meskipun sistem ini memiliki keunggulan pada beberapa kriteria, kinerja keseluruhannya masih kurang optimal dibandingkan alternatif lain. Kedua sistem ini dapat dipertimbangkan apabila terdapat kebutuhan atau kondisi tertentu yang sesuai dengan kelebihan mereka.

Di sisi lain, sistem irigasi seperti *Surface Irrigation System*, *Rain Irrigation System*, dan *Fog Irrigation System* mendapatkan skor dominan 0, yang menunjukkan bahwa mereka tidak memenuhi kriteria evaluasi dengan baik dan tidak direkomendasikan sebagai solusi alternatif. Oleh karena itu, fokus implementasi diberikan kepada sistem irigasi dengan skor dominan tertinggi untuk memastikan efisiensi dan efektivitas penggunaan sistem irigasi.

3.3 Evaluasi

Penelitian ini berhasil menunjukkan bagaimana penggabungan algoritma *Gradient Descent* dan metode MCDMELECTRE dapat meningkatkan efisiensi distribusi air pada sistem irigasi modern. *Gradient Descent* digunakan untuk mengoptimalkan bobot kriteria berdasarkan data sintesis, sementara *ELECTRE* memberikan rangkaian peringkat alternatif untuk pola irigasi. Hasil menunjukkan bahwa metode yang digunakan tidak hanya memberikan solusi optimal secara matematis tetapi juga memberikan panduan praktis bagi pengambilan keputusan dalam memilih strategi irigasi yang efisien dan sesuai kebutuhan.

3.3.1 Evaluasi Algoritma *Gradient Descent*

Algoritma *Gradient Descent* menunjukkan kinerja yang sangat baik dalam meminimalkan fungsi tujuan hingga mencapai konvergensi. Optimasi menghasilkan bobot untuk enam kriteria utama, dengan kriteria Jenis Tanah memiliki pengaruh tertinggi (bobot 0,1818), diikuti oleh Efisiensi Air (bobot 0,1711). Grafik proses optimasi menunjukkan bahwa nilai fungsi tujuan menurun secara konsisten hingga mencapai stabilitas (titik konvergensi). Penurunan nilai fungsi tujuan ini menunjukkan bahwa model secara bertahap mendekati solusi optimal dengan

memperbaiki parameter yang meminimalkan loss function.

Ketika nilai fungsi tujuan mulai datar setelah beberapa iterasi, hal ini menandakan bahwa proses optimasi telah mencapai titik di mana perubahan nilai fungsi tujuan sangat kecil atau tidak signifikan lagi, yang disebut sebagai konvergensi. Hal ini penting karena memastikan bahwa model tidak *overfitting* atau melakukan iterasi berlebih yang tidak memberikan keuntungan tambahan. Secara keseluruhan, *Gradient Descent* tidak hanya efektif dalam mengoptimalkan bobot kriteria tetapi juga efisien dalam menyelesaikan proses optimasi dengan jumlah iterasi yang wajar.

3.3.2 Evaluasi MCDM ELECTRE

Metode *ELECTRE* berhasil memberikan peringkat alternatif berdasarkan matriks dominan agregat. Dengan menggunakan bobot yang telah dioptimalkan menggunakan *Gradient Descent*, *ELECTRE* dapat secara efektif menyusun peringkat alternatif berdasarkan kriteria yang relevan. Bobot yang dihasilkan oleh *Gradient Descent*, seperti bobot tertinggi pada kriteria Jenis Tanah (0,1818) dan Efisiensi Air (0,1711), memberikan pengaruh signifikan pada hasil perankingan, memastikan bahwa alternatif yang lebih unggul dalam kriteria-kriteria ini mendapatkan prioritas yang lebih tinggi.

