BAB II

TINJAUAN PUSTAKA

2.1 Matriks

2.1.1 Pengertian Matriks

Matriks adalah kumpulan bilangan (unsur) yang disusun menurut baris dan kolom. Bilangan-bilangan yang disusun tersebut disebut elemen-elemen atau komponen-komponen matriks. Nama sebuah matriks dinyatakan dengan huruf kapital. Banyak baris × banyak kolom dari suatu matriks menentukan ukuran dari matriks yang disebut ordo matriks. Secara umum. Matriks ***A*** yang berordo (m×n) dinyatakan sebagai berikut:

$$A_{m\times n} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix}$$

(Anton, 2000).

2.1.2 Macam-macam Matriks

Menurut Anton (2000), macam-macam matriks berdasarkan elemen- elemen penyusunnya adalah sebagai berikut:

1. Matriks nol.
   Matriks nol adalah matriks yang semua elemennya nol, dilambangkan dengan “0”.
2. Matriks bujur sangkar (persegi).
   Matriks bujur sangkar (persegi) adalah matriks yang jumlah baris dan kolomnya sama.
3. Matriks baris.
   Matriks baris adalah matriks yang hanya terdiri atas satu baris.
4. Matriks kolom.
   Matriks kolom adalah matriks yang hanya terdiri atas satu kolom.
5. Matriks diagonal.
   Matriks diagonal adalah matriks persegi yang semua elemennya nol, kecuali pada diagonal utamanya ada yang tidak nol.

2.1.3 Matriks Identitas

Matriks identitas (I_n) adalah matriks yang nilai-nilai elemen pada diagonal utama selalu 1.

$$I_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(Anton, 2000).

2.1.4 Operasi Matriks dan Sifat-sifatnya

1. Penjumlahan dua matriks
   Jika A dan B adalah matriks-matriks berukuran sama maka $A+B$ adalah matriks yang diperoleh dengan menambahkan anggota-anggota B dengan anggota-anggota A yang sepadan.
   Jika $A+B=C$ maka elemen-elemen C diperoleh dari penjumlahan elemen-elemen A dan B yang seletak, yaitu $c_{ij} = a_{ij} + b_{ij}$, untuk elemen C pada baris ke-i dan kolom ke-j. Akibatnya matriks A dan B dapat dijumlahkan apabila kedua matriks memiliki ordo yang sama.
   Sifat-sifat penjumlahan matriks:
   i. $A + B = B + A$ (hukum komutatif)
   ii. $A + (B + C) = (A + B) + C$ (hukum asosiatif)
   iii. $A + 0 = 0 + A = A$
   iv. Ada matrik B sedemikian sehingga $A+B = B+A = 0$, di mana $B = -A$.

2. Pengurangan matriks
   Jika A dan B adalah matriks-matriks berukuran sama maka selisih A-B adalah matriks yang diperoleh dengan mengurangkan anggota-anggota A dengan anggota-anggota B yang sepadan.
   Jika $A-B=C$, maka elemen-elemen C diperoleh dari pengurangan elemen-elemen A dan B yang seletak, yaitu $c_{ij} = a_{ij} - b_{ij}$.

3. Perkalian matriks dengan bilangan riil (skalar)
   Jika B adalah sembarang matriks dan a adalah sembarang skalar maka hasil kali aB adalah matriks yang diperoleh dengan mengalikan setiap anggota B dengan a

Jika a dan b bilangan riil, B dan C adalah dua matriks dengan ordo n sedemikian sehingga dapat dilakukan operasi yang berlaku sifat-sifat perkalian matriks dengan skalar:

i. $a\,(B + C) = aB + aC$
ii. $a\,(B – C) = aB – aC$
iii. $(a+b)\,C = aC + bC$
iv. $(a – b)\,C = aC – bC$
v. $(ab)\,C = a\,(bC)$

4. Perkalian dua matriks

Jika A adalah sebuah matriks $(m×n)$ dan B adalah matriks $(n×p)$, maka hasil kali $A×B$ adalah matriks $(m×p)$. Untuk mencari anggota dalam baris i dan kolom j dari $A×B$, memilih baris i dari matriks A dan kolom j dari matriks B. mengalikan anggota-anggota yang berpadanan dari baris dan kolom secara bersama-sama dan kemudian menjumlahkan hasil kalinya.

Elemen-elemen dari $A×B$ diperoleh dari hasil kali setiap baris pada matriks A dengan kolom pada matriks B, kemudian dijumlahkan menjadi satu elemen.

Sifat-sifat perkalian dua matriks:

i. $A(BC) = (AB)C$
ii. $A(B+C) = AB + AC$
iii. $(B+C)A = BA+ CA$
iv. $A(B – C) = AB – AC$
v. $(B – C)A = BA – CA$
vi. $a(BC) = (aB)C = B(aC)$
vii. $AI = IA =A$

(Anton, 2000).

