Chapter 2
Simple Examples in GAMS

The main concept that is developed in this chapter is explaining some optimization categories that can be modeled in GAMS. These models include linear programming (LP), mixed integer programming (MIP), nonlinear programming (NLP), quadratic programming (QCP), mixed integer non-linear programming (MINLP), and multi-objective optimization problems.

Understanding the materials presented and discussed in this chapter does not require any background in power system studies. This makes it suitable for anybody who might be interested to start optimization modeling in GAMS.

2.1 Different Types of Optimization Models

The general form of an optimization problem is as follows:

$$\min_{X} f(X, I) \tag{2.1a}$$

$$G(X, I) \leq 0 \tag{2.1b}$$

$$H(X, I) = 0 \tag{2.1c}$$

where f is objective function, G and H are set of equality and inequality constraints, respectively, I is the input data of the optimization problem, and X is the set of decision variables that should not only satisfy G and H but also optimizes the f value.

A. Soroudi, *Power System Optimization Modeling in GAMS*,
DOI 10.1007/978-3-319-62350-4_2

2.1.1 Linear Programming (LP)

The linear programming problems are those that f, G, H are all linear in (2.1).

2.1.1.1 LP Example

A simple linear programming example is as follows:

$$\min_X \mathrm{OF} = x_1 + 3x_2 + 3x_3 \tag{2.2a}$$

$$x_1 + 2x_2 \geq 3 \tag{2.2b}$$

$$x_3 + x_2 \geq 5 \tag{2.2c}$$

$$x_1 + x_3 = 4 \tag{2.2d}$$

GCode 2.1 LP Example (2.2)

```
variables x1,x2,x3,of;
Equations
eq1
eq2
eq3
eq4;
eq1 .. x1+2*x2 =g=3;
eq2 .. x3+x2 =g=5;
eq3 .. x1+x3 =e=4;
eq4 .. x1+3*x2 +3*x3=e=OF;
model LP1 /all/;
Solve LP1 US LP min of;
display x1.l,x2.l,x3.l,of.l;
```

The optimal solution is $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \mathrm{OF} \end{bmatrix} = \begin{bmatrix} 0.333 \\ 1.333 \\ 3.667 \\ 15.333 \end{bmatrix}$. Clicking on the model statistic tap shows that this model has four blocks of equations (four single equations). It has also four variables ($x_{1,2,3}$,OF). The solution report would be as follows:

```
S O L V E S U M M A R Y
MODEL LP1 OBJECTIVE of
TYPE LP DIRECTION MINIMIZE
SOLVER CPLEX FROM LINE 12
**** SOLVER STATUS 1 Normal Completion
**** MODEL STATUS 1 Optimal
```

(continued)

```
**** OBJECTIVE VALUE 15.3333
RESOURCE USAGE, LIMIT 0.016 1000.000
ITERATION COUNT, LIMIT 2 2000000000
```

It means that the solver has successfully solved the model and the solution is globally optimal. The solver used for solving the model is CPLEX [1]. It states that the value of objective function is 15.333. It also gives the user some info regarding the computational burden needed for solving the problem. RESOURCE USAGE is indicating how much time was needed to solve the model in seconds (0.016 s) and what was the maximum time allowed to do so (1000 s). The number of iterations needed for finding the optimal solution is two in this case. The default value for this limit is 2,000,000,000.

Clicking on the SolEQU tab would show the following info regarding the model:

```
                  LOWER   LEVEL   UPPER   MARGINAL
----  EQU  eq1    3       3       +INF    0.333
----  EQU  eq2    5       5       +INF    2.333
----  EQU  eq3    4       4       4       0.667
----  EQU  eq4    0       0       0       -1
```

The lower limits of eq1,eq2 have some finite values (3,5) but their upper limits are $+\infty$. This means that these two equations are of $\geq$ type. Equations eq3,eq4 have equal values for lower and upper limits. This means that these equations are of equality type. The interesting part of the analysis is given in marginal column (the last column). As it can be seen, the level values of eq1,eq2,eq3 are equal to their lower limits. This has a certain meaning that these constraints are binding constraints. This means that if the lower limits are changed then the objective function value would change. The marginal values actually show the sensitivity coefficients of objective function to these equations. Let's check them in more detail. The marginal value of eq1 is 0.333. This means that $\frac{\Delta \mathrm{OF}}{\Delta \mathrm{RHS}\ \mathrm{eq1}} = 0.333$. The right-hand side of eq1 is 3 so if it is increased to 3.2 then $\Delta\mathrm{RHS}\ \mathrm{eq1} = 3.2 - 3 = 0.2$. The marginal value indicates that the new objective function would be $15.333 + 0.333 * 0.2 = 15.3996$. If the GAMS model is solved using the new RHS value of eq1 (3.2) then OF would be 15.4. The obtained value is close but not exactly what we were expecting but why? This is because the marginal values are accurate for very small change in RHS values of the equations. If the variations are small enough then the approximation would be accurate enough.

The marginal values constitute the values of dual variables. The decision maker can understand which constraint is binding (has nonzero marginal value) and also shows the most influential constraint on objective function (the biggest marginal value).

Clicking on the SolVAR tab would show the following info regarding the model:

```
                  LOWER   LEVEL    UPPER   MARGINAL
----  VAR   x1    -INF    0.333    +INF    0
----  VAR   x2    -INF    1.333    +INF    0
----  VAR   x3    -INF    3.667    +INF    0
----  VAR   OF    -INF    15.333   +INF    0
```

The variable attributes are given in the table above. The lower and upper limits of all variables are $-\infty$ and $+\infty$, respectively. The marginal values are zero (this is because no bound is defined for variables). Suppose that we define a lower limit for x_2 which is $2 \leq x_2$. Let's see the impact on marginal values of equations and variables: The code would be as follows:

```
Variables x1,x2,x3,of;
Equations eq1,eq2,eq3,eq4;
eq1 .. x1+2*x2 =g=3;
eq2 .. x3+x2 =g=5;
eq3 .. x1+x3 =e=4;
eq4 .. x1+3*x2 +3*x3=e=OF;
Model LP1 /all/;
x2.lo=2;
Solve LP1 US LP min of;
display x1.l,x2.l,x3.l,of.l;
```

Clicking on the SolEQU tab would show the following info regarding the model:

```
                  LOWER   LEVEL   UPPER   MARGINAL
----  EQU   eq1   3       5       +INF    0
----  EQU   eq2   5       5       +INF    2
----  EQU   eq3   4       4       4       1
----  EQU   eq4   0       0       0       -1
```

This table shows that eq1 is no longer the binding equation. This means that small change of RHS (which is 3 here) won't change the objective function. The marginal value of this equation is zero. The eq2 and eq3 are binding equations (they have nonzero marginal values and also their level is equal to their lower value).

Clicking on the SolVAR tab would show the following info regarding the model:

```
                    LOWER    LEVEL    UPPER    MARGINAL
----   VAR    x1    -INF     1        +INF     0
----   VAR    x2    2        2        +INF     1
----   VAR    x3    -INF     3        +INF     0
----   VAR    OF    -INF     16       +INF     0
```

The marginal values of all variables are zero except $x2$ which is 1. This means that setting a lower limit for $x2$ caused this situation. Originally, the optimal value of $x2$ was 1.333 and now the lower limit (which is set to be 2) stops it from reaching its optimal value. Any possible decrease in $x2$ can help reducing the overall objective function.

