Chapter 1
Introduction to Programming in GAMS

The General Algebraic Modeling System (GAMS) is a modeling tool for mathematical programming and optimization purpose. This chapter provides the instruction on different programming elements in GAMS. It can be used in solving different types of optimization problems. Some basic optimization models used in power system literature are described in this chapter.

1.1 Optimization Problems in Power System

The power system optimization problems are broadly categorized as operation and planning problems. The operation problems are usually related to how to exploit the existing devices/power plants. For example, optimal power flow is an operation problem. The planning problems usually refer to those problems which investigate whether to invest or not in some assets. For example, the decisions regarding the transmission network expansion belong to planning category. Some of these problems are shown in Fig. 1.1. Some power system planning problems are listed as follows:

- Generation expansion planning (GEP) [1, 2]: In GEP, the decision maker is trying to find out the investment decision regarding the generation technology, size, and time of investment.
- Transmission expansion planning (TEP) [3, 4]. In TEP, the decision maker is trying to find out the investment decision regarding the planning option and time of investment. The planning options include but not limited to: building new lines, reconductoring the existing lines, building, or reinforcing the substations.
- Distribution network planning [5]: this problem is trying to make smart investments in new feeders/substations to meet the technical constraints.

A. Soroudi, *Power System Optimization Modeling in GAMS*,
DOI 10.1007/978-3-319-62350-4_1

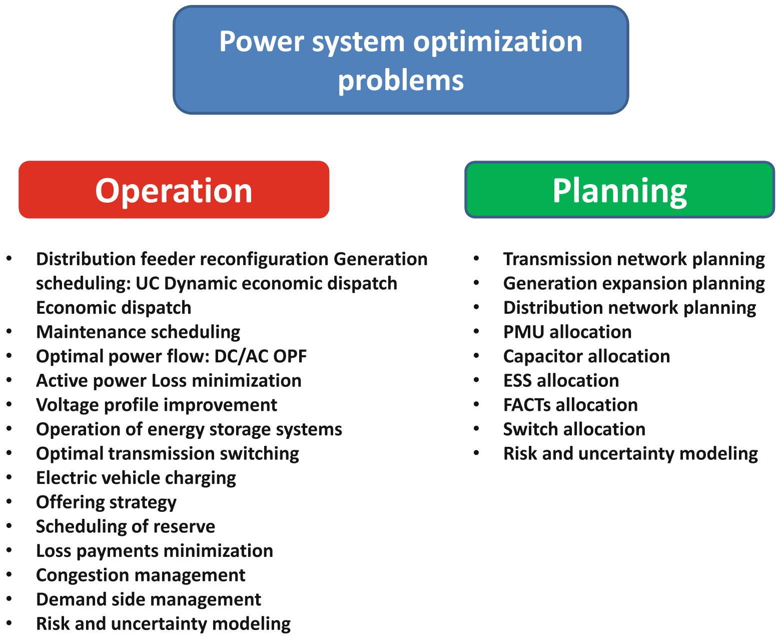

Power system optimization problems
Operation
Distribution feeder reconfiguration Generation scheduling: UC Dynamic economic dispatch Economic dispatch
Maintenance scheduling
Optimal power flow: DC/AC OPF
Active power Loss minimization
Voltage profile improvement
Operation of energy storage systems
Optimal transmission switching
Electric vehicle charging
Offering strategy
Scheduling of reserve
Loss payments minimization
Congestion management
Demand side management
Risk and uncertainty modeling
Planning
Transmission network planning
Generation expansion planning
Distribution network planning
PMU allocation
Capacitor allocation
ESS allocation
FACTs allocation
Switch allocation
Risk and uncertainty modeling

Fig. 1.1 Some optimization problems in power system studies

- FACTS device allocation [6]: This problem is a subset of transmission planning problem. The FACTS devices can control the power flowing through a line and make the transmission network more flexible.
- Distributed generation (DG) allocation in distribution networks [7–9]: The optimal location (or size) of DG unit in a distribution network.
- Capacitor allocation in distribution networks [10]: The capacitor allocation is usually done at the distribution level. The purpose is usually improving the voltage profile in the network.
- PMU allocation [11, 12]: The system observability is improved using optimal placement of phasor measurement units in the network.
- Energy Storage System (ESS) allocation [13, 14]: The ESS can deliver value to customers. This highly depends on how they are operated and located in the system. This problem tries to maximize its benefits by finding the optimal connection point of ESS to the grid.
- Risk and uncertainty modeling in planning studies: the decision-making process highly relies on accuracy of input data. As a matter of fact, the input data for any decision-making problem (especially in practical problems) are subject to uncertainty. If these uncertainties are not treated and handled properly, then costly consequences might occur. Based on the data availability some of the following techniques are applicable:
  - Information gap decision theory (IGDT): ESS allocation [15]
  - Scenario-based uncertainty modeling: DG planning [16]

  - Monte Carlo simulation: FACTS allocation [17]
  - Robust optimization: Transmission expansion planning [18]
  - Fuzzy modeling: distribution network planning [19]

Some power system operation problems are listed as follows:

- Distribution feeder reconfiguration [20]: the on/off statuses of switches are optimally determined to change the network configuration. In distribution networks, it can reduce the active losses by improving the voltage profile and reliability.
- Generation scheduling: UC [21, 22], Dynamic economic dispatch [23], Economic dispatch [24]: the generation level of power plants are determined to minimize the operating costs or maximizing the economic benefits.
- Maintenance scheduling [25]: The maintenance period of assets are determined to keep the adequacy level of the remaining system sufficient while minimizing the costs.
- Optimal power flow: DC OPF [26] and AC OPF [27] are two forms of OPF. In an OPF problem, the decision variables are generation and voltage level of generating units to minimize the operating costs. In DC OPF, it is assumed that all voltage values are 1 pu.
- Active power Loss minimization: minimizing the active losses is a way of improving the efficiency of power system [28].
- Voltage profile improvement [29]: keeping the voltage magnitudes within the normal operating limits is done by changing different decision variables such as reactive power management and demand response.
- Optimal transmission switching [30]: transmission network switching is a technique for changing network topology at the transmission level. It is demonstrated that this can lead to operating cost reduction if it is optimally managed.
- Electric vehicle charging [31]: the charging and discharging of electric vehicles are optimally determined to provide some flexibilities for distribution network operator.
- Offering strategy [32, 33]: the generating units submit their offers to the market operator. The purpose is to maximize the financial benefits.
- Scheduling of reserve [34, 35]: finding the optimal reserve quantity makes the system robust against contingencies and disturbances.
- Loss payments minimization [36]: this approach tries to minimize the payments toward the losses, not the losses by considering the market issues.
- Congestion management [37, 38]: network congestion would reduce the competition level at electricity market. Reducing the congestion would improve the market efficiency.
- Demand side management [39, 40]: the demand side management is harvesting the flexibility from the demand side. The customers are encouraged to shift/reduce their demands to minimize the system requirements for providing services.
- Risk and uncertainty modeling in operation studies:
  - Information gap decision theory (IGDT): Wind operation modeling in OPF [41], unit commitment [42]
  - Scenario-based uncertainty modeling [43]

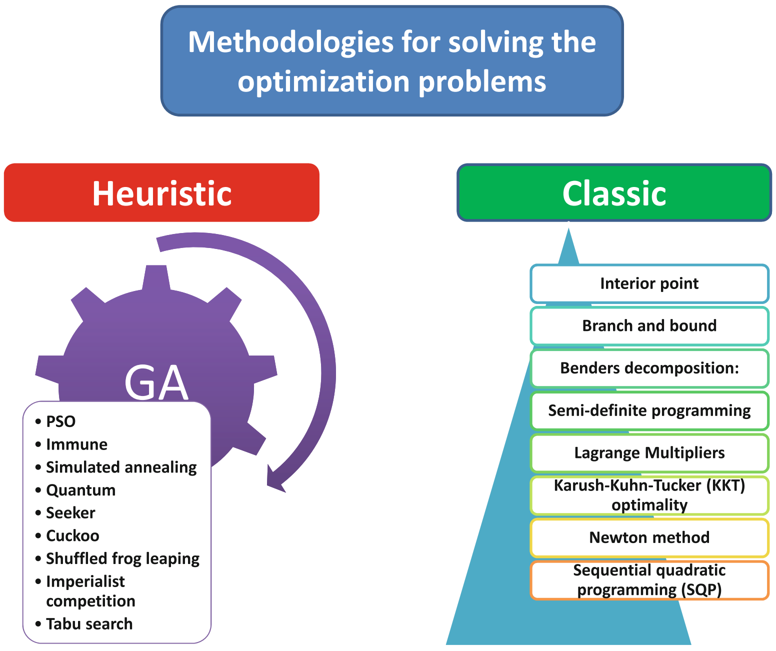


Fig. 1.2 Optimization methods in power system studies

– Monte Carlo simulation: reliability and risk analysis [44]
– Robust optimization: Demand response [40], loss payments minimization [36]
– Fuzzy modeling: DG impact assessment [45]

The methods used for solving the aforementioned optimization problems are categorized into classic and heuristic methods as shown in Fig. 1.2.