2.1.5 *Transpose* Matriks

Jika A adalah sembarang matriks $(m×n)$ maka *tranpose* A diyatakan dengan A^T, didefinisikan sebagai matriks $(n × m)$ yang didapatkan dengan mempertukarkan baris dan kolom dari A; yaitu, kolom pertama dari A^T adalah baris pertama dari A, kolom ke dua dari A^T adalah baris ke dua dari A, dan seterusnya.

Adapun sifat-sifat utama dari operasi matriks *transpose* adalah sebagai berikut :

i. $((A)^T)^T = A$

ii. $(A+B)^T = A^T + B^T$ dan $(A-B)^T = A^T - B^T$
iii. $(kA)^T = kA^T$ dengan k adalah sembarang skalar
iv. $(AB)^T = B^T A^T$.

(Anton, 2000).

2.1.6 Matriks Kofaktor

Jika A adalah suatu matriks bujur sangkar maka minor anggota a_{ij} dinyatakan oleh M_{ij} dan didefinisikan sebagai sub-matriks yang masih tersisa setelah baris ke-i dan kolom ke-j dihilangkan dari A. Bilangan $(-1)^{i+j}M_{ij}$ dinyatakan oleh C_{ij} dan disebut kofaktor anggota a_{ij}.

(Anton, 2000).

2.1.7 Determinan Matriks

Jika A adalah sembarang matriks ($n \times n$), maka minor anggota a_{ij} dinyatakan oleh M_{ij} dan didefinsikan sebagai determinan sub–matriks yang masih tersisa setelah baris ke–i dan kolom ke–j dihilangkan dari A. Bilangan $(-1)^{i+j} M_{ij}$ dinyatakan oleh C_{ij} dan disebut kofaktor dari a_{ij}. Determinan suatu matriks A yang berordo ($n \times n$) dapat dihitung dengan mengalikan anggota–anggota pada sebarang baris atau kolom dengan kofaktornya dan menjumlahkan hasil kali yang dihasilkan, yaitu untuk setiap $1 \le i \le n$ dan $1 \le j \le n$,

$$det(A) = a_{1j}c_{1j} + a_{2j}c_{2j} + \ldots + a_{nj}c_{nj}$$

(perluasan kofaktor sepanjang kolom ke–j),

$$det(A) = a_{i1}c_{i1} + a_{i2}c_{i2} + \ldots + a_{in}c_{in}$$

(perluasan kofaktor sepanjang baris ke–i).

(Anton, 2000).

2.1.8 Matriks Adjoin

Jika A adalah sembarang matriks ($n\times n$) dan C_{ij} adalah kofaktor dari A maka matriks

$$\begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1n} \\ C_{21} & C_{22} & \cdots & C_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ C_{n1} & C_{n2} & \cdots & C_{nn} \end{bmatrix}$$

disebut matriks kofaktor dari *A*. Tranpos dari matriks ini disebut adjoin dan diyatakan oleh *adj(A)*.

(Anton, 2000).

2.1.9 Matriks Invers

Jika *A* adalah sebuah matriks bujur sangkar dan matriks *B* yang berukuran sama bisa didapatkan *AB* = *BA* = *I*, maka *A* disebut bisa dibalik dan *B* disebut invers dari *A*.

Sebuah matriks $B_{n\times n}$ adalah invers dari matriks $A_{n\times n}$ jika memenuhi $AB = BA = I_n$. Invers matriks *A* dirumuskan

$$A^{-1} = \frac{1}{det(A)} adj\,(A),$$

(Taha, 1996).

2.2 Sistem Persamaan Linear

2.2.1 Fungsi Linear

Fungsi $f(x_1, x_2, \dots, x_n)$ adalah fungsi linear jika dan hanya jika untuk himpunan konstanta $a_1, \dots, a_n$,

$f(x_1, \dots, x_n) = a_1x_1 + a_2x_2 + \dots + a_n x_n$.

(Winston, 1993).

2.2.2 Persamaan linear

Persamaan linear adalah suatu persamaan di mana suku–suku variabelnya berderajat satu. Persamaan linear dalam *n* variabel $x_1,\dots,x_n$ dapat dinyatakan dalam

$$a_1x_1 + a_1x_1 + \cdots + a_nx_n = b,$$

dengan $a_1,\dots,a_n$ dan *b* adalah konstanta riil.

(Winston, 1993).

2.2.3 Sistem Persamaan Linear

Sistem persamaan linear adalah sekelompok persamaan linear yang mempunyai satu penyelesaian persekutuan. Sistem persamaan linear dengan *m* persamaan dan *n* variabel dapat ditulis sebagai berikut:

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\
\vdots \qquad\qquad & \quad \vdots \\
a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m
\end{aligned}
$$

dengan a_{ij}, b_i adalah konstanta dengan $i=1,2,\ldots,m$ dan $j=1,2,3,\ldots,n$.