Hasil perankingan menunjukkan bahwa *Sprinkler Irrigation System*, *Center Pivot Irrigation*, dan *Lateral Move Irrigation System* menempati posisi tertinggi dengan skor dominan 3, mencerminkan keunggulan mereka dalam memenuhi kebutuhan irigasi berdasarkan kriteria yang telah ditetapkan. Sebaliknya, *Rain Irrigation System* dan *Fog Irrigation System* memiliki skor dominan 0, yang menunjukkan bahwa sistem ini tidak dapat bersaing dengan alternatif lain dalam hal efisiensi atau fleksibilitas.

Validasi hasil menunjukkan bahwa *ELECTRE* mampu memberikan evaluasi yang akurat, menggabungkan pengaruh bobot dari *Gradient Descent* dengan analisis komparatif yang detail. Hal ini menghasilkan keputusan berbasis data yang transparan dan dapat dipertanggungjawabkan, menjadikan *ELECTRE* metode yang handal untuk membantu pengambilan keputusan dalam memilih strategi irigasi yang optimal.

3.3.3 Hubungan antara Algoritma Gradient Descent dan MCDM ELECTRE

Hubungan antara *Gradient Descent* dan *ELECTRE* dalam penelitian ini bersifat sinergis. *Gradient Descent* digunakan untuk mengoptimalkan bobot kriteria yang menjadi input penting bagi *ELECTRE*. Bobot yang telah dioptimalkan memberikan dasar yang lebih kuat untuk *ELECTRE* untuk memastikan bahwa proses perankingan dalam *ELECTRE* mencerminkan tingkat kepentingan setiap kriteria secara proporsional dan objektif. Sinergi ini menghasilkan proses pengambilan keputusan dan perankingan yang lebih akurat dan efisien, karena *Gradient Descent* menyediakan dasar matematis yang kuat untuk menetapkan prioritas, sementara *ELECTRE* menyusun peringkat berdasarkan analisis komparatif antar-alternatif. Dengan demikian, hubungan ini memungkinkan pendekatan hybrid yang mampu menghasilkan solusi optimal untuk sistem irigasi modern.

3.4 Kesimpulan dan Saran

Penelitian ini berhasil menunjukkan bagaimana penggabungan algoritma *Gradient Descent* dengan metode *ELECTRE* dapat meningkatkan efisiensi distribusi air pada sistem irigasi modern. Algoritma *Gradient Descent* digunakan untuk mengoptimalkan bobot kriteria, memberikan dasar yang kuat untuk proses evaluasi dalam *ELECTRE*. Bobot yang dihasilkan memastikan bahwa setiap kriteria dievaluasi secara proporsional berdasarkan kepentingannya. Metode *ELECTRE* kemudian berhasil menyusun peringkat alternatif irigasi secara akurat, dengan *Sprinkler Irrigation System*, *Center Pivot Irrigation*, dan *Lateral Move Irrigation System* menempati peringkat tertinggi. Hal ini menunjukkan bahwa kombinasi kedua metode ini dapat memberikan solusi yang efisien dan akurat untuk memilih sistem irigasi yang optimal. Meskipun penelitian ini berbasis data sintesis, hasilnya memberikan panduan yang jelas untuk meningkatkan efisiensi irigasi dan hasil panen.

Penelitian ini menggunakan data sintetis. Oleh karena itu untuk penelitian selanjutnya, disarankan menggunakan data lapangan guna memvalidasi efektivitas solusi di skala operasional sebenarnya. Selain itu, integrasi algoritma tambahan seperti *Genetic Algorithm* atau *Machine Learning* dapat dieksplorasi untuk meningkatkan akurasi prediksi kebutuhan irigasi. Penelitian selanjutnya juga disarankan untuk menggunakan algoritma dan Metode Pengambilan Keputusan Multikriteria (MCDM) lainnya, untuk membandingkannya dengan penelitian ini guna melihat apakah hasil yang diperoleh lebih baik atau tidak.