Some of the classic methods are listed as follows:

- Interior point: optimal reactive dispatch [46]
- Branch and bound: economic dispatch with disjoint prohibited zones considering network losses [47]
- Benders decomposition: transmission network design problems [48]
- Semi-definite programming: large scale OPF [49, 50]
- Lagrange Multipliers: pricing energy and ancillary services [51]
- Karush-Kuhn-Tucker (KKT) optimality condition: formulation of the terrorist threat problem [52]
- Newton method: Optimal power flow [53, 54]
- Sequential quadratic programming (SQP): UC [55], VAr compensation [56]

Most of the classic methods are gradient-based techniques (in nonlinear problems). This makes them unsuitable for large scale optimization problems. Solving the

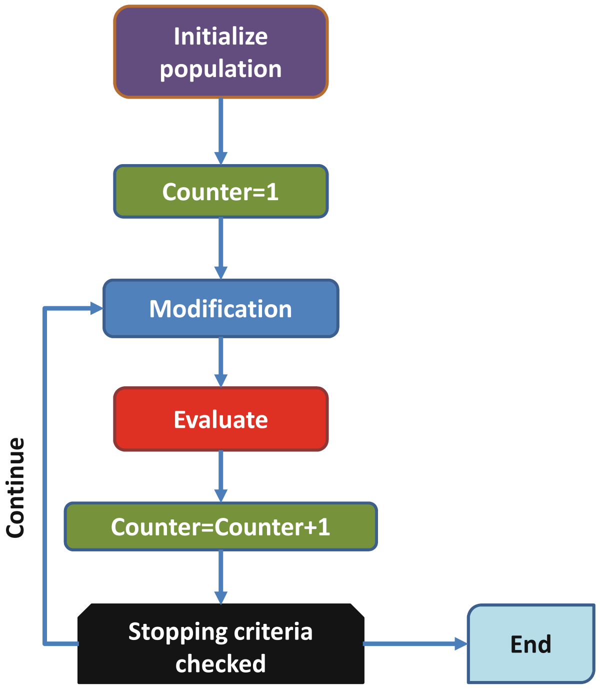


Fig. 1.3 The structure of heuristic optimization techniques

nonlinear problems with integer variables or non-convex constraints would be another challenge for classic techniques.

The heuristic methods are inspired by nature. The basic concept of these techniques is described in Fig. 1.3. In every heuristic method, an initial random population is generated, and it is tried to improve it by using some operator that modify the population (solution). The way each method modifies the population distinguishes that method from the other techniques. For example, the genetic algorithm uses the crossover and mutation operators to improve the solution regarding optimizing the objective function while satisfying the constraints. The particle swarm optimization uses the best solution found by the group and the individual particle to find the optimal solutions. The heuristic methods are also called iterative techniques. The number of iterations needed for optimizing the objective function has an inverse relation with the population number. If the population number is increased, then it can better explore the solution space; however, it takes more time to run. There is always a tradeoff between these two quantities. There are some challenges associated with heuristic techniques. Some of them are listed as follows:

- The parameter tuning is a challenge in these techniques which is usually problem dependent and should be tuned by the decision maker.
- It is not easy to check if the obtained solution is globally optimal or not.

- These methods are usually computationally expensive. In other words, it takes a long time for the decision maker to run. This makes these methods inconvenient for real-time applications.
- Since these methods are iterative, at every iteration, a new solution might be found (which is somehow better than the solution found in previous iterations). Unfortunately, if the problem is solved again (even with the same tuning parameters), it is not guaranteed to reach the similar results.
- Setting the stopping criteria is difficult. This is because the optimal solution of the problem in unknown. When to stop? It is a challenging question to answer. The decision maker usually ends the procedure when there is no significant change in objective function to a maximum number of iterations so reached.
- These solutions are not generally well accepted by the industry.

Some of the heuristic methods applied to power system studies are given as follows:

- Single objective genetic algorithm (GA): network reconfiguration [57]
- Multi-objective Non-dominated sorting genetic algorithm (NSGA): Transmission expansion planning [58]
- Particle swarm optimization (PSO): DG planning [16]
- Immune algorithm (IA): secondary voltage control [59]
- Simulated annealing (SA): maintenance scheduling [60]
- Quantum-inspired evolutionary algorithm: Real and Reactive Power Dispatch [61]
- Seeker Optimization Algorithm: Reactive Power Dispatch [62], coordination of directional over-current relays [63]
- Cuckoo search algorithm: Non-convex economic dispatch [64], capacitor allocation [65]
- Shuffled frog leaping algorithm: unit commitment [66]
- Imperialist competition algorithm: dynamic economic power dispatch [67]
- Tabu search: Economic dispatch [68]
- Ant colony algorithm: Reconfiguration and capacitor placement for loss reduction of distribution systems [69]

1.2 GAMS Installation

The first step is downloading the appropriate installation file from the following address:

https://www.gams.com/download/

The GAMS installation package is available for the following platforms:

Platform	Description
MS Windows 32 bit	Windows Vista or newer on AMD- or Intel-based (x86) architectures.
MS Windows 64 bit	Windows Vista or newer on AMD- or Intel-based (x64) architectures.
Linux 64 bit	AMD- or Intel-based 64-bit (x64) Linux systems with glibc 2.7 or higher.
MacOS X	Intel-based 64-bit (x64) Macintosh system with MacOS X 10.10 (Yosemite) or higher.
Solaris i86pc	AMD- or Intel-based 64-bit (x64) Solaris system. Built on Solaris 11.0.
Solaris SPARC 64bit	Sparc-based 64-bit (sparc-64) Solaris system. Built on Solaris 10.
IBM AIX	PowerPC based 64-bit (ppc-64) AIX system. Built on AIX 6.1

The GAMS interface is depicted in Fig. 1.4.

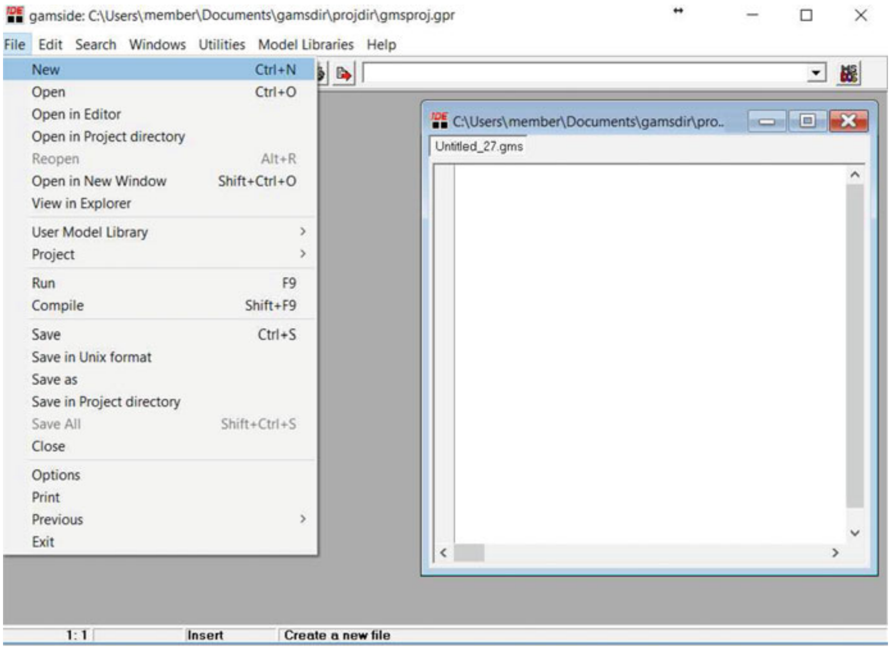


Fig. 1.4 The GAMS interface

1.3 GAMS Elements

Each GAMS model consists of the following main elements:

- Sets: sets are used to define the indices in the algebraic representations of models. For example, set of generating units, set of network buses, set of slack buses, set of time periods, etc.
- Data: The input data of each GAMS model are expressed in the form of Parameters, Tables, or Scalars. The parameters and tables are defined over the sets. The scalars are single value quantities.
- Variables: The variables are decision sets and are unknown before solving the model.
- Equations: The equations describe the relations between the data and variables.
- Model and Solve Statements: The model is defined as a set of equations which contain an objective function. The solve statement asks GAMS to solve the model.
- Output: There are several ways to see the outputs of the solved model such as saving them in XLS file and displaying them.