Persamaan di atas jika ditulis dalam bentuk matriks menjadi

$\boldsymbol{Ax} = \boldsymbol{b}$,

di mana

$$
A_{m\times n} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{m\times n} \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}
$$

(Winston, 1993).

2.2.4 Sistem Persamaan Linear Homogen

Sebuah sistem persamaan linear (SPL) dikatakan homogen jika semua suku konstanta ruas kanan sama dengan nol, yaitu sistem tersebut mempunyai bentuk:

$$
\boldsymbol{Ax} = \boldsymbol{b} \iff
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= 0 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= 0 \\
\vdots \qquad\qquad & \quad \vdots \\
a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= 0
\end{aligned}
$$

(Winston, 1993).

2.3 Pemrograman Linear

Keberhasilan sebuah teknik riset operasi pada akhirnya diukur berdasarkan penyebaran penggunaannya sebagai sebuah alat pengambilan keputusan. Sejak diperkenalkan di akhir 1940-an, pemrograman linear telah terbukti menjadi salah satu alat riset operasi yang paling efektif. Pemrograman linear (PL) adalah sebuah alat deterministik, yang berarti bahwa semua parameter model diasumsikan diketahui dengan pasti. Teknik PL memberikan analisis pasca-optimum dan analisis parametrik yang sistematis untuk memungkinkan pengambilan keputusan yang bersangkutan untuk menguji sensitivitas pemecahan optimum yang “statis” terhadap perubahan diskrit atau kontinu dalam berbagai parameter dari model (Taha, 1996).

Tipe penerapan pemrograman linear pada umumnya meliputi permasalahan pengalokasian sumber daya yang terbatas di tengah-tengah banyak aktivitas yang saling bersaing melalui jalan/cara yang terbaik (optimal). Lebih tepatnya, permasalahan tersebut meliputi pemilihan tingkat aktivitas-aktivitas tertentu yang saling bersaing untuk mendapatkan sumber daya yang terbatas, yang akan digunakan untuk menjalankan aktivitas-aktivitas tersebut. Pilihan tingkat aktivitas akan menentukan besarnya setiap sumber daya yang akan digunakan oleh setiap aktivitas (Hillier dan Lieberman, 2008).

2.3.1 Bentuk Umum Model Program Linear

Bentuk umum dari model program linear adalah

Maksimalkan $Z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$

dengan batasan

$$
\begin{array}{lcl}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & \leq & b_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & \leq & b_2 \\
\vdots & & \vdots \\
a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & \leq & b_m
\end{array}
$$

dan

$x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0.$

Suatu fungsi yang akan dimaksimalkan, $c_1x_1 + c_2x_2 + \cdots + c_nx_n$ disebut fungsi tujuan, dan batasan-batasannya disebut kendala. Kedala *m* pertama (dengan sebuah fungsi dengan semua variabel $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n$) disebut kendala fungsional, dan batasan $x_j \geq 0$ disebut kendala nonnegatif.

Beberapa bentuk lainnya adalah:

1. Meminimalkan fungsi tujuan:
   Minimalkan $Z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$.
2. Beberapa kendala fungsional pertidaksamaan dengan tanda lebih besar atau sama dengan: $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geq b_i$ untuk beberapa nilai *i*.

3. Beberapa kendala fungsional dengan bentuk persamaan: $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b\underline{i}$ untuk beberapa nilai *i*.
4. Menghilangkan kendala nonnegatif untuk beberapa variabel keputusan: x_j tidak dibatasi (*unrestricted*) untuk beberapa nilai *j*.

(Hillier dan Lieberman, 2008).

2.3.2 Model Pemrograman Linear

Beberapa simbol yang umum digunakan untuk menunjukkan berbagai komponen model pemrograman linear. Daftar simbol dengan tafsirannya untuk permasalahan umum pengalokasian sumber daya ke aktivitas sebagai berikut:

Z : nilai dari semua standar performasi,
x_j : tingkat aktivitas *j* (untuk $j = 1,2,...,n$),
c_j : penambahan terhadap *Z* yang diakibatkan oleh peningkatan tiap unit di tingkat aktivitas *j*,
b_i : jumlah sumber daya *i* yang tersedia untuk aktivitas (untuk $i = 1,2,...,m$),
a_{ij} : jumlah sumber daya *i* yang dipakai oleh tiap unit aktivitas *j*.

Suatu model akan membuat permasalahan menjadi suatu bentuk pengambilan keputusan mengenai tingkat aktivitas sehingga $x_1, x_2, x_3,..., x_n$ disebut variabel keputusan. Nilai c_j, b_i, dan a_{ij} (untuk $i = 1,2,...,m$ dan $j = 1,2,...,n$) adalah konstanta untuk suatu model, disebut juga parameter suatu model.

(Hillier dan Lieberman, 2008).