BAB 4

DOKUMENTASI *PYTHON SCRIPT*

4.1 Import Library

```
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
```

Kode 4.1: Import Library

Kode diatas merupakan kode yang digunakan untuk mengimpor tiga *library* utama dalam Python yang sering dipakai dalam analisis data dan visualisasi. *Library NumPy* (import numpy as np) digunakan untuk melakukan perhitungan matematis dan manipulasi array atau matriks dengan efisien. Selanjutnya, Pandas (import pandas as pd) adalah *library* yang berfungsi untuk membaca, mengolah, dan menganalisis data dalam format tabular seperti CSV atau Excel, sehingga mempermudah manipulasi data seperti filtering, pengelompokan, dan pembuatan tabel. Terakhir, Matplotlib melalui submodul pyplot (import matplotlib.pyplot as plt) dipakai untuk membuat berbagai jenis grafik seperti garis, batang, atau scatter plot untuk memvisualisasikan data.

4.2 Inisialisasi Dataset

4.2.1 Membuat Matriks Dari Dataset dan DataFrame

```
1 # Data Tabel Kriteria (Kolom)
2 data = {
3     "Topografi": [4, 5, 3, 3, 4, 4, 5, 3],
4     "Biaya Implementasi": [2, 3, 5, 5, 4, 4, 2, 5],
5     "Efisiensi Air": [5, 4, 3, 4, 5, 5, 2, 4],
6     "Jenis Tanah": [4, 5, 5, 5, 4, 5, 3, 3],
7     "Jenis Tanaman": [3, 4, 5, 5, 4, 4, 2, 5],
8     "Ketersediaan Air": [5, 4, 3, 3, 4, 4, 2, 3]
9 }
10
11 # Data Tabel Alternatif (Baris)
12 alternatives = [
13     "Surface Irrigation System",
14     "Sprinkler Irrigation System",
```



```

15     "Drip Irrigation System",
16     "Subsurface Irrigation System",
17     "Center Pivot Irrigation",
18     "Lateral Move Irrigation System",
19     "Rain Irrigation System",
20     "Fog Irrigation System"
21 ]
22
23 # Membuat dataframe
24 df = pd.DataFrame(data, index=alternatives)

```

Kode 4.2: Membuat Matriks Dari Dataset dan DataFrame

Kode diatas merupakan sebuah proses awal untuk inisialisasi dataset dimana kode diatas membuat matriks dari dataset dimana Data kriteria seperti "Topografi," "Biaya Implementasi," "Efisiensi Air," "Jenis Tanah," "Jenis Tanaman," dan "Ketersediaan Air" diberikan dalam bentuk nilai numerik yang mewakili penilaian masing-masing alternatif pada kriteria tersebut. Alternatif, seperti "*Surface Irrigation System*" dan "*Sprinkler Irrigation System*", dimasukkan sebagai daftar untuk digunakan sebagai indeks. Data tersebut kemudian diubah menjadi sebuah DataFrame menggunakan *library* Pandas, dengan kolom merepresentasikan kriteria dan baris merepresentasikan alternatif.

4.2.2 Visualisasi Dataset

```

1 df.plot(kind="bar", figsize=(12, 6),
2         title="Initial Dataset for Irrigation Strategies")
3 plt.xlabel("Irrigation Strategies")
4 plt.ylabel("Criteria Scores")
5 plt.xticks(rotation=45, ha="right")
6 plt.tight_layout()
7 plt.show()

```

Kode 4.3: Visualisasi Dataset

Kode diatas digunakan untuk memvisualisasikan dataset dalam bentuk grafik batang (bar chart) menggunakan library Pandas dan Matplotlib. Visualisasi ini memberikan gambaran awal mengenai skor setiap kriteria untuk berbagai strategi irigasi.

4.3 Proses Algoritma Gradient Descent

4.3.1 Inisialisasi Bobot Awal

```
1 initial_weights = np.array([0.2, 0.3, 0.2, 0.1, 0.1, 0.1])
2 print("Bobot : ")
3 print(initial_weights)
```

Kode 4.4: Inisialisasi Bobot Awal

Kode diatas digunakan untuk mendefinisikan bobot awal yang akan diterapkan pada setiap kriteria dalam analisis pengambilan keputusan, lalu mencetak bobot tersebut ke layar.