GCode 2.2 Finding the boundaries of a variable example (2.3)

```
Variables x1,x2,x3,of;
Equations
eq1,eq2,eq3,eq4;
eq1 .. x1+2*x2 =l=3;
eq2 .. x3+x2 =l=2;
eq3 .. x1+x2+x3 =e=4;
eq4 .. x1+2*x2 -3*x3=e=OF;
Model LP1 /all/;
x1.lo=0; x1.up=5; x2.lo=0; x2.up=3; x3.lo=0; x3.up=2;
Solve LP1 US LP max of;
display x1.l,x2.l,x3.l,of.l;
Solve LP1 US LP min of;
display x1.l,x2.l,x3.l,of.l;
```

2.1.1.2 Boundary Determination Example

Sometimes it is needed to find the maximum and minimum of the objective function for a given model. This means that the problem should be solved two times. The example given in (2.3) is describing such a situation.

$$\min / \max_{X} \text{OF} = x_1 + 2x_2 - 3x_3 \tag{2.3a}$$

$$x_1 + 2x_2 \leq 3 \tag{2.3b}$$

$$x_3 + x_2 \leq 2 \tag{2.3c}$$

$$x_1 + x_2 + x_3 = 4 \tag{2.3d}$$

$$0 \le x_1 \le 5 \tag{2.3e}$$

$$0 \le x_2 \le 3 \tag{2.3f}$$

$$0 \le x_3 \le 2 \tag{2.3g}$$

Please pay special attention to (2.3e)–(2.3g). These three constraints can be easily treated in GAMS using *.lo* and *.up* statements. In order to reduce the number of equations in the model it should be avoided defining them as six extra equations. The Gcode 2.2 for solving (2.3) is provided as follows:

The problem is solved and the solutions are obtained as $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \text{OF} \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 1 \\ 0 \end{bmatrix}_{\max}$ and $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \text{OF} \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 2 \\ -4 \end{bmatrix}_{\min}$. The application of this model is in interval optimization [2], fuzzy optimization [3], and DC power flow (which will be discussed in Chap. 6)

2.1.2 Mixed Integer Programming (MIP)

In mixed integer programming (MIP) problems, the decision maker is faced with constraints and objective function that are linear but there exist some integer/binary variables.

2.1.2.1 MIP Example

A MIP example is given for clarification as follows:

GCode 2.3 MIP example (2.4)

```
Variables x,of;
Binary variable y;
Equations eq1, eq2,eq3;
eq1 .. −3*x+2*y =g=1;
eq2 .. −8*x+10*y =l=10;
eq3 ..    x+y=e=OF;
Model MIP1 /all/;
x.up=0.3;
Solve MIP1 US MIP max of;
display y.l,x.l,of.l;
```

$$\max_{x,y} \text{OF} = x + y \tag{2.4a}$$

$$-3x + 2y \geq 1 \tag{2.4b}$$

$$-8x + 10y \leq 10 \tag{2.4c}$$

$$y \in \{0, 1\}, 0.3 \leq x \tag{2.4d}$$

y is a binary variable and x is a real number. The GAMS code for solving (2.4) is provided in GCode 2.3:

By running the GAMS code the optimal solution is found as follows: $\begin{bmatrix} x \\ y \\ \text{OF} \end{bmatrix} = \begin{bmatrix} 0.3 \\ 1 \\ 1.3 \end{bmatrix}_{\max}$

2.1.2.2 N-Queen Example

The N-queen problem is a classic MIP problem [4]. In this problem, it is tried to maximize the number of queens that can sit on a chessboard without attacking each other. The procedure is simple as follows: First of all, it is needed to define a variable x_{ij} which is a binary variable (0/1) and states whether the queen should sit (1) on block ij (row i, column j) or not (0). Additionally, if the queen is on block ij then no other queen can sit on row i or column j or the diagonal that contains cell ij. This is mathematically stated as follows:

$$\max_{x_{ij}} \text{OF} = \sum_{i,j} x_{ij} \tag{2.5a}$$

$$\sum_{i} x_{ij} \leq 1 \quad \forall j \tag{2.5b}$$

$$\sum_{j} x_{ij} \leq 1 \quad \forall i \tag{2.5c}$$

$$\sum_{c,r} x_{c,r} \leq 1 \quad \forall i, j \in \left| \frac{i-r}{j-c} \right| = 1 \tag{2.5d}$$

If a queen is on a cell then no other queen can exist on the same column (2.5b) or the same row (2.5c) or the same diagonal (2.5d). The GAMS code for solving N-queen problem (2.5) is given in GCode 2.4:

The N-queen problem is solved two times in Gcode 2.4. Two models are defined in this code namely *MIP2a* and *MIP2b*. The Queen placement is solved on a 4 × 4 board, Fig. 2.1a shows the wrong placement. This solution is obtained from *MIP2a* and considers $eq_{1,2,3}$. This is because we are not considering diagonal movement of queen. In order to overcome this shortcoming two additional constraints $eq_{4,5}$ are considered in Model *MIP2b*. The correct solution is depicted in Fig. 2.1b.

It should be noted that the solution for the defined problem is optimal but not unique. This means that other configurations might be obtained that can satisfy the defined constraints. The developed GCode 2.4 is general and can be used for solving the problem for various sizes of the board. The optimal solution for queen placement on a (a) 8 × 8 chessboard and (b) 16 × 16 board is shown in Fig. 2.2.

GCode 2.4 N-queen example (2.5)

```
Sets i /1*4/, j /1*4/;
alias(i,row);
alias(j,col);
variable of;
binary variable x(i,j);
Equations eq1,eq2,eq3,eq4,eq5;
eq1(j) .. sum(i,x(i,j)) =l=1;
eq2(i) .. sum(j,x(i,j)) =l=1;
eq3    .. sum((i,j),x(i,j))=e=OF;
eq4(i,j) ..  sum((row,col)$((ord(row)-ord(i))=(ord(col)-ord(j))),
     x(row,col))=l=1;
eq5(i,j) ..  sum((row,col)$((ord(row)-ord(i))=-(ord(col)-ord(j)))
     ,x(row,col))=l=1;
Model MIP2a /eq1,eq2,eq3/;
Model MIP2b /all/;
Solve MIP2a US MIP max of;
display x.l,of.l;
Solve MIP2b US MIP max of;
display x.l,of.l;
```

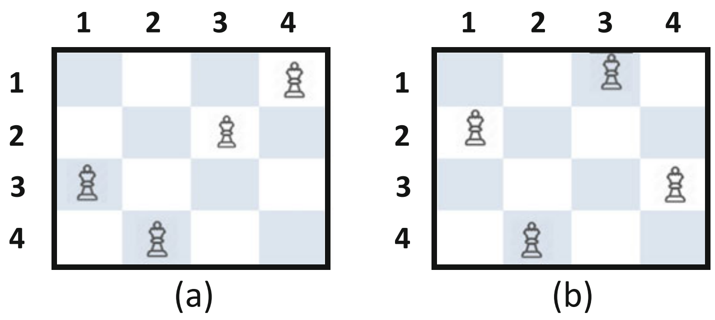


Fig. 2.1 Queen placement on a 4 × 4 board: (**a**) wrong placement, (**b**) optimal placement

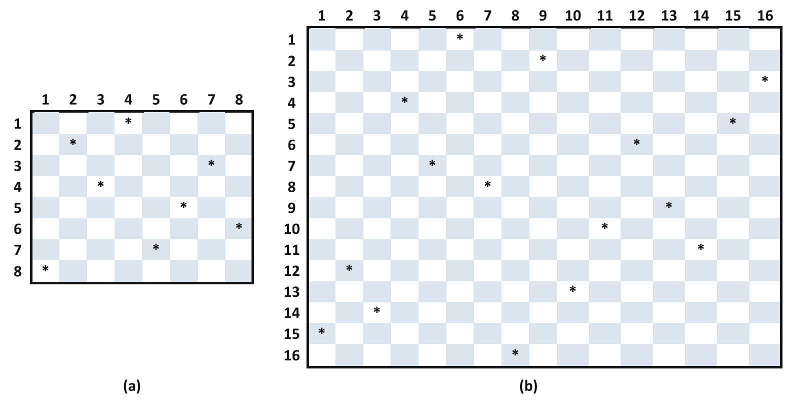


Fig. 2.2 Queen placement on a (**a**) 8 × 8 chessboard, (**b**) 16 × 16 board

2.1.2.3 Emergency Center Allocation

Consider six cities (1–6) which are located at different distances to each other. Each city should have access to an emergency center within a short period of time. The time required for moving from one city to another one is given in the following table in minutes.