2.3.3 Variabel Keputusan

Variabel keputusan adalah variabel persoalan yang akan mempengaruhi nilai tujuan yang akan dicapai. Di dalam proses pemodelan, penentuan variabel keputusan harus dilakukan terlebih dahulu sebelum merumuskan fungsi tujuan dan kendala-kendalanya.

Cara untuk menentukan variabel-variabel ini adalah dengan mengajukan pertanyaan: “Keputusan apa yang harus dibuat agar nilai fungsi tujuan menjadi maksimum atau minimum”.

Pertanyaan-pertanyaan seperti,

1. Berapa banyak produk yang harus diproduksi dan persediaan harus tersedia pada periode tertentu agar laba maksimum atau biaya minimum?
2. Saham mana yang harus dibeli dan berapa banyak harus dibeli agar tingkat pengembalian maksimum?
3. Investasi mana yang harus dipilih agar *Net Present Value* (NPV) maksimum?

adalah pertanyaan-pertanyaan yang sering muncul pada praktik organisasi bisnis dan industri. Jawaban terhadap pertanyaan-pertanyaan tersebut akan membawa kepada variabel keputusan yang sedang dicari.

(Siswanto, 2007).

2.3.4 Fungsi Tujuan

Dalam model program linear, tujuan yang hendak dicapai harus diwujudkan kedalam sebuah fungsi linear. Selanjutnya, fungsi tersebut dimaksimumkan terhadap kendala-kendala yang ada. Contoh-contoh tujuan yang hendak dicapai di dalam praktek manajemen adalah,

1. pemaksimalan laba perusahaan
2. peminimuman biaya distribusi
3. peminimuman waktu kerja
4. pemaksimalan daya jangkau media promosi
5. dan lain-lain.

(Siswanto,2007).

2.3.5 Kendala Fungsional

Manajemen menghadapi berbagai kendala untuk mewujudkan tujuan-tujuan. Kenyataan tentang eksistensi kendala-kendala tersebut selalu ada, misal:

1. Keputusan untuk meningkatkan volume produksi dibatasi oleh faktor-faktor seperti kemampuan mesin, jumlah sumber daya manusia (SDM), dan teknologi yang tersedia.
2. Manajer produksi harus menjaga tingkat produksi agar permintaan pasar terpenuhi.
3. Agar kualitas produk yang dihasilkan memenuhi standar tertentu maka unsur bahan baku yang digunakan harus memenuhi kualitas minimum, dan lain-lain.

Kendala dengan demikian dapat diumpamakan sebagai pembatas terhadap kumpulan keputusan yang mungkin dibuat dan harus dituangkan ke dalam fungsi linear. Dalam hal ini, ada tiga macam kendala:

1. Kendala berupa pembatas
2. Kendala berupa syarat
3. Kendala berupa keharusan

Ketiga macam kendala tersebut akan selalu dijumpai di dalam setiap susunan kendala kasus pemrograman linear, baik yang sejenis maupun gabungan dari ketiganya.

(Siswanto, 2007)

2.3.6 Variabel *Slack* dan *Surplus*

Variabel *slack* adalah variabel yang menyatakan penggunaan jumlah kelebihan sumber daya untuk menjadikan kendala bertanda kurang dari (≤) menjadi persamaan (=). Sedangkan variabel *surplus* adalah variabel yang menyatakan penyerapan persamaan sisi kiri untuk memenuhi batasan minimum sumber daya sehingga menjadikan konstrain bertanda (≥) menjadi persamaan (=).

Jika nilai variabel *slack* atau *surplus* tidak sama dengan nol, maka perubahan kendala sebesar minus *slack* atau *surplus* belum berpengaruh pada nilai optimum. Namun, jika nilai *slack* atau *surplus* sama dengan nol, maka variabel terkait menjadi variabel basis (variabel yang nilainya tidak sama dengan nol pada sembarang iterasi).

(Hillier dan Lieberman, 2008).

2.3.7 Beberapa Istilah dalam Menyelesaikan Program Linear

1. Solusi layak (*feasible solution*) adalah suatu solusi atau penyelesaian *feasible* di mana semua kendala yang ada dipenuhi.
2. Daerah *feasible* merupakan kumpulan dari semua penyelesaian yang *feasible*.
3. Penyelesaian optimal merupakan penyelesaian *feasible* yang memiliki nilai paling menguntungkan untuk fungsi tujuan.

(Hillier dan Lieberman, 2008).

2.3.8 Asumsi-asumsi dalam Program Linear

Menurut Hillier and Lieberman, (2008), program linear mempunyai beberapa asumsi diantara:

1. Proporsionalitas (*Proportionality*)

   Proporsionalitas merupakan asumsi mengenai kegiatan salah satu variabel keputusan yang dipertimbangkan secara independen dari yang lainnya. Oleh karena itu, perhatikan kasus di mana hanya satu di antara *n* kegiatan yang dilaksanakan.

   kegiatan k, $x_j = 0$, untuk setiap $j = 1,2,\ldots, n$ kecuali $j = k$.