4.3.2 Normalisasi Bobot

```
1 initial_weights /= initial_weights.sum()
2 print("Bobot normalisasi:", initial_weights)
```

Kode 4.5: Normalisasi Bobot

Kode diatas digunakan untuk melakukan normalisasi bobot, memastikan bahwa total bobot berjumlah 1 dengan membagi setiap elemen bobot dalam array initial weights dengan jumlah total dari semua elemen di array tersebut.

4.3.3 Membuat Fungsi Tujuan dan Penalty L2 (*Ridge Regularization*)

```
1 def objective_function(weights, df, lambda_penalty=0.1):
2     # Penalty L2
3     penalty = lambda_penalty * np.sum(weights**2)
4     # Menghitung nilai fungsi tujuan
5     return -(weights @ df.mean(axis=0).values) + penalty
```

Kode 4.6: Membuat Fungsi Tujuan dan Penalty L2 (*Ridge Regularization*)

Kode diatas mendefinisikan fungsi tujuan dalam optimasi yang menghitung nilai berdasarkan bobot dan rata-rata nilai kriteria. Fungsi ini menambahkan penalti L2 untuk mencegah bobot terlalu besar, dengan tujuan untuk meminimalkan hasil fungsi tersebut.

4.3.4 Menghitung Gradien

```

1 def gradient(weights, df, lambda_penalty=0.1):
2     # Gradien dari fungsi tujuan utama
3     grad = -df.mean(axis=0).values
4     # Gradien dari penalti L2
5     penalty_grad = 2 * lambda_penalty * weights
6     return grad + penalty_grad

```

Kode 4.7: Menghitung Gradien

Kode diatas menghitung gradien untuk proses optimasi, termasuk penalti regularisasi L2, yang membantu mengarahkan pembaruan bobot dalam algoritma *Gradient Descent*.

4.3.5 Implementasi Algoritma Gradient Descent Untuk Optimasi Bobot

```

1 def gradient_descent(weights, df, learning_rate=0.01, max_iter
2     =1000, tolerance=1e-6, lambda_penalty=0.1):
3
4     history = []
5     objective_values = []
6
7     for i in range(max_iter):
8         obj_value = objective_function(weights, df, lambda_penalty
9             )
10
11         grad = gradient(weights, df, lambda_penalty)
12
13         weights_next = weights - learning_rate * grad
14         weights_next = np.clip(weights_next, 0, 1)
15         weights_next /= weights_next.sum()
16
17         history.append(weights_next)
18         objective_values.append(obj_value)
19
20         if np.linalg.norm(grad) < tolerance:
21             print(f"Converged after {i + 1} iterations")
22             return weights_next, history, objective_values
23
24         weights = weights_next
25
26     print(f"Mencapai iterasi maksimum ({max_iter})")
27     return weights, history, objective_values

```

```

27 optimal_weights, optimization_history, objective_values =
    gradient_descent(
28     initial_weights, df, learning_rate=0.01, max_iter=1000,
        tolerance=1e-6, lambda_penalty=0.1
29 )
30
31 print("\nOptimal Weights:\n")
32 for i, w in enumerate(optimal_weights):
33     print(f"{df.columns[i]}: {w:.4f}")

```

Kode 4.8: Implementasi Algoritma *Gradient Descent* Untuk Optimasi Bobot

Kode ini mengimplementasikan algoritma *Gradient Descent* untuk mengoptimalkan bobot, mencatat perubahan selama iterasi, dan menghentikan proses ketika konvergensi tercapai atau iterasi maksimum dilewati. Keluarannya adalah bobot optimal, riwayat optimasi, dan nilai fungsi tujuan.