The General GAMS code structure and elements are shown in Fig. 1.5.

The GAMS elements are explained in a simple example as follows: Suppose a factory produces tree types of products $P_{1,2,3}$. Each product should be processed on two different machines. The available machine hours per day and the time required

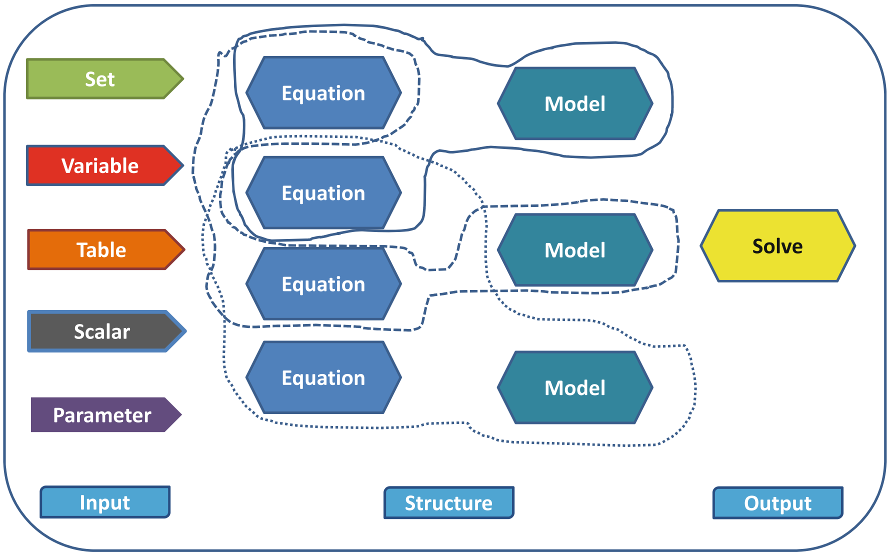


Fig. 1.5 General GAMS code structure and elements

Table 1.1 Data for illustrative example

Required time for task completion (h)			
Machine	P_1	P_2	P_3
M_1	2	5	2
M_2	3	4	1
Profit per kg ($/kg)		Machine availability (h)	
P_1	10	M_1	16
P_2	12	M_2	12
P_3	13.5		

for each product are considered as the input data. The profits/kg of each item is also known. The input data of this example is described in Table 1.1.

- Sets: Two sets should be defined: machines (M) and products (P) Sets $M/m1 * m2/, P/p1 * p3/;$
- Data: Looking at Table 1.1 shows that we have a table (required time for task completion) and two parameters (Profit per item and Machine availability). The task completion table is defined over two sets (machines and products), the profit per item parameter is defined over products while the machine availability is defined over product set. It is assumed that the minimum required amount of each product is 1 kg.

```
Parameter profit(P)
/p1 10
p2 12
p3 13.5/;
Parameter availability(M)
/m1 16
m2 12/;
Table task(m,p)
p1 p2 p3
m1 2 5 2
m2 3 4 1;
```

- Variables: We are about to decide how many kg should be produced of each product. This variable is defined over product set. Another variable should be defined as the total profits which should be optimized. This variable (objective function) should not have any set index. Variables are OF, $X(p)$.
- Equations: One equation should describe the objective function and another one should enforce the constraint for availability of each machine.

```
Equations eq1,  eq2;
eq1(m).. sum(p, task(m,p)*x(p))=l=availability(M);
eq2 .. of=e=sum(p, profit(p)*x(p));
```

As it can be seen in this example, there are two types of equations as follows:

- Scalar equation: eq2 is an example of a scalar equation which is not defined over any set.
- Indexed equation: eq1 is an example of the indexed equation. We have to define the equation on index *m* since the index *m* exists in the formulation. In other words, the equation is valid for any element in the set *M*.

Three different relations can be defined in equations as follows:

- =e= Equality: this means that both sides of the equations should be equal to each other.
- =g= Greater than or equal: this means that the left-hand side of the equation should be greater than or equal to the right-hand side of the equation
- =l= Less than or equal: this means that the left-hand side of the equation should be less than or equal to the right-hand side of the equation

• Model: This example has two equations. Both should be included in the model definition.

```
Model example /all/;
```

It can also be defined as follows:

```
Model example /eq1,eq2/;
```

• Solve statement: The solve statement tells GAMS that the model is linear and the direction of the optimization should be toward maximizing the total benefits.

```
Solve example us LP Max OF;
```

The model type used in this code is linear programming (LP). There are various models that can be coded in GAMS coding as follows:

LP: linear programming
QCP: Quadratic programming (the model can only contain linear and quadratic terms)
NLP: Nonlinear problem with continuous constraints
DNLP: Nonlinear problem with discontinuous constraints
MIP: Mixed-integer linear programming
MIQCP: Mixed-integer quadratic constraint programming
MINLP: Mixed-integer nonlinear programming
RMIP: Mixed-integer problem where the integer variables are relaxed

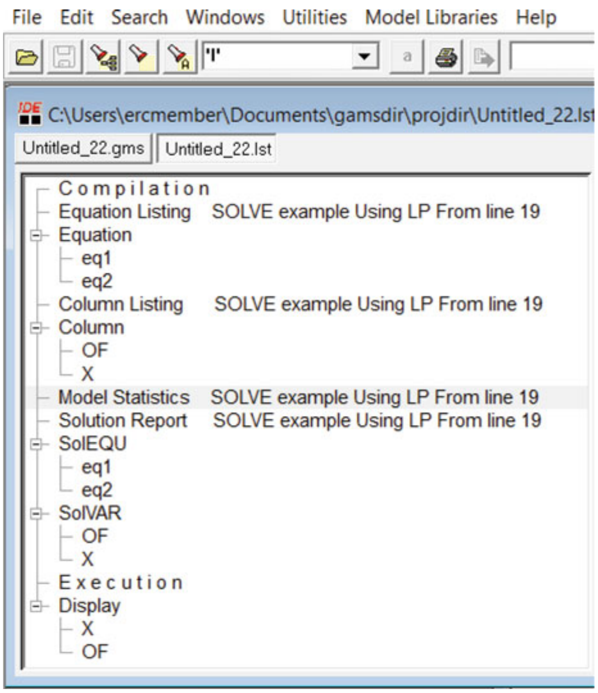
- Output: The outputs of the GAMS model can be displayed and also saved in an XSL file.

```
Display X.l,Of.l;
execute_unload "Example.gdx" X.l
execute 'gdxxrw.exe Example.gdx var=X   rng=Product!a1'
execute_unload "Example.gdx" OF.l
execute 'gdxxrw.exe Example.gdx var=OF   rng=OF!a1'
```

The overall GAMS code for solving the illustrative example is provided in GCode 1.1.