	1	2	3	4	5	6
1	0	30	16	22	24	29
2		0	54	32	43	24
3			0	44	50	28
4				0	14	43
5					0	12
6						0

It should be noted that the values of this table are symmetrical. For example, distance from city 1 to city 2 is 30 min. It means that the distance from city 2 to city 1 is also 30 min. The critical time for reaching to the emergency center is assumed to be 20 min. The question is what is the minimum number of cities that should host emergency center? Which cities should be chosen?

By observing the first row of distance matrix, it is understood that if city 1 or city 6 have the emergency center then city 1 meets the access requirement. The same concept applies for city 2. City 2 should definitely have an emergency center because no other city is located within 20 min distance of this city. If the allocation decision is defined as a binary variable x_i then the following inequality should be satisfied for city 1:

$$x_1 + x_6 \geq 1 \tag{2.6}$$

The following inequality should be satisfied for city 2:

$$x_2 \geq 1 \tag{2.7}$$

The overall constraints are as follows:

$$x_1 + x_6 \geq 1 \tag{2.8}$$

$$x_2 \geq 1 \tag{2.9}$$

$$x_3 + x_5 \geq 1 \tag{2.10}$$

$$x_4 + x_5 \geq 1 \tag{2.11}$$

$$x_3 + x_4 + x_5 + x_6 \geq 1 \tag{2.12}$$

$$x_1 + x_5 + x_6 \geq 1 \tag{2.13}$$

Two different approaches will be presented here for modeling this problem.
Scalar equations:

In this approach, we initially analyzed the distance data and understood what kind of relations should be enforced for different variables. The GCode 2.5 is describing how to do this.

GCode 2.5 Scalar equations for emergency centre allocation

```
binary variable x1,x2,x3,x4,x5,x6;
variable OF;
equations
eq1,eq2,eq3,eq4,eq5,eq6,eq7;
eq1 .. x1+x6 =g=1;
eq2 .. x2 =g=1;
eq3 .. x3+x5 =g=1;
eq4 .. x4+x5 =g=1;
eq5 .. x3+x4+x5+x6 =g=1;
eq6 .. x1+x5+x6 =g=1;
eq7 .. x1+x2+x3+x4+x5+x6 =e=OF;
Model emergency /all/;
Solve emergency us mip min of;
```

The optimal answer is OF = 3 (three cities should host emergency centers). The candidate cities are x_1, x_2, x_5. The problem with this kind of modeling is that it needs pre-processing of the raw data and also in case, the number of cities are changed then it is need to extensively modify the code. A much more efficient way of coding this problem is using the extended equations.
Indexed equations:

GCode 2.6 Indexed equations for emergency centre allocation

```
set city /1*6/        ;
alias(city,town);
binary variable x(city);
variable OF;
table data(city,town)
     1     2     3     4     5     6
1    0     30    46    22    24    19
2          0     54    32    43    24
3                0     44    16    28
4                      0     14    43
5                            0     12
6                                  0;
data(city,town)$data(town,city)=data(town,city);
scalar criticaltime /20/;
Equations
eq1,eq2;
eq1(city) .. sum(town$(data(city,town)<criticaltime), x(town)) =g
    =1;
eq2 .. OF=e=sum(city,x(city));
Model emergency /all/;
Solve emergency us mip min of;
```

The developed code will provide the same answer as before but it has the following features:

- It is not needed to manually write one equation for each constraint.
- It works for any number of cities.
- The distance data is fed to the model using a table. This will be useful for cases that the input data might change.
- Debugging and tracing the code are much easier for the users.
- The code does not change if the critical access time is updated.

The following line of the code is to make the data matrix symmetrical.

```
data(city,town)$data(town,city)=data(town,city);
```

The following line describes the condition for accessing the emergency center (for each city). The equation eq1 is defined over the set “city.”

```
eq1(city) .. sum(town$(data(city,town)<criticaltime), x(town)) =g=1;
```

The optimal solution is:

```
—- VAR x
LOWER LEVEL UPPER MARGINAL
1 . . 1.000 1.000
2 . 1.000 1.000 1.000
3 . . 1.000 1.000
4 . . 1.000 1.000
5 . 1.000 1.000 1.000
6 . 1.000 1.000 1.000
```

2.1.3 Nonlinear Programming (NLP)

In nonlinear programming problems, at least one of f, G, H in (2.1) is nonlinear.

2.1.3.1 NLP Example (2.14)

$$\max_{x_i} \text{OF} = x_1x_4(x_1 + x_2 + x_3) + x_2 \tag{2.14a}$$

$$x_1x_2x_3x_4 \geq 20 \tag{2.14b}$$

$$x_1^2 + x_2^2 + x_3^2 + x_4^2 = 30 \tag{2.14c}$$

$$1 \leq x_1, x_2, x_3, x_4 \leq 3 \tag{2.14d}$$

The GAMS code for solving example (2.14) is given in GCode 2.7:

GCode 2.7 Example (2.14)

```
variable of,x1,x2,x3,x4;
equations
eq1,eq2,eq3;
eq1 .. x1*x4*(x1+x2+x3)+x2=e=OF;
eq2 .. x1*x2*x3*x4 =g=20;
eq3 .. x1*x1+x2*x2+x3*x3+x4*x4=e=30;
x1.lo=1;
x1.up=3;
x2.lo=1;
x2.up=3;
x3.lo=1;
x3.up=3;
x4.lo=1;
x4.up=3;
Model NLP1 /all/;
Solve NLP1 US NLP max of;
```

The solution for example (2.14) obtained by GCode 2.7 is as follows:

Variables	Lower	Level	Upper	Marginal
of	$-\infty$	73.605	$+\infty$	0
$x1$	1.000	3.000	3.000	21.025
$x2$	1.000	2.575	3.000	0
$x3$	1.000	2.317	3.000	0
$x4$	1.000	3.000	3.000	12.025

It is worth noting that providing a starting point for the variables can help the GAMS in finding a better solution. Generally speaking, finding the optimal solution of the model is not guaranteed in NLP problems. The initial values for variables are set by "X.l=initial value" command before solve statement.

2.1.3.2 Circle Placement Example (2.15)

Suppose there are n circles with known radius values (R_i). The question is: what is the minimum surface of the table that these circles can be placed on it without any overlapping. The concept of optimal circle placement on a given table is shown in Fig. 2.3. The decision variables of this problem are table dimensions $\{w, h\}$ and center locations $\{x_i, y_i\}$. The objective function is defined as the surface of the table ($\text{OF} = w \times h$). The constraints are described as:

- Each circle should be completely on the table. This requires: $R_i \le x_i \le w - R_i$ and similarly $R_i \le y_i \le h - R_i$.
- For every two circles, no overlap should happen. This means that: $(x_i - x_j)^2 + (y_i - y_j)^2 \ge (R_i + R_j)^2$

The circle placement problem is formulated as:

$$\min_{x_i, y_i, w, h} \text{OF} = hw \tag{2.15a}$$

$$R_i \le x_i \le w - R_i \tag{2.15b}$$

$$R_i \le y_i \le h - R_i \tag{2.15c}$$

$$(x_i - x_j)^2 + (y_i - y_j)^2 \ge (R_i + R_j)^2 \tag{2.15d}$$

It is assumed that R_i are known in advance and they are treated as input data. It is also assumed that there are six circles available. The GAMS code for solving circle placement example (2.15) is as GCode 2.8:

GCode 2.8 Circle placement example (2.15)

```
set i /1*6/; alias(i,j);
Positive variables x(i),y(i),w,h; variable of;
parameter radius(i)
/1 2
2 1.2
3 1.8
4 0.9
5 3.2
6 0.7/;
Equations eq1,eq2,eq3,eq4;
eq1 .. w*h=e=OF;
eq2(i) .. x(i)=l=w-radius(i);
eq3(i) .. y(i)=l=h-radius(i);
eq4(i,j)$(ord(i)<>ord(j))..power(y(j)-y(i),2)+power(x(j)-x(i),2)=
    g=(radius(i)+radius(j),2);
x.lo(i)=radius(i); y.lo(i)=radius(i);
Model NLP1 /all/;
Solve NLP1 US NLP min of;
parameter report(i,*);
report(i,'X')=x.l(i); report(i,'y')=y.l(i);
report(i,'R')=radius(i);
display report,of.l,w.l,h.l;
```

As it is observable in GCode 2.8, the radius of circles are given as parameters. The constraint given in (2.15b) is actually two constraints $R_i \leq x_i$ and $x_i \leq w - R_i$. The first one should be treated using *.lo* statement and the second one should be modeled using equations. This is because it involves two variables x_i, w which should be valid for every member of set i. This is why the equation $eq2$ is defined over set i.

The optimal solution of circle placement on a table is given in Fig. 2.4.

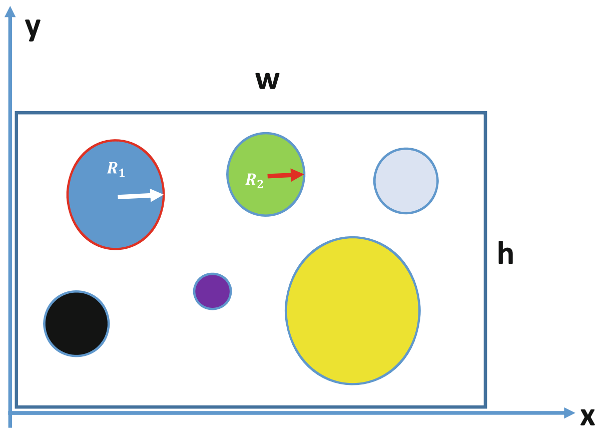
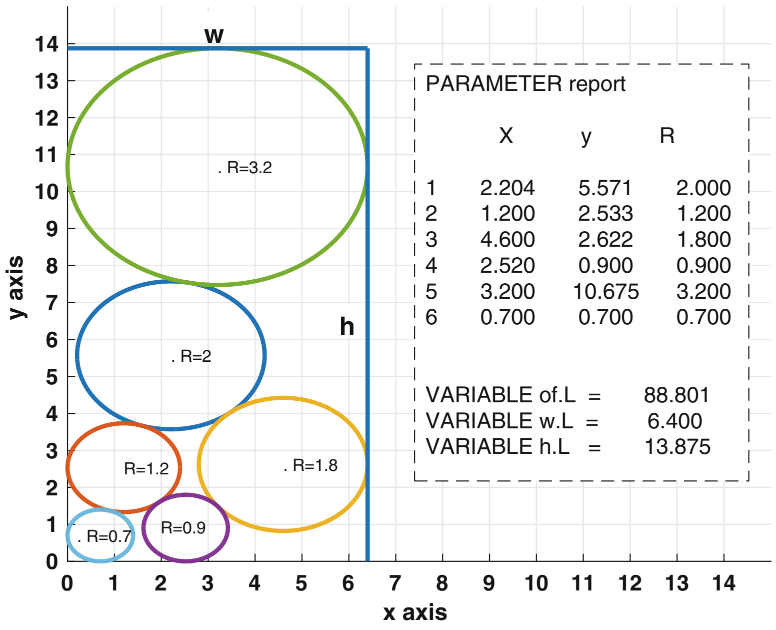


Fig. 2.3 Optimal circle placement on a given surface



```
PARAMETER report

        X        y        R

1     2.204    5.571    2.000
2     1.200    2.533    1.200
3     4.600    2.622    1.800
4     2.520    0.900    0.900
5     3.200   10.675    3.200
6     0.700    0.700    0.700


VARIABLE of.L  =     88.801
VARIABLE w.L   =      6.400
VARIABLE h.L   =     13.875
```

Fig. 2.4 The optimal solution of circle placement on a surface

2.1.3.3 Maximizing the Area of a Right Triangle with Constant Circumference

Suppose that we have some limited meters of wire fences (C) and are requested to enclose an area (right triangle shape). Determine the dimensions of such a triangle. Considering Fig. 2.5, the following optimization should be solved:

$$\max_{H,B} \text{OF} = \frac{H * B}{2} \tag{2.16a}$$

$$H + B + \sqrt{H^2 + B^2} = C \tag{2.16b}$$

where C is the length of the available wire fence. H and B are the height and base of the triangle, respectively. The GCode 2.9 provides the solution for maximizing the area of a triangle with constant circumference.

GCode 2.9 Maximizing the area of a triangle

```
Positive variables h, b;
Variable OF;
Scalar C  /15/; h.up=C; b.up=C;
Equations eq1,eq2;
eq1 .. h+b+sqrt(h*h+b*b)=e=C;
eq2 .. OF=e=0.5*h*b;
Model triangle /all/;
Solve triangle us nlp min of;
```

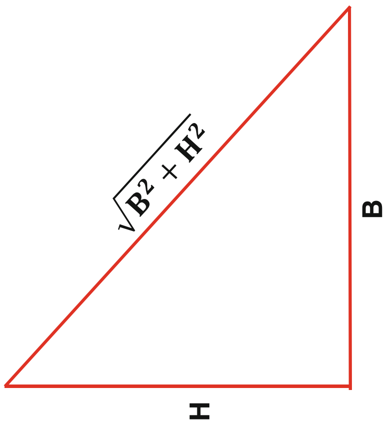


Fig. 2.5 Maximizing the area of a triangle with constant circumference

2.1.4 *Quadratic Constrained Programming (QCP)*

The quadratic programming is a special case of NLP problems. In QCP problems, at least one of f, G, H in (2.1) is nonlinear but nonlinearity is of quadratic form.

2.1.4.1 QCP Example (2.17)

Consider the following QCP optimization problem:

$$\max_{x_i} \mathrm{OF} = -3x_1^2 - 10x_1 + x_2^2 - 3x_2 \tag{2.17a}$$

$$x_1 + x_2^2 \geq 2.5 \tag{2.17b}$$

$$2x_1 + x_2 = 1 \tag{2.17c}$$

The GAMS code for solving example (2.17) is as GCode 2.10:

GCode 2.10 QCP Example (2.17)

```
variable of,x1,x2;
equations
eq1,eq2,eq3;
eq1 .. -3*x1*x1 -10*x1 +x2*x2-3*x2=e=OF;
eq2 .. x1+x2*x2 =g=2.5;
eq3 .. 2*x1+x2=e=1;
Model QCP1 /all/;
Solve QCP1 US QCP max of;
```

The solution for example (2.17) obtained by GCode 2.10 is as follows:

Variables	Lower	Level	Upper	Marginal
of	$-\infty$	-9.550	$+\infty$	0
$x1$	$-\infty$	1.093	$+\infty$	0
$x2$	$-\infty$	-1.186	$+\infty$	0

2.1.4.2 QCP Example (2.18)

Another simple QCP problem is described as follows:

$$\min_{x_i} \text{OF} = x_1^2 - 10x_1 + x_2^2 - 3x_2 \tag{2.18a}$$

$$x_1^2 + x_2 \leq 5 \tag{2.18b}$$

$$2x_1 - x_2 \geq 1 \tag{2.18c}$$

GCode 2.11 Example (2.18)

```
Variable of,x1,x2;
Equations eq1,eq2,eq3;
eq1 .. x1*x1 -10*x1 +x2*x2-3*x2=e=OF;
eq2 .. x1*x1+x2 =l=5;
eq3 .. 2*x1-x2=g=1;
Model QCP1 /all/;
Solve QCP1 US QCP min of;
```

The solution for example (2.18) (obtained by GCode 2.11) is as follows:

Variables	Lower	Level	Upper	Marginal
of	$-\infty$	-18.054	$+\infty$	0
$x1$	$-\infty$	2.054	$+\infty$	0
$x2$	$-\infty$	0.783	$+\infty$	0

2.1.5 Mixed Integer Nonlinear Programming (MINLP)

In MINLP problems, at least one of f, G, H in (2.1) is nonlinear and integer variables are involved.