4.3.6 Visualisasi Perubahan Nilai Fungsi Tujuan

```

1 plt.figure(figsize=(10, 6))
2 plt.plot(range(len(objective_values)), objective_values, color='r',
    , label="Nilai Fungsi Tujuan")
3 plt.title("Perubahan Nilai Fungsi Tujuan selama Optimasi")
4 plt.xlabel("Iterasi")
5 plt.ylabel("Nilai Fungsi Tujuan")
6 plt.legend()
7 plt.grid()
8 plt.show()

```

Kode 4.9: Visualisasi Perubahan Nilai Fungsi Tujuan

Kode ini menghasilkan grafik yang menunjukkan bagaimana nilai fungsi tujuan berubah selama iterasi optimasi, membantu menganalisis proses konvergensi optimasi.

4.3.7 Visualisasi Proses Optimasi Bobot Dengan Gradient Descent

```

1 plt.figure(figsize=(10, 6))
2 optimization_history = np.array(optimization_history)
3 for i in range(optimization_history.shape[1]):
4     plt.plot(optimization_history[:, i], label=f"{df.columns[i]}")
5
6 plt.title("Proses Optimasi Bobot dengan Gradient Descent")

```

```

7 plt.xlabel("Iterasi")
8 plt.ylabel("Nilai Bobot")
9 plt.legend()
10 plt.grid()
11 plt.show()

```

Kode 4.10: Visualisasi Proses Optimasi Bobot Dengan *Gradient Descent*

Kode ini menghasilkan grafik yang menunjukkan perubahan nilai bobot selama proses iterasi optimasi dengan *Gradient Descent*, memudahkan analisis tren konvergensi bobot.

4.4 Proses Metode MCDM Menggunakan ELECTRE

4.4.1 Menampilkan Matriks Keputusan

```

1 decision_matrix = df.to_numpy()
2 print("Matriks Keputusan")
3 df

```

Kode 4.11: Menampilkan Matriks Keputusan

4.4.2 Normalisasi Matriks Keputusan

```

1 decision_matrix = df.to_numpy()
2 norm_matrix = decision_matrix / np.sqrt((decision_matrix**2).sum(
    axis=0))
3 norm_df = pd.DataFrame(norm_matrix, index=alternatives, columns=
    data.keys())
4 print("Tahap 1: Normalisasi Matriks Keputusan")
5 norm_df

```

Kode 4.12: Normalisasi Matriks Keputusan

Kode ini menghasilkan Matriks Keputusan Ternormalisasi yang digunakan sebagai langkah awal dalam metode pengambilan keputusan berbasis multi-kriteria, memastikan data pada setiap kriteria memiliki skala yang sama.

4.4.3 Matriks Ternormalisasi Bobot

```

1 # Memberikan Hasil Optimasi Bobot Dari Gradient Descent
2 optimal_weights = np.array([0.1658, 0.1604, 0.1711, 0.1818,
    0.1711, 0.1497])

```

```

3
4 # Melakukan Normalisasi Bobot
5 weighted_matrix = norm_matrix * optimal_weights
6 weighted_df = pd.DataFrame(weighted_matrix , index=alternatives ,
    columns=data.keys())
7 print("Tahap 2: Matriks Ternormalisasi Terbobot")
8 weighted_df

```

Kode 4.13: Matriks Ternormalisasi Bobot

Kode ini menghitung matriks bobot ternormalisasi dengan mengalikan matriks data yang telah dinormalisasi dengan bobot optimal. Matriks ini digunakan untuk analisis perbandingan alternatif berdasarkan bobot yang telah diperhitungkan secara optimal.

4.4.4 Menghitung Matriks *Concordance*

```

1 concordance_matrix = np.zeros((len(alternatives), len(alternatives)
    ))
2
3 for i in range(len(alternatives)):
4     for j in range(len(alternatives)):
5         if i != j:
6             concordance_matrix[i, j] = sum(
7                 optimal_weights[k] for k in range(len(
8                     optimal_weights))
9                 if weighted_matrix[i, k] >= weighted_matrix[j, k]
10            )
11 concordance_df = pd.DataFrame(concordance_matrix , index=
    alternatives , columns=alternatives)
12 print("Tahap 3: Matriks Concordance")
13 concordance_df

```

Kode 4.14: Menghitung Matriks *Concordance*

Kode ini menghitung matriks *concordance*, yang menunjukkan tingkat kesesuaian antar alternatif berdasarkan bobot kriteria. Matriks ini digunakan dalam metode pengambilan keputusan untuk menilai keunggulan relatif setiap alternatif.