GCode 1.1 Illustrative GAMS example

```
Sets  M /m1*m2/, P /p1*p3/ ;
variables OF,X(p);
parameter profit(P)
/p1 10
 p2 12
 p3 13.5/;
parameter availability (M)
/m1 16
 m2 12/;
Table task(m,p)
      p1      p2      p3
m1    2       5       2
m2    3       4       1  ;
equations eq1,eq2;
eq1(m)..   sum(p, task(m,p)*x(p))=l=availability(M);
eq2    ..   of=l=sum(p, profit(p)*x(p));
X.lo(p)=1;
model example /all/;
Solve example us LP max OF;
display X.l,Of.l;
execute_unload "Example.gdx" X.l
execute 'gdxxrw.exe Example.gdx var=X  rng=Product!a1'
execute_unload "Example.gdx" OF.l
execute 'gdxxrw.exe Example.gdx var=OF  rng=OF!a1'
```


File Edit Search Windows Utilities Model Libraries Help
C:\Users\ercmember\Documents\gamsdir\projdir\Untitled_22.lst
Untitled_22.gms
Untitled_22.lst

```
Compilation
Equation Listing    SOLVE example Using LP From line 19
Equation
  eq1
  eq2
Column Listing      SOLVE example Using LP From line 19
Column
  OF
  X
Model Statistics    SOLVE example Using LP From line 19
Solution Report     SOLVE example Using LP From line 19
SolEQU
  eq1
  eq2
SolVAR
  OF
  X
Execution
Display
  X
  OF
```

Fig. 1.6 The structure of GAMS listing file

Once the model is solved the solve summary is available by clicking on the file.lst. The structure of GAMS listing file is shown in Fig. 1.6.

The listing file is explained as follows:

```
S O L V E S U M M A R Y
MODEL example OBJECTIVE OF
TYPE LP DIRECTION MAXIMIZE
SOLVER CPLEX FROM LINE 23
**** SOLVER STATUS 1 Normal Completion
**** MODEL STATUS 1 Optimal
**** OBJECTIVE VALUE 82.7500
```

The summary shows that the name of the model is *example*. The objective function to be optimized is named OF. The type of the model is linear programming (LP). The direction of the optimization is maximization. The solver used for solving the LP model is CPLEX. The solver status is 1 which means that the model is normally solved without error. The model status is 1 optimal. It means that the global optimal solution is found. The value of the objective function (OF) is 82.75. Different solver status might be reported once the model is solved.

SOLVER STATUS CODE	DESCRIPTION
1	Normal Completion
2	Iteration Interrupt
3	Resource Interrupt
4	Terminated by Solver
5	Evaluation Error Limit
6	Capability Problems
7	Licensing Problems
8	User Interrupt
9	Error Setup Failure
10	Error Solver Failure
11	Error Internal Solver Error
12	Solve Processing Skipped
13	Error System Failure

Different model status might be reported once the model is solved.

MODEL STATUS CODE	DESCRIPTION
1	Optimal
2	Locally Optimal
3	Unbounded
4	Infeasible
5	Locally Infeasible
6	Intermediate Infeasible
7	Intermediate Nonoptimal
8	Integer Solution
9	Intermediate Non-Integer
10	Integer Infeasible
11	Licensing Problems - No Solution
12	Error Unknown
13	Error No Solution
14	No Solution Returned
15	Solved Unique
16	Solved
17	Solved Singular
18	Unbounded - No Solution
19	Infeasible - No Solution

The model statistic can provide some useful information regarding the developed model. It is indicating that there are two blocks of equations (eq1,eq2) in the developed model. The single equations are three because eq1 has two single equations ($m1, m2$) and the eq2 has only one equation. There are two blocks of variables x, OF but since X has three variables ($p1, p2, p3$), then the number of total variables is 4.

```
MODEL STATISTICS
BLOCKS OF EQUATIONS 2 SINGLE EQUATIONS 3
BLOCKS OF VARIABLES 2 SINGLE VARIABLES 4
```

By clicking on the SolVar tab, some useful information regarding the variables will be obtained as follows:

```
LOWER LEVEL UPPER MARGINAL
—- VAR OF -INF 82.750 +INF .
—- VAR X
LOWER LEVEL UPPER MARGINAL
p1 1.000 1.000 +INF -3.500
p2 1.000 1.000 +INF -21.750
p3 1.000 4.500 +INF .
```

This means that the minimum limit of variable X is 1 and the upper limits are $+\infty$. The marginal values are also revealed. Each variable in GAMS has five attributes as follows:

- Variable.lo: indicates the minimum limit of a variable
- Variable.up: indicates the maximum limit of a variable
- Variable.l: indicates the level of the variable. In other words, this is the actual value of the variable.
- Variable.fx: indicates the level of the variable is fixed and is not changing. It has the same impact of defining the low and up attributes the same as each other.
- Variable.m: indicates the marginal value of a variable. It shows how much sensitive is the objective function to the changes of this variable. For example, $X.m(p1) = -3.5$ this means that for 1 unit of increase in $X(p1)$ the objective function will reduce by -3.5.

The variables used in this code are all real variables; however, different types of variables can be defined in GAMS as follows:

Variable type	Description	Lower limit	Upper limit
Free	No bounds on variable. Both bounds can be changed from the default values by the user	$-\infty$	$+\infty$
Positive	The positive variable can only take positive values	0	$+\infty$
Negative	The negative variable can only take negative values	$-\infty$	0
Binary	Discrete variable that can only take values of 0 or 1	0	1
Integer	Discrete variable that can only take integer values between the limits	0	100

It should be noted that the default values for variable limits can be modified by the user.

1.4 Conditional Statements

The Dollar ($) condition is broadly used in GAMS coding. Various applications are explained through some examples.

- Suppose we need to have a conditional statement like this if $A = C$ then $B = D$. This statement is modeled as follows:

```
(B=D)$(A=C)
```

If $(A \neq C)$ then no assignment to B will happen.

- If $A = C$ then $B = D$ otherwise $B = 0$. This statement is modeled as follows:

```
B=D$(A=C)
```

- Suppose that we need to filter some elements in eq1(m). In other words, this equation should be valid for every m elements except some of them.

```
eq1(m)$(ord(m)=1) .. sum(p, task(m,p)*x(p))=l=availability(M);
```

This would force the equation only for $m1$.

```
eq1(m)$(ord(m) ≥ 2) .. sum(p, task(m,p)*x(p))=l=availability(M);
```

This would force the equation only for $m2$ and $m3$. It should be noted that no variable can exist in conditional statement.

- If A and B then $C = D$, this is coded as follows:

```
(C=D)$( A and B)
```

1.5 Loop Definition in GAMS

The looping is used for executing one or more statements repeatedly. There are different forms of looping in GAMS as follows:

1.5.1 LOOP Statement

One of the techniques for defining the loop in GAMS is loop statement.

```
Loop(setname,
Statement 1 ;
Statement 2 ;
Statement 3 ;
);
```

The statements 1–3 are executed N times. N is equal to the cardinality of the "setname." These statements can assign values to the variable limits or describe the relation between some parameters (not the variables). No variable can appear in the loop statements.

```
set counter /c1*c4/;
Loop(counter,
X.lo(p)=0.1*ord(counter);
Solve example us LP max OF;
);
```

In the first counter, the minimum values of all X variables would be $0.1 * 1$ and the model is solved. The second counter sets the minimum values of X variable equal to $0.1 * 2 = 0.2$ and then the model is solved. The point that should be noted here is that the set/parameter/variable/model definition should be outside the loop.

This kind of modeling would be useful in conducting sensitivity analysis. Some parameters are varied and then the impacts on the objective function are investigated as follows:

```
set counter /c1*c4/;
parameter report(counter,*);
Loop(counter,
X.lo(p)=0.1*ord(counter);
Solve example us LP max OF;
report(counter,'lowerlimit')=0.1*ord(counter);
report(counter,'OF')=OF.l;
);
```

1.5.2 WHILE Statement

One of the techniques for defining the loop in GAMS is while statement.

```
while (condition,
Statement 1 ;
Statement 2 ;
Statement 3 ;
);
```

The statements 1–3 are executed as far as the condition is true. The condition will be changed inside the loop based on some logic otherwise the loop will continue forever!

1.5.3 FOR Statement

One of the techniques for defining the loop in GAMS is FOR statement.

```
scalar Itermax /10/;
scalar iteration ;
for (iteration=1 to Itermax,
Statement 1 ;
Statement 2 ;
Statement 3 ;
);
```

The statements 1–3 are executed for Itermax times.

1.5.4 REPEAT-UNTIL Statement

One of the techniques for defining the loop in GAMS is Repeat-until statement.

```
repeat ( Statement 1 ;
Statement 2 ;
Statement 3 ;
until condition );
```

The statements 1–3 are executed until the condition becomes logically true (it should be initially false).