   Asumsinya adalah bahwa

   a. Ukuran keseluruhan mengenai nilai fungsi tujuan Z adalah sama dengan $c_k x_k$.

   b. Pemakaian setiap sumber daya i adalah sama dengan $a_{ik} x_k$.

   Misalkan,

   i. $Z = c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$

      Setiap penambahan satu unit x_1 menaikkan Z dengan c_1, kemudian setiap penambahan satu unit x_2 menaikkan Z dengan c_2 dan seterusnya.

   ii. $a_{i1} x_1 + a_{i2} x_2 + \cdots + a_{in} x_n \leq b_i,$

      Setiap penambahan satu unit x_1 akan menaikkan penggunaan sumber daya ke–1 dengan a_{11}, kemudian satu unit x_2 menaikkan penggunaan sumber daya ke – 1 dengan a_{12} dan seterusnya.

   Dengan demikian, kedua asumsi di atas proporsional (sebanding) secara langsung dengan tingkat setiap kegiatan k yang dilaksanakan secara tersendiri $(k = 1,2,\ldots,n)$.

2. Adivitas (*Additivity*)

   Asumsi adivitas berlaku baik pada fungsi tujuan maupun fungsi–fungsi pada ruas kiri dan fungsi kendala. Untuk kedua jenis fungsi tersebut asumsinya berkaitan dengan perbandingan antara nilai fungsi total yang diperoleh pada saat melaksanakan kegiatan–kegiatan bersama–sama ($x_1, x_2, \ldots, x_n$) dan kontribusi individual kepada nilai fungsi sebagai akibat melaksanakan kegiatan itu masing–masing. Yang dimaksud dengan kontribusi individual adalah $c_j x_j$ untuk fungsi tujuan dan $a_{ij} x_j$ untuk suatu fungsi kendala. Dengan demikian, asumsi adivitas adalah bahwa untuk setiap fungsi, nilai fungsi total dapat diperoleh dengan menjumlahkan kontribusi individu dari masing–masing kegiatan. Misalkan,

$$Z = c_1x_1 + c_1x_2$$

di mana,

$$x_1 = p \text{ dan } x_1 = q$$

sehingga,

$$Z = c_1p + c_2q$$

Andaikan x_1 bertambah 1 unit, maka diperoleh:

$$Z = c_1(p + 1) + c_2q$$

Nilai Z ini sama juga dengan penambahan langsung dari kenaikan x_1 kepada harga Z mula–mula tanpa mengurangi kegiatan–kegiatan Z yang diperoleh dari kegiatan ke dua (x_2), yaitu:

$$Z = c_1p + c_2q + c_1$$

3. *Divisibilitas*
   Kadang–kadang penyelesaian optimal yang diperoleh pada program linear sering merupakan penyelesaian non–integer. Oleh karena itu, asumsi divisibilitas menjamin bahwa unit–unit kegiatan (variabel keputusan) dapat dibagi ke dalam bagian yang sekecil–kecilnya, sehingga nilai–nilai non–integer pada variabel keputusan adalah mungkin. Demikian pula halnya dengan fungsi tujuan yang dihasilkan.

4. Kepastian (*Deterministik*)
   Asumsi mengenai kepastian adalah bahwa semua nilai–nilai (a_{ij}, b_i, c_j) merupakan konstanta–konstanta yang diketahui. Dalam praktiknya asumsi ini jarang dipenuhi secara tepat, karena model–model program linear biasanya dirumuskan untuk memilih suatu tindakan pada waktu yang akan datang, sehingga konstanta–konstanta yang dipakai akan didasarkan pada suatu prediksi (perkiraan) mengenai kondisi–kondisi di masa mendatang.

2.3.9 Model Program Linear yang Diperluas

Suatu perhitungan akan lebih mudah dalam menghadapi bentuk–bentuk persamaan dibandingkan dengan bentuk pertidaksamaan. Dengan alasan tersebut, maka bentuk pertidaksamaan dari model program linear akan diubah menjadi bentuk persamaan. Perubahan ini dapat dilakukan dengan penambahan suatu variabel tambahan atau variabel *slack* pada tiap–tiap kendala pertidaksamaan. Sementara itu kendala–kendala non–negatif dapat diabaikan sebagai pertidaksamaan, karena dalam perhitungan kendala non negatif tidak diikutsertakan.

Model program linear yang telah diperluas bentuknya menjadi sebagai berikut :

Maksimum $Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$

dengan kendala,

$$
\begin{array}{lcl}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + s_1 & = & b_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + s_2 & = & b_2 \\
\vdots & & \vdots \\
a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n + s_m & = & b_m \\
x_j \geq 0, j = 1,2, \ldots, n + m & &
\end{array}
$$

di mana,

$s_1, s_2, \ldots, s_m$ adalah variabel *slack*.