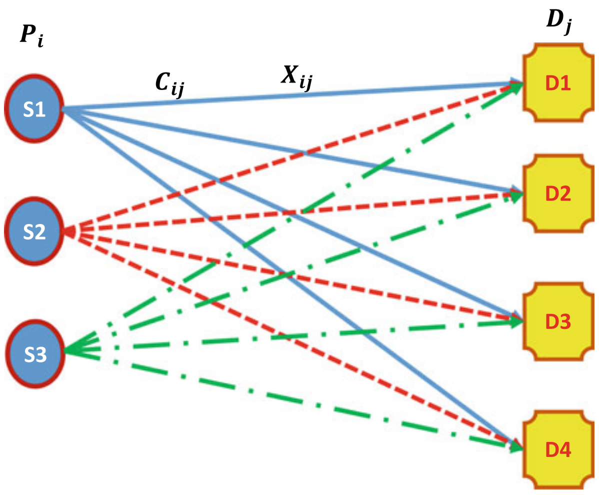


Fig. 2.6 Optimal transportation problem

2.1.5.1 Minimum Transportation Cost Example (2.19)

The transportation problem is a classic example which has been modified for this example. There are some suppliers (node i) and some demands (node j) which should be supplied. The problem is to determine how much each supplier should produce and how to transfer them to the demand points. The transportation costs should be minimized.

The cost of transportation is assumed to be related to the square of quantity transported from node i to node j . The transportation costs are proportional to the square of product transported from i to j and the length of the rout. The object is minimizing the total transportation costs and supplying all demand in different nodes. The optimal transportation problem is depicted in Fig. 2.6. The cost coefficient and maximum flow of each rout (C_{ij}), demand (D_j), and capacity of each producer are known.

GCode 2.12 Example (2.19)

```
sets
i /s1*s3/
j /D1*D4/;
table C(i,j)
    d1          d2          d3          d4
s1  0.0755      0.0655      0.0498      0.0585
s2  0.0276      0.0163      0.096       0.0224
s3  0.068       0.0119      0.034       0.0751;
table data(i,*)
    'Pmin'  'Pmax'
s1   100     450
```

```
s2   50       350
s3   30       500;
parameter demand(j)
/d1 217
d2 150
d3 145
d4 244/;
variable of,x(i,j),P(i);
binary variable U(i);
equations
eq1,eq2,eq3,eq4,eq5;
eq1  ..  OF=e=sum((i,j),C(i,j)*x(i,j)*x(i,j));
eq2(i)  ..    P(i)=l=data(i,'Pmax')*U(i);
eq3(i)  ..    P(i)=g=data(i,'Pmin')*U(i);
eq4(j)  .. sum(i,x(i,j))=g=demand(j);
eq5(i)  .. sum(j,x(i,j))=e=P(i);
P.lo(i)=0;
P.up(i)=data(i,'Pmax');
x.lo(i,j)=0;
x.up(i,j)=100;
Model minlp1 /all/;
Solve minlp1 US minlp min of;
```

The optimal transportation problem is formulated in (2.19).

$$\min_{x_{ij}} \mathrm{OF} = \sum_{ij} C_{ij} x_{ij}^2 \tag{2.19a}$$

$$\begin{cases} P_i^{\min} \le P_i \le P_i^{\max}, & \text{if unit } i \text{ is on,} \\ P_i = 0 & \text{if unit } i \text{ is off} \end{cases} \tag{2.19b}$$

$$\sum_i x_{ij} \ge D_j \tag{2.19c}$$

$$\sum_j x_{ij} = P_i \tag{2.19d}$$

$$0 \le x_{ij} \le x_{ij}^{\max} \tag{2.19e}$$

The road flow limit is assumed to be $x_{ij}^{\max} = 100$.

	$P_i^{\min}$	$P_i^{\max}$	j			
i	100	450	0.0755	0.0655	0.0498	0.0585
	50	350	0.0276	0.0163	0.096	0.0224
	30	500	0.068	0.0119	0.034	0.0751
D_j			217	150	145	244

The GAMS code for solving example (2.19) is as GCode 2.12:
The solution for example (2.19) obtained by GCode 2.12 is as follows:

P_i	X_{ij}			
199.245	55.443	14.255	48.601	80.946
282.494	100	57.282	25.212	100
274.261	61.557	78.463	71.187	63.054

2.1.5.2 Benefit Maximization Transportation Example (2.20)

Reconsider the optimal transportation problem formulated in (2.19). This problem is a minimum cost problem and the goal is service provision to the consumers. Now suppose that the objective is benefit maximization. The machine (i) will produce equal to P_i and sell it to different consumer j. The maximum value of purchase that a consumer may procure is D_j. The revenue that is obtained from selling product is k \$/unit. Now the benefit maximization is modeled as follows:

$$\max_{x_{ij}} \text{OF} = \sum_i P_i k - \sum_{ij} C_{ij} x_{ij}^2 \tag{2.20a}$$

$$\begin{cases} P_i^{\min} \le P_i \le P_i^{\max}, & \text{if unit } i \text{ is on ,} \\ P_i = 0 & \text{if unit } i \text{ is off} \end{cases} \tag{2.20b}$$

$$\sum_i x_{ij} \le D_j \tag{2.20c}$$

$$\sum_j x_{ij} = P_i \tag{2.20d}$$

$$0 \le x_{ij} \le x_{ij}^{\max} \tag{2.20e}$$

The GAMS code for solving the example (2.20) is given in GCode 2.13.
The solution for example (2.20) obtained by GCode 2.13 is as follows:

	P_i	X_{ij}			
i	0	0	0	0	0
	137.377	32.609	55.215	9.375	40.179
	0	0	0	0	0

2.2 Random Numbers in GAMS

Random number generation in GAMS is a simple task. There are different built-in probability density function that can be used for generating random numbers. Some popular random number generators are listed as follows:

- Beta function

$$\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}(1-x)^{\beta-1}(x)^{\alpha-1} \tag{2.21}$$

The format of this function in GAMS is beta(α,β)

- Uniform function (continuous)

$$P(x)=\begin{cases}\frac{1}{(b-a)}, & \text{for } a\le x\le b\\ 0, & \text{otherwise}\end{cases} \tag{2.22}$$

The format of this function in GAMS is uniform(a, b)

- Uniform function (discrete)

$$P(x)=\begin{cases}\frac{1}{N}, & \text{for } a\le x\le b\\ 0, & \text{otherwise}\end{cases} \tag{2.23}$$