4.4.5 Menghitung Matriks *Discordance*

```

1 discordance_matrix = np.zeros((len(alternatives), len(alternatives)
2 ))
3 for i in range(len(alternatives)):
4     for j in range(len(alternatives)):
5         if i != j:
6             discordance_matrix[i, j] = max(
7                 abs(weighted_matrix[i, k] - weighted_matrix[j, k])
8                 for k in range(len(optimal_weights))
9             ) / max(abs(weighted_matrix).flatten())
10 discordance_df = pd.DataFrame(discordance_matrix, index=
11     alternatives, columns=alternatives)
12 print("Tahap 4: Matriks Discordance")
13 discordance_df

```

Kode 4.15: Menghitung Matriks *Discordance*

Kode ini menghitung matriks *discordance*, yang menunjukkan ketidaksesuaian antara alternatif berdasarkan bobot kriteria. Matriks ini digunakan dalam analisis perbandingan alternatif untuk membantu pengambilan keputusan.

4.4.6 Menghitung Matriks Dominan *Concordance* dan *Discordance*

```

1 print("Tahap 5: Matriks Dominan Concordance dan Discordance\n")
2
3 # Threshold matriks Concordance
4 concordance_threshold = concordance_matrix.mean()
5 print("Concordance Threshold:")
6 print(concordance_threshold)
7
8 # Threshold matriks Discordance
9 discordance_threshold = discordance_matrix.mean()
10 print("\nDiscordance Threshold:")
11 print(discordance_threshold)
12
13 # Matriks Dominan Concordance
14 dominant_concordance = (concordance_matrix >=
15     concordance_threshold).astype(int)
16 print("\nMatriks Dominan Concordance:")
17 print(dominant_concordance)
18
19 # Matriks Dominan Discordance

```

```

19 dominant_discordance = (discordance_matrix <=
    discordance_threshold).astype(int)
20 print("\nMatriks Dominan Discordance:")
21 print(dominant_discordance)

```

Kode 4.16: Menghitung Matriks Dominan *Concordance* dan *Discordance*

Kode ini menentukan matriks dominan berdasarkan ambang batas rata-rata untuk *concordance* dan *discordance*. Matriks dominan digunakan untuk analisis dominasi antar alternatif.

4.4.7 Menentukan Matriks Dominan Agregat

```

1 aggregate_dominance_matrix = dominant_concordance *
    dominant_discordance
2
3 aggregate_dominance_df = pd.DataFrame(
4     aggregate_dominance_matrix,
5     index=alternatives,
6     columns=alternatives
7 )
8
9 print("Tahap 6: Matriks Dominan Agregat")
10 aggregate_dominance_df
11 print(aggregate_dominance_matrix)

```

Kode 4.17: Menentukan Matriks Dominan Agregat

Kode di atas digunakan untuk menentukan Matriks Dominan Agregat dengan mengalikan matriks *dominant_concordance* dan *dominant_discordance*. Hasilnya disimpan dalam bentuk *DataFrame* untuk mempermudah visualisasi, dengan indeks dan kolom sesuai nama alternatif. Hasil matriks kemudian ditampilkan untuk analisis lebih lanjut.

4.4.8 Melakukan Perankingan Alternatif

```

1 dominance_scores = aggregate_dominance_matrix.sum(axis=1)
2
3 final_ranking = sorted(
4     zip(alternatives, dominance_scores), key=lambda x: x[1],
5     reverse=True
6 )
7 final_ranking_df = pd.DataFrame(
8     final_ranking, columns=["Alternatif", "Skor Dominan"]
9 )

```



```

10 print("Final Ranking:\n")
11 print(final_ranking_df)

```

Kode 4.18: Melakukan Perankingan Alternatif

Kode ini digunakan untuk menghitung skor dominasi, mengurutkan alternatif berdasarkan skor tersebut, dan menyajikan hasil perankingan dalam bentuk tabel. Proses ini membantu menentukan alternatif terbaik berdasarkan skor dominasi tertinggi.