Model program linear dapat juga ditulis dengan menggunakan notasi matriks, seperti sebagai berikut:

Maksimum $Z = \boldsymbol{cx}$

dengan kendala, $\boldsymbol{Ax} \leq \boldsymbol{b}$

$\boldsymbol{x} \geq 0$

di mana,

c : vektor baris $[c_1\ c_2\ c_3 \ldots c_n]$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix} \quad \boldsymbol{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

A = merupakan matriks koefisien kendala

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{m \times n} \end{bmatrix}$$

Sedangkan pada program linear dalam bentuk yang diperluas, modelnya adalah sebagai berikut:

Maksimumkan $Z = \mathbf{c}^{\mathbf{T}}\mathbf{x}$

dengan kendala ,

$$[\mathbf{A}, \mathbf{I}]\begin{bmatrix} \mathbf{x} \\ \mathbf{s_m} \end{bmatrix} = \mathbf{b}$$

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{s_m} \end{bmatrix} \geq \mathbf{o}$$

di mana,

$\mathbf{I}$ = merupakan matriks identitas

$\mathbf{S_m}$ = vektor kolom dari variabel *slack*

$\mathbf{o}$ = vektor kolom 0 yang mempunyai (n + m) unsur.

(Hillier and Lieberman, 2008).

2.3.10 Langkah-langkah Perumusan Pemrograman Linear

1. Menentukan jenis permasalahan pemrograman linear
   a. Jika permasalahan membicarakan keuntungan (*profit*), maka jenis permasalahan PL adalah maksimalisasi.
   b. Jika permasalahan membicarakan biaya (*cost*), maka jenis permasalahan PL adalah minimalisasi.
   c. Jika ada informasi tentang selisih antara hasil penjualan (*sales*) dan biaya dengan pokok pembicaraan *profit*, maka jenis permasalahannya adalah maksimalisasi.
2. Mendefinisikan variabel keputusan (*decision variable*), yaitu pernyataan dalam permasalahan yang hendak dicari penyelesaiannya. Beberapa hal yang harus diperhatikan adalah:
   a. Banyaknya koefisien variabel keputusan membantu dalam mengidentifikasikan variabel-variabel keputusan.
   b. Jika x dimisalkan sebagai peubah keputusan berkaitan dengan kursi yang diproduksi, maka $x \neq$ kursi, tetapi x = banyaknya kursi yang diproduksi.
3. Merumuskan fungsi tujuan/sasaran (*objective function*)
   a. Jenis permasalahan PL dan definisi variabel keputusan akan merumuskan fungsi tujuan.
   b. Jika variabel keputusan terdefinisi dengan jelas, maka fungsi tujuan akan mudah ditetapkan.
4. Merumuskan model kendala/syarat/ batasan (*constraint*). Dua pendekatan umum perumusan model kendala:

a. Pendekatan ruas "kanan"
   i. Ruas kanan suatu kendala tunggal dan konstan.
   ii. Maksimalisasi: ruas kanan sering menyatakan "total sumber daya yang ada". Prosedur pembentukannya:
      - mengidentifikasi nilai total sumber daya dan sesuaikan tanda pertidaksamaan dengan masing-masing total sumber daya, biasanya "≤".
      - mengelompokkan peubah keputusan yang terkait di sebelah kiri tanda pertidaksamaan .
      - menentukan koefisien setiap peubah keputusan. Model kendala terbentuk
   iii. Minimalisasi: ruas kanan sering menyatakan "minimal sumber daya yang dibutuhkan". Prosedur sama dengan merumuskan maksimalisasi, kecuali tanda pertidaksamaan, biasanya "≥".

b. Pendekatan ruas "kiri"
   Semua nilai koefisien dan peubah-peubah keputusan disusun dalam bentuk matriks. Setelah matriks ini terbentuk, mengidentifikasikan nilai-nilai ruas kanan dan menambahkan tanda pertidaksamaan.

5. Menetapkan syarat non negatif
   Setiap peubah keputusan dari kedua jenis permasalahan PL tidak boleh negatif (harus lebih besar atau sama dengan nol)
   (Hillier dan Lieberman, 2008).

2.3.11 Solusi Program Linear

Penyelesaian masalah pemrograman linear bisa dilakukan dengan menggunakan metode simpleks. Namun, penggunaan metode simpleks hanya terbatas untuk mengatasi masalah kecil dan kurang efisien untuk menyelesaiakan masalah yang besar. Pada tahun 1984 seorang ahli matematika muda di AT&T Bell Laboratories, Narendra Karmarkar, berhasil mengembangkan algoritma baru untuk pemrograman linear dengan jenis pendekatan *titik interior*.