GCode 2.13 Example (2.20)

```
sets
i /s1*s3/
j /D1*D4/;
scalar k /1.8/;
table C(i,j)
    d1          d2          d3          d4
s1  0.0755      0.0655      0.0498      0.0585
s2  0.0276      0.0163      0.096       0.0224
s3  0.068       0.0119      0.034       0.0751;
table data(i,*)
    'Pmin'  'Pmax'
s1   100     450
s2   50      350
s3   30      500;
parameter demand(j)
/d1 217
d2 150
d3 145
d4 244/;
variable of,x(i,j),P(i);
binary variable U(i);
```

```
equations
eq1 ,eq2 ,eq3 ,eq4 ,eq5 ;
eq1 .. OF=e=sum(i ,k*P(i))−sum((i ,j) ,C(i ,j)*x(i ,j)*x(i ,j));
eq2(i) ..    P(i)=l=data(i ,'Pmax')*U(i);
eq3(i) ..    P(i)=g=data(i ,'Pmin')*U(i);
eq4(j) .. sum(i ,x(i ,j))=l=demand(j);
eq5(i) .. sum(j ,x(i ,j))=e=P(i);
P.lo(i)=0;
P.up(i)=data(i ,'Pmax');
x.lo(i ,j)=0;
x.up(i ,j)=100;
Model minlp2 /all/;
Solve minlp2 US minlp max of;
```

where N is the total integer numbers in $[a, b]$ interval. The format of this function in GAMS is uniformint(a, b)

2.2.1 Estimating the π Number

Calculating the π number is investigated and coded in GAMS. Consider a circle inscribed in a square as shown in Fig. 2.7. If a point is randomly dropped on this square then the probability of sitting on the circle would be $\frac{\text{Circle area}}{\text{Square area}}$. In order to calculate the π number a simple experience is done. The point is dropped on the square area for N times. The number of events that the point is on circle area would be n. The following equation would be valid if N is a big number. The GAMS code for solving this problem is described in GCode 2.14.

$$\frac{\text{Circle area}}{\text{Square area}} = \frac{n}{N} \tag{2.24}$$

$$\frac{\pi R^2}{4R^2} = \frac{n}{N} \tag{2.25}$$

GCode 2.14 π number estimation

```
scalar low /0/, High /1/,pistimate;
set counter /c1*c200/;
parameter report(counter,*);
report(counter,'x')=uniform(LOW,HIGH);
report(counter,'y')=uniform(LOW,HIGH);
pistimate=4*sum(counter$(power(report(counter,'x')
−0.5,2)+power(report(counter,'y')−0.5,2)<=0.25),1)/card(counter);
display report , pistimate;
```

As it can be seen in GCode 2.14, there is no solve statement needed. This is because no variable is defined in the model and no optimization is going to take place.

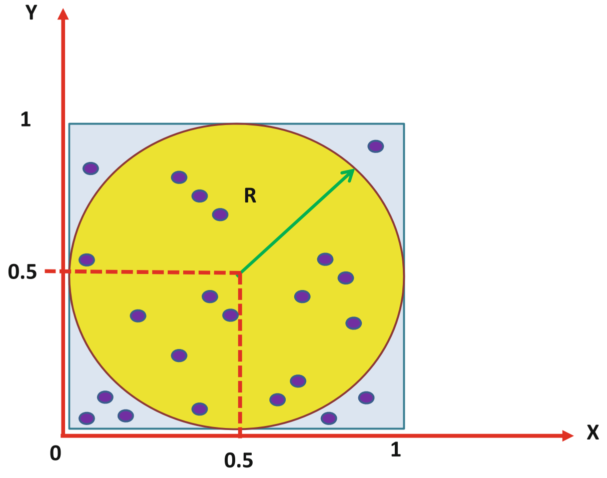


Fig. 2.7 Estimating the π number

2.2.2 Integration Calculation

The random numbers can be used for calculating the integration problems. A simple example is given as follows:

$$Z = \int_0^1 1 - x\sin(20x)dx \tag{2.26}$$

In order to calculate the area under the graph as it is shown in Fig. 2.8, the following steps are followed:

- Set Counter = 1; $n = 1$;
- Generate a pair of random numbers (x, y) where $0 \le X \le 1$ and $0 \le Y \le 2$.
- Check if $y \le f(X)$
- Set Counter = Counter + 1;

This procedure is repeated for max number of counter (N). For large values of N, the Z can be calculated using the following relation:

$$\frac{Z}{2 * 1} = \frac{n}{N} \tag{2.27}$$

The following command should be added to the code (before calling the uniform function) in order to have a set of new random numbers every time the code is run:

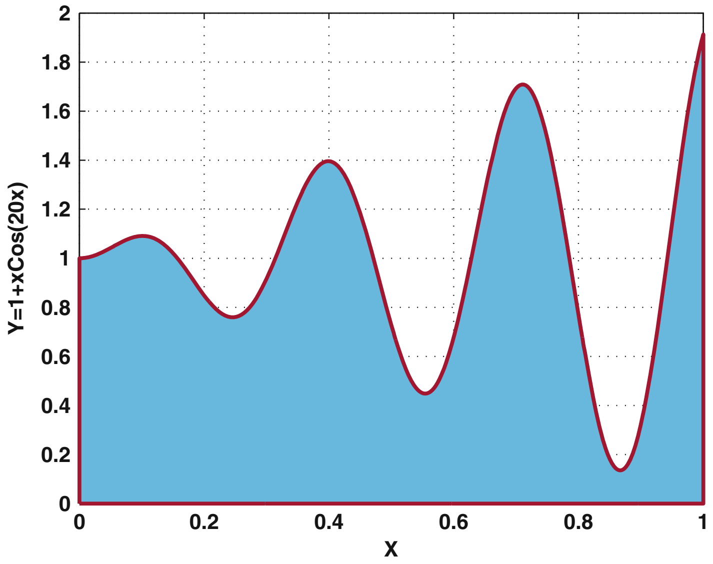


Fig. 2.8 Numerical integration using random numbers

GCode 2.15 Integration calculation using random numbers

```
Scalar Zstimate;
Set counter /c1*c200000/;
parameter report(counter,*);
report(counter,'x')=uniform(0,1);
report(counter,'y')=uniform(0,2);
Zstimate=2*sum(counter$(report(counter,'y')<1+
report(counter,'x')*sin(report(counter,'x')*20)),1)/card(counter);
Sisplay report , Zstimate;
```

```
execseed = 1+gmillisec(jnow);
```

Further info on how to use random numbers in GAMS can be found in [5].

2.2.3 LP Problems with Uncertain Coefficients

Consider the following LP problem:

$$Z = \max_{x_1,x_2} 750x_1 + 1000x_2 \tag{2.28}$$

$$x_1 + x_2 \leq a \tag{2.29}$$

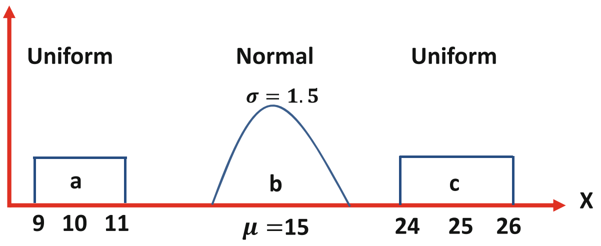


Fig. 2.9 The probability density function for each uncertain parameter

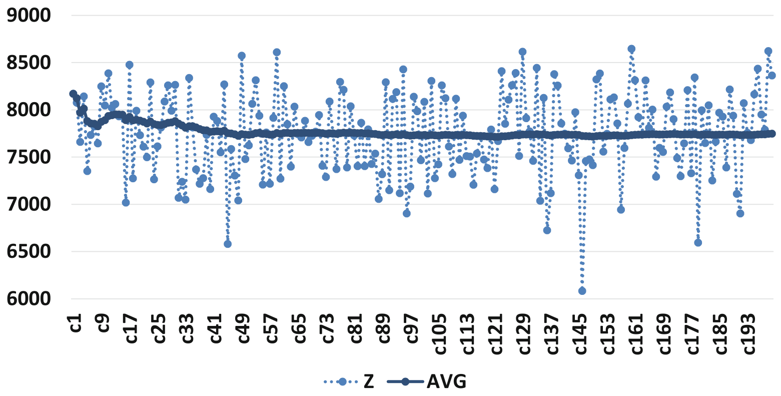


Fig. 2.10 Uncertain coefficients in LP problems

$$x_1 + 2x_2 \leq b \tag{2.30}$$

$$4x_1 + 3x_2 \leq c \tag{2.31}$$

$$x_1 \geq 0 \tag{2.32}$$

$$x_2 \geq 0 \tag{2.33}$$

where a, b, and c are uncertain random parameters. The probability density function for each parameter is given in Fig. 2.9. The question is how to describe the probability density of Z? If all uncertain parameters (a,b,c) are equal to their expected values (10,15,25) then the objective function would be 7750. The following steps are followed to find out the distribution of Z in case of uncertain (a,b,c):