1.6 Linking GAMS and Excels

1.6.1 Reading from Excel

GAMS is able to read the data from xls files. It is also convenient to use xls as an interface between GAMS and other platforms. Reading the data from xls file is straightforward as follows:

```
Parameter Level(m,p);
\$CALL GDXXRW.EXE Example.xls par=Level rng=task!E5:H7
\$GDXIN Example.gdx
\$LOAD Level
```

In this case, a parameter like Level (m,p) is defined. The xls file name (Example.xls) is indicated and the range of data is specified (sheet=task, cells:

E5:H7). The data is read from xls file and is written on Example.gdx file. The parameter level is loaded and can be used in the code.

1.6.2 Writing to Excel

It is usually desirable to save the output of a GAMS code in an excel file. The procedure is quite straightforward.

For variables, the following commands should be executed. The user can indicate which variable (X.l,OF.l) should be written in Excel file. The name of the excel file (Example.xls) and what would be the sheet name (product, OF). The cell a1 is chosen as the starting cell in the specified sheet for writing the data.

```
execute_unload "Example.gdx" X.l
execute 'gdxxrw.exe Example.gdx var=X  rng=Product!a1'
execute_unload "Example.gdx" OF.l
execute 'gdxxrw.exe Example.gdx var=OF  rng=OF!a1'
```

If the user wants to write the parameter X to an excel file then the command would be as follows:

```
execute_unload "Example.gdx" X
execute 'gdxxrw.exe Example.gdx Par=X  rng=Product!a1'
```

In this case, no '.l' is needed since it is a parameter, not a variable.

1.7 Error Debugging in GAMS

Once the code is written the "Shift key + F9" should be pressed. The GAMS compiler will check the developed code without executing it. The coding errors are unavoidable in every programming language, and GAMS is not an exception. Usually, a list of errors is generated which should be taken care of. The first thing to keep in mind is to start from the first error and fix them one by one. In this section, some common coding errors and best way of debugging them are explained.

- Error 140: Unknown symbol: GAMS compiler generates this error when there are some undefined symbols in the model.
- Error 246: Objective variable is not a free variable

  This error will usually happen when the objective function (which should be minimized/maximized) is not set as a free variable. For example, it is defined as a positive/negative variable. In case that the decision maker needs the objective function be a positive variable, the lower limit of the variable should be specified using $.lo = 0$ command.

- Error: Unbounded variable:
  This means that the objective function is equal to $-\infty$ or $+\infty$. This usually happens when the bounds of the objective function are not properly defined, or optimization direction is not properly defined in solve statement. For example, if the objective is to be minimized but it is maximized by mistake. The following error will be generated by GAMS.

```
**** ERRORS/WARNINGS IN VARIABLE OF
1 error(s): Unbounded variable
```

- Error 149: Uncontrolled set entered as constant
  This error usually happens when the statements are not properly written. Consider the following line of code

```
eq1 .. sum(p, task(m,p)*x(p))=l=availability(M);
```

  The equation should be valid for every *m*, but the equation label is not defined over the set *m*. The correct format for defining this equation is as follows:

```
eq1(m) .. sum(p, task(m,p)*x(p))=l=availability(M);
```

- Error 143: A suffix is missing
  This error will happen when a variable is used outside of the equation environment. For example, the following line is trying to assign a value to variable $X(p)$ but not in any equation:

```
x(p)=1;
```

  This is not a valid way of doing this. If the variable $X(p)$ should be fixed to a constant value it should be defined as follows:

```
x.fx(p)=1;
```

  As a general rule, the variables cannot appear outside the equations. Only the variable's attributes can be modified outside the equation environment.

This means $X.lo(p)$, $X.l(p)$, $X.up(p)$ can be changed without using equation environment.

Another situation that this error might happen is in display command. Consider the following part of a GAMS code:

```
Solve example us LP max Z; Display X , Of.l;
```

This will generate the following error:

```
143 A suffix is missing
**** 1 ERROR(S) 0 WARNING(S)
```

This is because *X* is a variable and display command can only show the attributes of the *X* variable. The correct form for showing the value of *X* variable is as follows:

```
solve example us LP max Z;
Display X.l , Of.l;
```

- Error 245: Objective variable not referenced in model
  This means that the solve statement is trying to minimize/maximize a variable which does not exist in the defined model.
- Error 141: Symbol declared, but no values have been assigned
  This error happens when the variable or parameter is defined, but it is not implicitly assigned any value and also does not appear in any equation. In other words, GAMS has no info about this symbol, but we are asking GAMS some information about it. Consider the following GAMS code:

```
Sets M /m1*m2/, P /p1*p3/ ;
Variables OF,x(p),Z;
parameter profit(P)
/p1 10
p2 12
p3 13.5/;
Parameter availability (M)
/m1 16
```

(continued)

```
m2 12/;
Table task(m,p)
p1 p2 p3
m1 2 5 2
m2 3 4 1 ;
Equations eq1,eq2;
eq1(m) .. sum(p, task(m,p)*x(p))=l=availability(M);
eq2 .. of=l=sum(p, profit(p)*x(p));
X.lo(p)=1;
Model example /all/;
Solve example us LP max OF;
display X.l , Of.l , Z.l;
```

This code would generate Error 141 because although variable *Z* is defined but since it has no initial assignment and does not appear in the model then GAMS knows nothing about it.

- Error 225: Floating entry ignored

  This error is a very common error specially for beginners. This happens when a table is not properly typed in GAMS. All data should be aligned under the column label.

```
      P1   P2   P3
M1     2    5         2
M2     3    4    1
```

The correct form of typing the above table is as follows:

```
      P1   P2   P3
M1    2    5    2
M2    3    4    1
```

Handling the large tables in GAMS would be much easier if the tool named "xls2gams.exe" is used. This tool is available in the same folder that GAMS is installed. The function of this tool is copying data from xls file into GAMS format. Using this tool to import xls data to GAMS is highly recommended.

- Error 56: Endogenous operands for * not allowed in linear models

Consider the following code:

```
Sets M /m1*m2/, P /p1*p3/ ;
Variables OF,x(p),Z;
parameter profit(P)
/p1 10
p2 12
p3 13.5/;
Parameter availability (M)
/m1 16
m2 12/;
Table task(m,p)
p1 p2 p3
m1 2 5 2
m2 3 4 1 ;
Equations eq1,eq2;
eq1(m) .. sum(p, task(m,p)*x(p))=l=availability(M);
eq2 .. of=l=sum(p, profit(p)*x(p)*x(p));
X.lo(p)=1;
Model example /all/;
Solve example us LP max OF;
display X.l , Of.l , Z.l;
```

The following error will be generated after compiling this code. In order to find out why this error is happening we should double-check the model. The solve statement is indicating that the model is LP but in eq2 we can see $profit(p) * x(p) * x(p)$. This means that the model is not LP and solve statement should be modified as follows:

```
Solve example us NLP max OF;
or
Solve example us QCP max OF;
```

- Error 37:
  Consider the following line in a GAMS code:

```
eq2 .. of=sum(p, profit(p)*x(p));
```

This line is not properly defined because the correct operator for stating the equality is not used in equation environment. The only operators that can be used are ‘=l=’ or ‘=e=’ or ‘=g=’. the correct format is as follows:

```
eq2 .. of=e=sum(p, profit(p)*x(p));
```

- Error 409:
  Consider the following lines in a GAMS code:

```
eq1(m) .. sum(p, task(m,p)*x(p))=l=availability(M)
eq2 .. of=sum(p, profit(p)*x(p)*x(p));
```

This is because a semicolon (;) is missing at the end of eq1.

1.8 General Programming Remarks

Like other programming languages, there are different ways of modeling the problem in GAMS. The so-called “best way” is yet to be discovered. Some programming tips are given in this section as follows:

- GAMS compiler is not case sensitive. The lowercase letters and capital letters are not distinguished. For example, $P(t)$ is the same as $p(T)$ or $P(T)$ or $p(t)$.
- Try to add explanations for your equations and symbols you define in your code. Use * for commenting a line or $ontext $offtext for a block of comments.
- Choose the symbol names carefully and meaningfully. For example for the power generated by power plants, $P(g)$ is a better choice than $X(i)$. You can define g as the set of generating units and variable P as the generated power.
- Don’t get frightened by the errors once your code is compiled. The best thing to see is the error flag. Sometimes although the model does not generate any error flag (since you have obeyed the GAMS syntax) but it is not correctly modeling the given problem.
- Keep in mind that the machine is always right so GAMS cannot go wrong. Debugging is also a part of the coding process.
- Always check the GAMS output. The variables, model statistics, and solver statistics should be checked to see if any flag is raised. If the output of GAMS model is not what you expect then there should be something wrong with your model (not the GAMS). Double-check the equations and see if the developed model is actually what it is desired to be or not.