Beberapa varian dari pendekatan *titik interior* antara lain algoritma Khachiyan, algoritma Karmarkar, dan algoritma *Affine Scaling*. Algoritma Khachiyan diselesaikan pertama kali oleh Leonid Kachiyan pada tahun 1979, algoritma ini termasuk algoritma terburuk dalam menyelesaikan masalah polinomial pada program linear. Algoritma *Karmakar* diselesaikan pertama kali oleh Narendra Karmakar pada tahun 1984 algoritma ini terinspirasi dari algoritma Khachiyan dan diklaim lebih baik di banding algoritma Khachiyan. Dan, algoritma *Affine Scaling*, algoritma ini termasuk algoritma yang paling populer sejak tahun 1990-an dibanding algoritma yang lainnya.

(Chong,dkk, 2001)

2.4 Algoritma *Affine Scaling*

2.4.1 Sejarah Algoritma *Affine Scaling*

Perkembangan baru dalam penelitian operasional selama era 1980-an adalah penemuan pendekatan titik interior (*interior point approach*) untuk menyelesaikan masalah pemrograman linear. Penemuan ini terjadi ditahun 1984 oleh seorang ahli matematika muda di AT&T Bell Laboratories, Narendra Karmarkar, saat berhasil mengembangkan algoritma baru untuk pemrograman linear dengan jenis pendekatan ini. Walaupun algoritma ini hanya berhasil saat dibandingan dengan metode simplek, konsep solusi kunci yang dibahas sepertinya berpotensi mampu menyelesaikan masalah pemrograman linear yang memiliki jumlah variabel banyak melebihi yang bisa dilakukan oleh metode simpleks. Banyak peneliti yang kemudian berupaya memodifikasi algoritma Karmarkar sehingga banyak kemajuan yang terjadi dan sejumlah algoritma yang menggunakan pendekatan *interior point* dikembangkan.

Walaupun berbeda dengan metode simplek, Algoritma Karmarkar memiliki beberapa karakteristik yang sama yaitu algoritma iteratif. Algoritma ini dimulai dengan mengidentifikasi solusi percobaan yang berada pada daerah *feasible*. Di masing-masing iterasi, terjadi perpindahan dari solusi percobaan awal menuju solusi percobaan yang lebih baik dan proses ini berlangsung hingga mencapai solusi percobaan yang optimal.

Perbedaan besar terletak pada solusi-solusi percobaan. Dalam metode simplek solusi percobaan merupakan solusi *corner point feasible* (CPF) sehingga segala gerakan berlangsung di sepanjang batas daerah *feasible*. Dalam Algoritma Karmakar solusi-solusi percobaanya adalah *interior point*, yaitu titik-titik yang berada dalam daerah *feasible*. Oleh karena itu algoritma Karmakar dan variannya sering disebut algoritma *interior point*. Algoritma ini dirancang untuk menyelesaikan masalah yang memiliki variabel yang banyak secara efisien, tetapi algoritma ini kurang efisien dibandingkan dengan metode simpleks untuk yang memiliki variabel sedikit.

(Hillier dan Lieberman, 2008).

2.4.2 Pengertian Metode *Interior Point* dan Penyelesaian Percobaan Awal

Algoritma *Affine Scaling* adalah algoritma yang memotong atau menembus titik dalam dari daerah *feasible* untuk mencapai solusi yang optimum. Titik Interior adalah titik–titik yang berada di daerah *feasible* (Hillier and Lieberman, 2008). Pendekatan umum dari algoritma ini adalah untuk memperoleh suatu urutan penyelesaian percobaan awal yang semakin baik sampai tercapai suatu penyelesaian optimal. Dalam menentukan percobaan awal optimal atau tidak, dapat dilihat pada akhir tiap iterasi yang dilakukan bersifat konvergen dari nilai yang diperoleh.

Langkah awal akan dibahas bagaimana menentukan penyelesaian percobaan awal pada iterasi pertama. Penyelesaian percobaan awal dapat ditentukan dengan mengambil sembarang titik ($x \geq 0$) yang terletak di dalam batas–batas daerah *feasible*. Titik yang diperoleh inilah yang disebut dengan titik dalam (titik interior).

Misalkan:

$Max\ \ Z = c_1x_1 + c_2x_2$

dengan kendala,

$$a_{11}x_1 + a_{12}x_2 \leq b_1$$
$$x_1, x_2 \geq 0 \qquad (2.1)$$

$D(x_1, x_2)$ adalah penyelesaian percobaan awal yang dipilih sembarang dari setiap titik yang berada di dalam daerah *feasible*. Kemudian persamaan (2.1) diperluas dengan menambahkan variabel yang ke-tiga (x_3) pada persamaan (2.1) yang merupakan variabel *slack*. Oleh karena itu, masalahnya berubah menjadi:

$Max\ \ Z = c_1x_1 + c_2x_2$

dengan kendala,

$$a_{11}x_1 + a_{12}x_2 + x_3 \le b_1$$

$$x_1, x_2, x_3 \ge 0$$

Dengan demikian, penyelesaian percobaan awal yang dipilih adalah $D(x_1, x_2, x_3)$, dimana $D(x_1, x_2, x_3)$ adalah titik dalam (interior) pada himpunan daerah *feasible*.