- Set Counter = 1;
- Generate a sample of random parameters (a,b,c) using the specified probability distribution functions
- Calculate the optimal Z and save the (a,b,c,x_1,x_2,Z).

The variation of Z along with the average value of Z are plotted in Fig. 2.10. The graph shows how the average value of Z converges.

2.3 Multi-Objective Optimization

In Multi-Objective Decision Making (MODM) methods, the decision alternatives are found considering the constraints of the given problem. In most practical optimization problems, particularly those applicable in power system, there exists more than one objective function which should be optimized simultaneously. These objectives functions might be in conflict, interdependent or independent of each other, so it is impossible to satisfy them all at once. The main differences between the multi-objective optimization and traditional single optimization techniques can be categorized into two groups:

1. Several objective functions are to be optimized at the same time.
2. There exists a set of optimal solutions which are mathematically equally good solutions (it means any of them cannot be preferred against others and a trade-off should be made to select one) instead of a unique optimal solution

2.3.1 *Weighted Sum Approach*

Some methods try to convert the multi-objective optimization problems into single objective one. However, this approach is not always applicable especially in the following cases:

- The objectives are not of the same type. For example voltage deviation and cost, weight and volume, surface and time.
- The weight coefficients in weighted sum approach cannot be easily determined. Additionally, If decision maker's preference is changed then the problem should be solved again.
- The single objective approach provides just one solution to the decision maker. It cannot show the tradeoff between two objective functions.
- The objective function is of the same type but there are multiple decision makers with different preferences. Each decision maker is trying to optimize its own objective function.

2.3.2 *Pareto Optimality*

The notion of optimality has been redefined in this context and instead of aiming to find a single solution, it is tried to produce a set of acceptable compromises or trade-offs from which the decision maker can choose one. The set of all optimal solutions which are non-dominated by any other solution is known as Pareto-optimal set. Suppose a minimizing problem with two objectives in conflict the Pareto optimal fronts are plotted in Fig. 2.11.

For every two solution in each Pareto front (like A, B) none of them is better than the others considering all objective functions. Here, f_1 is better minimized in solution A compared to B and f_2 is better minimized in solution B compared to A.

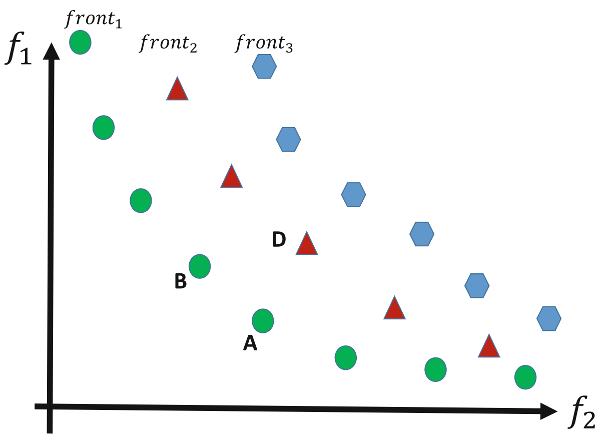


Fig. 2.11 Classification of a population to *k* non-dominated fronts

The same concept also applies for solutions in other fronts. For every solution in Pareto front *k* (for example D in the second front) there exists at least one solution in front $k - 1$ (here A in the first front) that dominates it (is better considering all objective functions). Since solutions A and B belong to the first front, there is no solution better than them in respect to all objectives. Consider the bi-objective optimization described in (2.34):

$$\min_x f_1, f_2 \tag{2.34}$$

$$\text{Constraints}$$

Each solution in Pareto optimal set has two basic characteristics:

1. For every two solutions belonging to the same Pareto front (2.35) holds:

$$\forall i \exists j, n | f_n(\bar{x}_i) > f_n(\bar{x}_j) \tag{2.35}$$
$$\bar{x}_j, \bar{x}_i \in S$$

   This means that for every solution X_i belonging to Pareto front S, at least one solution exists as X_j which is better than X_i at least in one objective function (named n here). In other words, there is no solution in Pareto optimal front which is the best among all members of this set considering all objectives.
2. For every solution belonging to an upper Pareto front and the ones in the lower fronts, (2.36) holds:

$$\forall k \in \{1 \ldots N_O\} f_k(\bar{x}_1) \le f_k(\bar{x}_2) \tag{2.36}$$
$$\exists k' \in \{1 \ldots N_O\} f_{k'}(\bar{x}_1) < f_{k'}(\bar{x}_2) \tag{2.37}$$
$$\bar{x}_1 \in S, \bar{x}_2 \in S^*$$
$$S < S^*$$

This means for every solution belonging to an upper Pareto front, there exists at least one solution in a lower Pareto front which is not worse in any objective function and is better in at least one objective function. N_O is the number of objective functions.

The classic approach for finding the Pareto optimal set is preference-based method in which a relative preference vector is used to weight the objectives and change them into a scalar value. By converting a multi-objective optimization problem into a single objective one, only one optimum solution can be achieved which is very sensitive to the given weights. The GAMS structure can solve only one objective function at once. This means it is needed to solve the multi-objective problem several times to obtain the Pareto optimal front. For this purpose, consider the bi-objective problem described in (2.38):

$$\max_{x_{1,2}} f_1 = 4x_1 - 0.5x_2^2 \tag{2.38a}$$

$$\max_{x_{1,2}} f_2 = -x_1^2 + 5x_2 \tag{2.38b}$$

$$2x_1 + 3x_2 \leq 10 \tag{2.38c}$$

$$2x_1 - x_2 \geq 0 \tag{2.38d}$$

$$1 \leq x_1 \leq 2 \tag{2.38e}$$

$$1 \leq x_1 \leq 3 \tag{2.38f}$$

GCode 2.16 Pareto optimal front example (2.38)

```
set counter /c1*c21/;
scalar E;
parameter report(counter,*);
variable of1,of2,x1,x2;
equations
eq1,eq2,eq3,eq4;
eq1 .. of1=e=4*x1-0.5*x2*x2;
eq2 .. of2=e=-x1*x1+5*x2;
eq3 .. 2*x1+3*x2=l=10;
eq4 .. 2*x1-x2=g=0;
x1.lo=1;
x1.up=2;
x2.lo=1;
x2.up=3;
Model pareto1 /all/;
parameter ranges(*);
Solve pareto1 US nLP max of1;
ranges('OF1max')=of1.l;
ranges('OF2min')=of2.l;
Solve pareto1 US nLP max of2;
ranges('OF2max')=of2.l;
ranges('OF1min')=of1.l;
```

```
E=ranges('OF1min');
loop(counter,
E=(ranges('OF2max')-ranges('OF2min'))*(ord(counter)-1)/(card
    (counter)-1)+ranges('OF2min');
of2.lo=E;
Solve pareto1 US nLP max of1;
report(counter,'OF1')=OF1.l;
report(counter,'OF2')=OF2.l;
report(counter,'E')=E;
);
Display report;
```

In this problem, two objectives should be maximized simultaneously. The procedure is as follows:

1. Find the maximum of each objective function and save them.
2. Add one of the objective functions to the constraints as follows:

$$f_2 \geq \epsilon \tag{2.39}$$

The ϵ value will be varied from $f_2^{\min}$ to $f_2^{\max}$ and the f_1 is maximized.
The solution for example (2.38) obtained by GCode 2.16 is shown in Fig. 2.12.