- If you need to specify some limits for your variables then try to do it using .lo or .up attributes. Although creating a new equation for modeling this constraint is “legal”, however, it is totally “inappropriate”. For example, you need to state that the minimum value of variable $P(g)$ is 1 then it can be easily done using $P.\mathrm{lo}(g) = 1$. Sometimes the variables should be limited using some tables of parameters. For example, the minimum values of $P(g)$ are stored in a parameter $P\mathrm{min}(g)$ then $P.\mathrm{lo}(g) = P\mathrm{min}(g)$ will do the job. As far as no variable exists then no equation is needed. If $P\mathrm{min}(g)$ is a variable, then the following line will be appropriate for modeling this constraint.

```
eq(g) .. P(g)=g=Pmin(g);
```

  It should be noted that *eq* is defined over the set *g* since the inequality should be valid for every element belonging to the set *g*.
- Although GAMS is a robust programming language less number of variables and equations are always welcome. Try to avoid unnecessary equations and variables. Use filtering to omit unnecessary equations. For example, we need to define an equation which calculates the power flow of a line connecting ‘bus’ to ‘node’ in DC power flow formulation. The following GAMS code might be the first attempt to model it.

```
Eq(bus,node) .. Pij(bus,node)=e=(d(bus)-d(node))/data(bus,node,’x’);
```

  Suppose the network has 186 lines and 118 buses. This kind of coding would create two problems:

  – The parameter data(bus, node,‘x’) is zero when bus and node are not connected to each other. GAMS might generate a division by Zero error! It’s not always easy to find and remove the cause of error.
  – If you check the model you can see that this equation adds 13,924 single equations to the model. You might ask yourself why? We were expecting to have only 186 equations (or double that since flow in opposite direction should also be calculated). It is almost 74 times bigger than what we were expecting to have. This is because GAMS is considering every combination between “bus” and “node” (118*118).

  The better way to code this line is as follows:

```
Eq(bus,node)$branch(bus,node,’x’)..Pij(bus,node)=e=(d(bus)-
d(node))/data(bus,node,’x’);
```

In this way, the equation is filtered and is only valid for those lines which have nonzero reactance values. It also reduces the computation time and the chance of difficulties in finding a feasible/optimal solution.

- Don’t forget to put a semicolon (;) at the end of every line (sometimes it is not necessary but if you have just started coding in GAMS, do it).
- Different solvers might use different approaches to solve the model. Try different solver to find out which one is more successful/faster in finding a solution for your problem.
- The default value for undefined elements of a table or parameter is zero. For example, consider a parameter like Data(bus,node) representing the line reactance; Suppose that we define Data(‘2’,‘4’)=0.5; If this parameter is displayed, you can see that all elements are zero except Data(‘2’,‘4’). If you are a power engineer, you know that the reactance between bus number 2 and bus number 4 should be the same as reactance between bus number 4 and bus number 2. In other words, it should be symmetrical. Unfortunately (or fortunately) GAMS is not a power system engineer and does not understand even a word from power system or electrical engineering. GAMS only understands whatever it is told by the user (code developer).
- Go to your GAMS model library and have a look at those developed codes. There are loads of new things to learn even for experienced GAMS code developers. The GAMS model library can be accessed through the interface as shown in Fig. 1.7.
- Try to provide all of the variables appropriate upper and lower limits as well as initial starting values. The initial values can be assigned to the variables using .l expression (do it before solving statement). For example, voltage variables can be assigned 1 per unit values as the initial starting values. This is a double-edged sword. This is because providing a good starting value can highly improve the solution procedure especially in nonlinear or mixed integer nonlinear models but a poor starting value might slowdown the solver.
- It is very likely that you get an infeasible status for your model. Especially when the model is large and includes a large number of variables. The following steps might be helpful in resolving the problem:

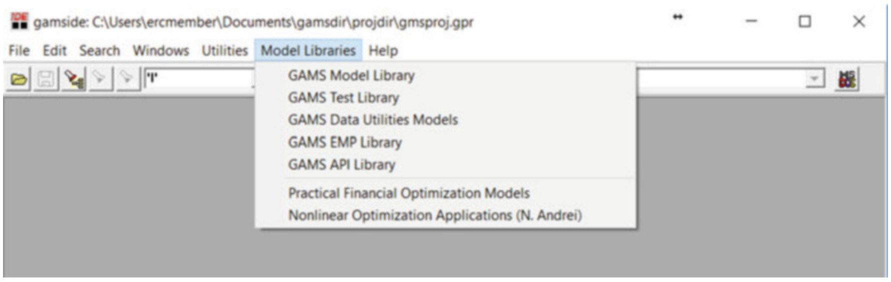

gamside: C:\Users\ercmember\Documents\gamsdir\projdir\gmsproj.gpr
File Edit Search Windows Utilities Model Libraries Help
GAMS Model Library
GAMS Test Library
GAMS Data Utilities Models
GAMS EMP Library
GAMS API Library
Practical Financial Optimization Models
Nonlinear Optimization Applications (N. Andrei)

Fig. 1.7 GAMS model library

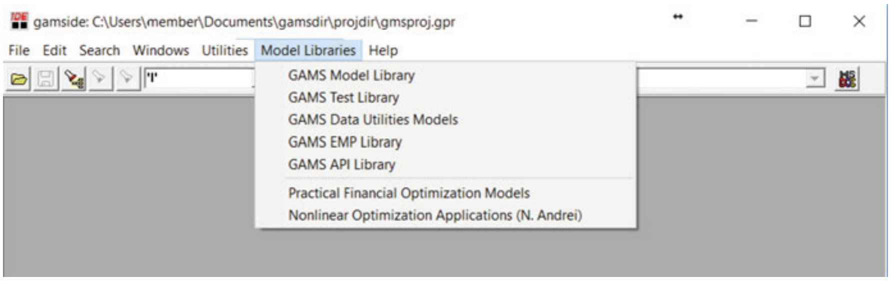

gamside: C:\Users\member\Documents\gamsdir\projdir\gmsproj.gpr
File Edit Search Windows Utilities Model Libraries Help
GAMS Model Library
GAMS Test Library
GAMS Data Utilities Models
GAMS EMP Library
GAMS API Library
Practical Financial Optimization Models
Nonlinear Optimization Applications (N. Andrei)

Fig. 1.8 GAMS help menu

- Check to see if you can better express your model. It is always desired to stay as close as possible to the linear form of expressing the equations.
- Provide better initial values for your variables.
- Relax the variable limits, rerun the model. If the problem is resolved, then it means that the variable limits should be revised. If your model contains integer/binary variables, then you can solve the model using relaxed option. For example, the MIP model can be solved using RMIP. This would ask the GAMS to neglect integer nature of the variables (the variable limits remain unchanged). For relaxing the MINLP and MIQCP, you should use RMINLP and RMIQCP, respectively.
- Remove some equations or add some slack variables to the model to see if you can find the trouble making equations
- Ask support from those experienced GAMS code developers (if they have time and are willing to contribute to your project). This option is intentionally placed at the end of suggestion list.