(Hillier dan Lieberman, 2008).

2.4.3 Konsep Dasar Algoritma *Affine Scaling*

Tiga konsep dasar dalam menyelesaikan permasalahan menggunakan algoritma Affine Scaling antara lain:

Konsep 1. Menunjukan melalui bagian dalam (interior) daerah *feasible* menuju solusi yang optimal.

Konsep 2. Bergerak dengan sebuah arah yang meningkatkan nilai fungsi tujuan pada kemungkinan rata-rata yang paling cepat.

Konsep 3. Mengubah daerah *feasible* untuk memindahkan solusi percobaan ke dekat solusi yang optimal sehingga memungkinkan sebuah peningkatan besar saat konsep 2 diterapkan.

Bentuk umum masalah pemrograman linear sebagai berikut:

Maksimumkan $Z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$

dengan kendala

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &\le b_1 \\ \vdots \qquad\qquad &\ \ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &\le b_m \\ x_1, \ldots, x_n &\ge 0 \end{aligned}$$

(Hillier dan Lieberman, 2008).

2.5 Beras dan Gabah

2.5.1 Gabah

Gabah merupakan butir padi yang sudah lepas dari tangkainya dan masih berkulit. Gabah memiliki nilai susut mutu (rendemen). Menurut Nugraha dkk. (1998), nilai rendemen beras giling dipengaruhi oleh banyak faktor yang terbagi dalam tiga kelompok. Kelompok pertama adalah faktor yang mempengaruhi rendemen melalui pengaruhnya terhadap mutu gabah sebagai bahan baku dalam proses penggilingan yang meliputi varietas, teknik budidaya, cekaman lingkungan , agroekosistem, dan iklim. Kelompok ke dua merupakan faktor penentu rendemen yang terlibat dalam proses konversi gabah menjadi beras, yaitu teknik penggilingan dan alat penggilingan. Kelompok ketiga menunjukan kualitas beras terutama derajat sosoh yang diinginkan, karena semakin tinggi derajat sosoh maka rendemen akan semakin rendah. Derajat sosoh adalah tingkat pelepasan lapisan aleuron dan lembaga dari butir beras selama proses penyosohan. Jika derajat sosoh 80%, berarti masih ada 20% lapisan aleuron yang menempel pada butir beras. Bila derajat sosoh mencapai 100% berarti tidak ada lapisan aleuron yang menempel pada butir beras. Makin tinggi derajat sosoh makin bersih penampakan beras. Namun, penyosohan yang lebih lama dengan tujuan untuk lebih mengilapkan beras akan menurunkan kandungan protein beras.

Besarnya rendemen dapat dihitung dengan:

$$\text{Rendemen} = \frac{\text{Berat Beras Sosoh}}{\text{Berat Gabah Awal}} \times 100\%$$

(Ismunadji dkk, 1988).

2.5.2 Penggilingan Gabah

Kegiatan Operasional penggilingan padi antara lain:

1. Penjemuran
   Penjemuran merupakan kegiatan yang harus diperhatikan sejak awal karena kualitas jemur yang bagus sangat mempengaruhi mutu beras yang dihasilkan. Penjemuran harus langsung dilakukan pada saat hari pertama panen atau paling lambat dua hari setelah panen, hal ini dilakukan guna menghindari penyimpanan padi yang masih basah karena akan mengurangi kualitas padi sendiri.
2. Penyimpanan
   Setelah gabah benar-benar kering dengan kualitas simpan, gabah bisa masuk ke gudang penyimpanan. Gudang penyimpanan yang digunakan untuk menyimpan gabah kering giling dan beras harus memiliki sirkulasi udara yang bagus sehingga gabah bisa bertahan

pada standar gabah kering simpan yang normal dan tidak cepat rusak.

3. Penggilingan
   Penggilingan beras berfungsi untuk menghilangkan sekam dari bijinya dan lapisan aleuron, sebagian maupun seluruhnya agar menghasilkan beras yang putih serta beras pecah sekecil mungkin. Setelah gabah dikupas kulitnya dengan mengunakan alat pecah kulit, kemudian gabah tersebut dimasukan ke dalam alat penyosoh untuk membuang lapisan aleuron yang menempel pada beras. Selama penyosohan terjadi, penekanan terhadap butir beras sehingga terjadi butir patah. Menir merupakan kelanjutan dari butir patah menjadi bentuk yang lebih kecil daripada butir patah.

(Suryana,dkk, 2001)

2.5.3 Beras

Beras merupakan sumber utama Kalori bagi sebagian besar penduduk inodonesia. Pangsa beras pada konsumsi kalori total adalah 54,3% atau dengan kata lain setengah dari *intake* kalori masyarakat Indonesia bersumber dari beras (Suryana,dkk, 2001).