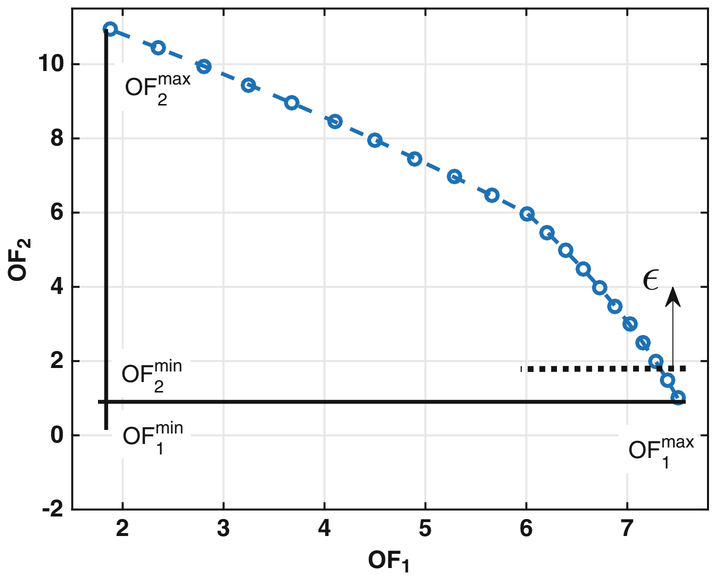


Fig. 2.12 The Pareto optimal front for bi-objective problem

2.4 Applications

Some optimization models used in power system studies are given in this section:

- LP programming: Transmission network estimation [6], short-term hydro scheduling [7], relay coordination [8], security constrained economic dispatch [9].
- MIP: Optimal PMU placement [10], unit commitment [11], Phase shifter placement in large-scale systems [12], minimum-losses radial configuration of electrical distribution networks [13].
- QCP: Topology identification in distribution network [14], optimum active and reactive generation dispatch [15].
- NLP: Voltage stability security margin calculation [16], reactive power planning [17], OPF [18].
- MINLP: Unit commitment with AC OPF constraints [19], flexible transmission expansion planning with uncertainties in an electricity market [20], optimal DG allocation [21].
- Multi-objective optimization: Transmission expansion planning [22], generation expansion planning [23], distribution network planning [24].

References

1. A. Meeraus, A. Brooke, D. Kendrick, R. Raman, *GAMS/Cplex 7.0 User Notes* (GAMS Development Corporation, Washington, DC, 2000)
2. Y. Wang, Q. Xia, C. Kang, Unit commitment with volatile node injections by using interval optimization. IEEE Trans. Power Syst. **26**(3), 1705–1713 (2011)
3. A. Soroudi, Possibilistic-scenario model for DG impact assessment on distribution networks in an uncertain environment. IEEE Trans. Power Syst. **27**(3), 1283–1293 (2012)
4. X. Hu, R.C. Eberhart, Y. Shi, Swarm intelligence for permutation optimization: a case study of n-queens problem, in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE* (IEEE, Piscataway, 2003), pp. 243–246
5. E. Kalvelagen, Some notes on random number generation with GAMS, 2005
6. L.L. Garver, Transmission network estimation using linear programming. IEEE Trans. Power Apparatus Syst. **PAS-89**(7), 1688–1697 (1970)
7. G.W. Chang, M. Aganagic, J.G. Waight, J. Medina, T. Burton, S. Reeves, M. Christoforidis, Experiences with mixed integer linear programming based approaches on short-term hydro scheduling. IEEE Trans. Power Syst. **16**(4), 743–749 (2001)
8. B. Chattopadhyay, M.S. Sachdev, T.S. Sidhu, An on-line relay coordination algorithm for adaptive protection using linear programming technique. IEEE Trans. Power Delivery **11**(1), 165–173 (1996)
9. R.A. Jabr, A.H. Coonick, B.J. Cory, A homogeneous linear programming algorithm for the security constrained economic dispatch problem. IEEE Trans. Power Syst. **15**(3), 930–936 (2000)
10. B. Gou, Generalized integer linear programming formulation for optimal PMU placement. IEEE Trans. Power Syst. **23**(3), 1099–1104 (2008)
11. J. Ostrowski, M.F. Anjos, A. Vannelli, Tight mixed integer linear programming formulations for the unit commitment problem. IEEE Trans. Power Syst. **27**(1), 39–46 (2012)

12. F.G.M. Lima, F.D. Galiana, I. Kockar, J. Munoz, Phase shifter placement in large-scale systems via mixed integer linear programming. IEEE Trans. Power Syst. **18**(3), 1029–1034 (2003)
13. A. Borghetti, A mixed-integer linear programming approach for the computation of the minimum-losses radial configuration of electrical distribution networks. IEEE Trans. Power Syst. **27**(3), 1264–1273 (2012)
14. Z. Tian, W. Wu, B. Zhang, A mixed integer quadratic programming model for topology identification in distribution network. IEEE Trans. Power Syst. **31**(1), 823–824 (2016)
15. H. Nicholson, M.J.H. Sterling, Optimum dispatch of active and reactive generation by quadratic programming. IEEE Trans. Power Apparatus Syst. **PAS-92**(2), 644–654 (1973)
16. L.A.L. Zarate, C.A. Castro, J.L.M. Ramos, E.R. Ramos, Fast computation of voltage stability security margins using nonlinear programming techniques. IEEE Trans. Power Syst. **21**(1), 19–27 (2006)
17. L.L. Lai, J.T. Ma, Application of evolutionary programming to reactive power planning-comparison with nonlinear programming approach. IEEE Trans. Power Syst. **12**(1), 198–206 (1997)
18. J.A. Momoh, R. Adapa, M.E. El-Hawary, A review of selected optimal power flow literature to 1993. I. nonlinear and quadratic programming approaches. IEEE Trans. Power Syst. **14**(1), 96–104 (1999)
19. A. Castillo, C. Laird, C.A. Silva-Monroy, J.P. Watson, R.P. ONeill, The unit commitment problem with ac optimal power flow constraints. IEEE Trans. Power Syst. **31**(6), 4853–4866 (2016)
20. J.H. Zhao, Z.Y. Dong, P. Lindsay, K.P. Wong, Flexible transmission expansion planning with uncertainties in an electricity market. IEEE Trans. Power Syst. **24**(1), 479–488 (2009)
21. A. Kumar, W. Gao, Optimal distributed generation location using mixed integer non-linear programming in hybrid electricity markets. IET Gener. Transm. Distrib. **4**(2), 281–298 (2010)
22. P. Maghouli, S.H. Hosseini, M.O. Buygi, M. Shahidehpour, A scenario-based multi-objective model for multi-stage transmission expansion planning. IEEE Trans. Power Syst. **26**(1), 470–478 (2011)
23. S. Kannan, S. Baskar, J.D. McCalley, P. Murugan, Application of NSGA-II algorithm to generation expansion planning. IEEE Trans. Power Syst. **24**(1), 454–461 (2009)
24. V. Miranda, J.V. Ranito, L.M. Proenca. Genetic algorithms in optimal multistage distribution network planning. IEEE Trans. Power Syst. **9**(4), 1927–1933 (1994)