There is a set of good resources that can be accessed through the GAMS interface as it is shown in Fig. 1.8. Additionally, some other useful tutorials are listed as follows:

- A GAMS tutorial [70]
- GAMS language guide [71]
- GAMS—Modeling and Solving Optimization Problems [72]
- GAMS and MATLAB interface [73–75]
- Grid-enabled optimization with GAMS [76]
- Practical financial optimization in GAMS [77]

1.9 Book Structure

The objectives pursued in this book are the following:

- Familiarizing the readers with optimization concept through multiple examples
- Introducing different optimization problems that the decision makers might face in power system studies

- Providing robust solutions for different operation/planning problems in power system studies
- Exploring the capabilities of GAMS in conducting sensitivity analysis

This book consists of ten chapters. A brief description of each chapter is provided here:

- Chapter 1 provides the instructions on how to start coding in GAMS. Different programming elements and some coding tricks are introduced and explained. Reading this chapter would be helpful to understand what to expect from GAMS in dealing with optimization problems.
- Chapter 2 provides some insights to the reader on how to solve simple optimization problems through some illustrative examples. These examples are pure mathematics, and no power system or electrical engineering background is required for the readers. This chapter can also be used for those who might be only interested in learning GAMS for interdisciplinary applications.
- Chapter 3 discusses how to model the dispatching problem of different power plants in a single snapshot (single period). Different power plant technologies are explained such as thermal power, wind turbine, CHP, and hydro-power plants.
- Chapter 4 provides a solution for dynamic economic dispatch problem (multi-period problems). The time-dependent optimal dispatch decisions are modeled and solved.
- Chapter 5 explains how unit commitment problem can be modeled and solved in GAMS. The developed codes in this chapter are linear and categorized as mixed integer linear programming (MIP) models. The inputs are generator's characteristics, electricity prices, and demands. The outputs of these codes are on/off status of units and their operating schedules.
- Chapter 6 provides a solution for optimal power flow (OPF) problem. The active/reactive power output of generating units as well as the network variables (voltage magnitudes and angles) are determined in OPF to minimize total operating costs. Different OPF models are investigated such as single and multi-period DC/AC optimal power flow. The appropriate linear/nonlinear models are developed and solved in this chapter.
- Chapter 7 provides a solution for modeling the operation and planning problems of energy storage systems (ESS). The inputs are generator's characteristics, electricity prices, demands, and network topology. The outputs of this code are operating schedules/locations and sizes of ESS units.
- Chapter 8 provides a solution for allocation of Phasor Measurement Units (PMU) problem in transmission networks to maximize the power system observability. The PMU can measure the voltage phasor at the connection bus, and also it measures the current phasor of any branch connected to the bus hosting the PMU. Different cases are analyzed, and the problem is tested on some standard IEEE cases.
- Chapter 9 provides a solution for some transmission network operation and planning studies in GAMS. The transmission investment regarding building new lines and power flow controllers (phase shifter), sensitivity factors, and

transmission switching have been discussed in this chapter. The GAMS code for solving each optimization problem is developed and discussed.

- Chapter 10 provides a solution for Energy System Integration (ESI) problem in GAMS. The question which is answered in this chapter is how to harvest the flexibilities in different energy sectors by coordinated operation of these sectors.

References

1. J.L.C. Meza, M.B. Yildirim, A.S.M. Masud, A model for the multiperiod multiobjective power generation expansion problem. IEEE Trans. Power Syst. **22**(2), 871–878 (2007)
2. 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)
3. R. Fang, D.J. Hill, A new strategy for transmission expansion in competitive electricity markets. IEEE Trans. Power Syst. **18**(1), 374–380 (2003)
4. R. Romero, A. Monticelli, A. Garcia, S. Haffner, Test systems and mathematical models for transmission network expansion planning. IEE Proc. Gener. Transm. Distrib. **149**(1), 27–36 (2002)
5. 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)
6. S. Gerbex, R. Cherkaoui, A.J. Germond, Optimal location of multi-type facts devices in a power system by means of genetic algorithms. IEEE Trans. Power Syst. **16**(3), 537–544 (2001)
7. C. Wang, M.H. Nehrir, Analytical approaches for optimal placement of distributed generation sources in power systems. IEEE Trans. Power Syst. **19**(4), 2068–2076 (2004)
8. W. El-Khattam, K. Bhattacharya, Y. Hegazy, M.M.A. Salama, Optimal investment planning for distributed generation in a competitive electricity market. IEEE Trans. Power Syst. **19**(3), 1674–1684 (2004)
9. A. Keane, M. O'Malley, Optimal allocation of embedded generation on distribution networks. IEEE Trans. Power Syst. **20**(3), 1640–1646 (2005)
10. S. Sundhararajan, A. Pahwa, Optimal selection of capacitors for radial distribution systems using a genetic algorithm. IEEE Trans. Power Syst. **9**(3), 1499–1507 (1994)
11. B. Milosevic, M. Begovic, Nondominated sorting genetic algorithm for optimal phasor measurement placement. IEEE Trans. Power Syst. **18**(1), 69–75 (2003)
12. B. Gou, Generalized integer linear programming formulation for optimal PMU placement. IEEE Trans. Power Syst. **23**(3), 1099–1104 (2008)
13. Y.M. Atwa, E.F. El-Saadany, Optimal allocation of ESS in distribution systems with a high penetration of wind energy. IEEE Trans. Power Syst. **25**(4), 1815–1822 (2010)
14. H. Oh, Optimal planning to include storage devices in power systems. IEEE Trans. Power Syst. **26**(3), 1118–1128 (2011)
15. P. Maghouli, A. Soroudi, A. Keane, Robust computational framework for mid-term techno-economical assessment of energy storage. IET Gener. Transm. Distrib. **10**(3), 822–831 (2016)
16. A. Soroudi, M. Afrasiab, Binary PSO-based dynamic multi-objective model for distributed generation planning under uncertainty. IET Renew. Power Gener. **6**(2), 67–78 (2012)
17. G. Blanco, F. Olsina, F. Garces, C. Rehtanz, Real option valuation of facts investments based on the least square monte carlo method. IEEE Trans. Power Syst. **26**(3), 1389–1398 (2011)
18. B. Chen, J. Wang, L. Wang, Y. He, Z. Wang, Robust optimization for transmission expansion planning: minimax cost vs. minimax regret. IEEE Trans. Power Syst. **29**(6), 3069–3077 (2014)
19. V. Miranda, M.A.C.C. Matos, Distribution system planning with fuzzy models and techniques, in *10th International Conference on Electricity Distribution, 1989 (CIRED 1989)*, vol. 6, Brighton (1989), pp. 472–476

20. S. Civanlar, J.J. Grainger, H. Yin, S.S.H. Lee, Distribution feeder reconfiguration for loss reduction. IEEE Trans. Power Delivery **3**(3), 1217–1223 (1988)
21. M. Carrion, J.M. Arroyo, A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem. IEEE Trans. Power Syst. **21**(3), 1371–1378 (2006)
22. J.M. Arroyo, A.J. Conejo, Optimal response of a thermal unit to an electricity spot market. IEEE Trans. Power Syst. **15**(3), 1098–1104 (2000)
23. D.W. Ross, S. Kim, Dynamic economic dispatch of generation. IEEE Trans. Power Apparatus Syst. **PAS-99**(6), 2060–2068 (1980)
24. Z.-L. Gaing, Particle swarm optimization to solving the economic dispatch considering the generator constraints. IEEE Trans. Power Syst. **18**(3), 1187–1195 (2003)
25. J. Endrenyi, S. Aboresheid, R.N. Allan, G.J. Anders, S. Asgarpoor, R. Billinton, N. Chowdhury, E.N. Dialynas, M. Fipper, R.H. Fletcher, C. Grigg, J. McCalley, S. Meliopoulos, T.C. Mielnik, P. Nitu, N. Rau, N.D. Reppen, L. Salvaderi, A. Schneider, C. Singh, The present status of maintenance strategies and the impact of maintenance on reliability. IEEE Trans. Power Syst. **16**(4), 638–646 (2001)
26. A.G. Bakirtzis, P.N. Biskas, A decentralized solution to the DC-OPF of interconnected power systems. IEEE Trans. Power Syst. **18**(3), 1007–1013 (2003)
27. R.D. Zimmerman, C.E. Murillo-Sanchez, R.J. Thomas, Matpower: steady-state operations, planning, and analysis tools for power systems research and education. IEEE Trans. Power Syst. **26**(1), 12–19 (2011)
28. Y.M. Atwa, E.F. El-Saadany, M.M.A. Salama, R. Seethapathy, Optimal renewable resources mix for distribution system energy loss minimization. IEEE Trans. Power Syst. **25**(1), 360–370 (2010)
29. K.R.C. Mamandur, R.D. Chenoweth, Optimal control of reactive power flow for improvements in voltage profiles and for real power loss minimization. IEEE Trans. Power Apparatus Syst. **PAS-100**(7), 3185–3194 (1981)
30. Y. Bai, H. Zhong, Q. Xia, C. Kang, A two-level approach to ac optimal transmission switching with an accelerating technique. IEEE Trans. Power Syst. **32**(2), 1616–1625 (2017)
31. N. Rotering, M. Ilic, Optimal charge control of plug-in hybrid electric vehicles in deregulated electricity markets. IEEE Trans. Power Syst. **26**(3), 1021–1029 (2011)
32. F. Wen, A.K. David, Optimal bidding strategies and modeling of imperfect information among competitive generators. IEEE Trans. Power Syst. **16**(1), 15–21 (2001)
33. A. Baillo, M. Ventosa, M. Rivier, A. Ramos, Optimal offering strategies for generation companies operating in electricity spot markets. IEEE Trans. Power Syst. **19**(2), 745–753 (2004)
34. H.B. Gooi, D.P. Mendes, K.R.W. Bell, D.S. Kirschen, Optimal scheduling of spinning reserve. IEEE Trans. Power Syst. **14**(4), 1485–1492 (1999)
35. K.A. Papadogiannis, N.D. Hatziargyriou, Optimal allocation of primary reserve services in energy markets. IEEE Trans. Power Syst. **19**(1), 652–659 (2004)
36. A. Soroudi, P. Siano, A. Keane, Optimal DR and ESS scheduling for distribution losses payments minimization under electricity price uncertainty. IEEE Trans. Smart Grid **7**(1), 261–272 (2016)
37. S. Dutta, S.P. Singh, Optimal rescheduling of generators for congestion management based on particle swarm optimization. IEEE Trans. Power Syst. **23**(4), 1560–1569 (2008)
38. C. Murphy, A. Soroudi, A. Keane, Information gap decision theory-based congestion and voltage management in the presence of uncertain wind power. IEEE Trans. Sust. Energy **7**(2), 841–849 (2016)
39. B. Hayes, I. Hernando-Gil, A. Collin, G. Harrison, S. Djoki, Optimal power flow for maximizing network benefits from demand-side management. IEEE Trans. Power Syst. **29**(4), 1739–1747 (2014)
40. A.J. Conejo, J.M. Morales, L. Baringo, Real-time demand response model. IEEE Trans. Smart Grid **1**(3), 236–242 (2010)
41. A. Rabiee, A. Soroudi, A. Keane, Information gap decision theory based OPF with HVDC connected wind farms. IEEE Trans. Power Syst. **30**(6), 3396–3406 (2015)