4.4.9 Visualisasi Ranking Alternatif

```

1 plt.figure(figsize=(10, 6))
2 plt.title("Ranking Alternatif")
3 plt.bar(final_ranking_df["Alternatif"], final_ranking_df["Skor Dominan"], color="skyblue")
4 plt.xticks(rotation=45, ha="right")
5 plt.xlabel("Alternatif")
6 plt.ylabel("Skor Dominan")
7 plt.tight_layout()
8 plt.show()

```

Kode 4.19: Visualisasi Ranking Alternatif

Kode di atas digunakan untuk membuat diagram batang yang menampilkan Skor Dominan untuk setiap Alternatif. Grafik diberi judul, label pada sumbu x dan y, serta warna batang biru muda. Rotasi label sumbu x diatur agar mudah dibaca, dan tata letak grafik disesuaikan secara otomatis agar elemen tidak tumpang tindih. Kode ini membantu memvisualisasikan data secara jelas dan rapi.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

DAFTAR PUSTAKA

- [1] M. Siswanto, M. Ali, M. A. Haikal *et al.*, “Optimasi pid kontroller pada sistem pengaturan irigasi menggunakan metode bat algorithm (ba),” *Jurnal JEETech*, vol. 3, no. 2, pp. 78–83, 2022.
- [2] A. A. Wahyudi, N. N. Anwar, and E. E. Edijanto, “Studi optimasi pola tanam pada daerah irigasi warujayeng kertosono dengan program linier,” *Jurnal Teknik ITS*, vol. 3, no. 1, pp. D30–D35, 2014.
- [3] C. Arif and M. B. Caroline, “Optimasi sistem irigasi bawah permukaan untuk peningkatan produktivitas tanaman dan air dengan algoritma genetika,” *Jurnal Teknik Sipil dan Lingkungan*, vol. 8, no. 02, pp. 85–94, 2023.
- [4] M. Yanto, R. Sovia, and E. P. W. Mandala, “Jaringan syaraf tiruan perceptron untuk penentuan pola sistem irigasi lahan pertanian di kabupaten pesisir selatan sumatra barat,” *Sebatik*, vol. 22, no. 2, pp. 111–115, 2018.
- [5] A. Haris, N. Anggraini, and H. Sikumbang, “Teknologi irigasi cerdas pada sistem irigasi drip dengan algoritma ant colony optimization,” *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 9, no. 6, pp. 1289–1296, 2022.
- [6] R. Afandi, “Strategi pembentukan unit pengelola irigasi modern di daerah irigasi kromong, kabupaten mojokerto,” Ph.D. dissertation, Institut Teknologi Sepuluh Nopember, 2020.
- [7] D. Agriculture, “Different types of irrigation systems in agriculture — drip, gun sprinkler, center pivot irrigation,” YouTube, September 2022, available: <https://youtu.be/Z9HAy9EYKKs?si=Xg31EOGB0Da1VM9e>.
- [8] B. Y. Angguniko and S. Hidayah, “Rancangan unit pengelola irigasi modern di indonesia,” *Jurnal Irigasi*, vol. 12, no. 1, pp. 23–36, 2017.
- [9] Y. Zhang, “Utterance variation and its stylistic features in japanese language education using the dominant relevance approach,” *Applied Mathematics and Nonlinear Sciences*.
- [10] N. P. Rahayu, R. R. M. Putri, and A. W. Widodo, “Sistem pendukung keputusan (spk) pemilihan tanaman pangan berdasarkan kondisi tanah menggunakan metode electre dan topsis,” *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 2, no. 8, pp. 2323–2332, 2018, [Online]. [Online]. Available: <https://j-ptiik.ub.ac.id>