42. A. Soroudi, A. Rabiee, A. Keane, Information gap decision theory approach to deal with wind power uncertainty in unit commitment. Electr. Power Syst. Res. **145**, 137–148 (2017)
43. A.J. Conejo, M. Carrión, J.M. Morales, *Decision Making Under Uncertainty in Electricity Markets*, vol. 1 (Springer, New York, 2010)
44. E. Zio, *The Monte Carlo Simulation Method for System Reliability and Risk Analysis* (Springer, London, 2013)
45. 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)
46. S. Granville, Optimal reactive dispatch through interior point methods. IEEE Trans. Power Syst. **9**(1), 136–146 (1994)
47. T. Ding, R. Bo, F. Li, H. Sun, A bi-level branch and bound method for economic dispatch with disjoint prohibited zones considering network losses. IEEE Trans. Power Syst. **30**(6), 2841–2855 (2015)
48. S. Binato, M.V.F. Pereira, S. Granville, A new benders decomposition approach to solve power transmission network design problems. IEEE Trans. Power Syst. **16**(2), 235–240 (2001)
49. D.K. Molzahn, J.T. Holzer, B.C. Lesieutre, C.L. DeMarco, Implementation of a large-scale optimal power flow solver based on semidefinite programming. IEEE Trans. Power Syst. **28**(4), 3987–3998 (2013)
50. R.A. Jabr, Exploiting sparsity in SDP relaxations of the OPF problem. IEEE Trans. Power Syst. **27**(2), 1138–1139 (2012)
51. T. Wu, M. Rothleder, Z. Alaywan, A.D. Papalexopoulos, Pricing energy and ancillary services in integrated market systems by an optimal power flow. IEEE Trans. Power Syst. **19**(1), 339–347 (2004)
52. J.M. Arroyo, F.D. Galiana, On the solution of the bilevel programming formulation of the terrorist threat problem. IEEE Trans. Power Syst. **20**(2), 789–797 (2005)
53. D.I. Sun, B. Ashley, B. Brewer, A. Hughes, W.F. Tinney, Optimal power flow by Newton approach. IEEE Trans. Power Apparatus Syst. **PAS-103**(10), 2864–2880 (1984)
54. F. Milano, Continuous Newton's method for power flow analysis. IEEE Trans. Power Syst. **24**(1), 50–57 (2009)
55. E.C. Finardi, E.L. da Silva, Solving the hydro unit commitment problem via dual decomposition and sequential quadratic programming. IEEE Trans. Power Syst. **21**(2), 835–844 (2006)
56. I.P. Abril, J.A.G. Quintero, Var compensation by sequential quadratic programming. IEEE Trans. Power Syst. **18**(1), 36–41 (2003)
57. B. Enacheanu, B. Raison, R. Caire, O. Devaux, W. Bienia, N. HadjSaid, Radial network reconfiguration using genetic algorithm based on the matroid theory. IEEE Trans. Power Syst. **23**(1), 186–195 (2008)
58. 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)
59. T. Amraee, A.M Ranjbar, R. Feuillet. Immune-based selection of pilot nodes for secondary voltage control. Eur. Trans. Electr. Power **20**(7), 938–951 (2010)
60. T. Satoh, K. Nara, Maintenance scheduling by using simulated annealing method [for power plants]. IEEE Trans. Power Syst. **6**(2), 850–857 (1991)
61. J.G. Vlachogiannis, K.Y. Lee. Quantum-inspired evolutionary algorithm for real and reactive power dispatch. IEEE Trans. Power Syst. **23**(4), 1627–1636 (2008)
62. C. Dai, W. Chen, Y. Zhu, X. Zhang, Seeker optimization algorithm for optimal reactive power dispatch. IEEE Trans. Power Syst. **24**(3), 1218–1231 (2009)
63. T. Amraee, Coordination of directional overcurrent relays using seeker algorithm. IEEE Trans. Power Delivery **27**(3), 1415–1422 (2012)
64. D.N. Vo, P. Schegner, W. Ongsakul, Cuckoo search algorithm for non-convex economic dispatch. IET Gener. Transm. Distrib. **7**(6), 645–654 (2013)
65. A.A. El-fergany, A.Y. Abdelaziz, Capacitor allocations in radial distribution networks using cuckoo search algorithm. IET Gener. Transm. Distrib. **8**(2), 223–232 (2014)

66. M. Barati, M.M. Farsangi, Solving unit commitment problem by a binary shuffled frog leaping algorithm. IET Gener. Transm. Distrib. **8**(6), 1050–1060 (2014)
67. R. Roche, L. Idoumghar, B. Blunier, A. Miraoui, Imperialist competitive algorithm for dynamic optimization of economic dispatch in power systems, in *International Conference on Artificial Evolution (Evolution Artificielle)* (Springer, Berlin, 2011), pp. 217–228
68. W.-M. Lin, F.-S. Cheng, M.-T. Tsay, An improved tabu search for economic dispatch with multiple minima. IEEE Trans. Power Syst. **17**(1), 108–112 (2002)
69. C.F. Chang, Reconfiguration and capacitor placement for loss reduction of distribution systems by ant colony search algorithm. IEEE Trans. Power Syst. **23**(4), 1747–1755 (2008)
70. R.E. Rosenthal, A GAMS tutorial. Technical note (1992)
71. A. Brooke, D. Kendrick, A. Meeraus, R. Raman, R.E. Rosenthal, *Gams. A Users Guide* (GAMS Development Corporation, Washington, DC, 2005)
72. A. Geletu, *Gams-Modeling and Solving Optimization Problems* (TU-Ilmenau, Faculty of Mathematics and Natural Sciences, Department of Operation Research & Stochastrics, Ilmenau, 2008)
73. M.C. Ferris, Matlab and GAMS: interfacing optimization and visualization software. Mathematical Programming Technical Report, 98:19 (1998)
74. L. Wong et al., *Linking Matlab and Gams: A Supplement* (University of Victoria, Department of Economics, Victoria, BC, 2009)
75. M.C. Ferris, R. Jain, S. Dirkse, Gdxmrw: Interfacing GAMS and matlab (2011). http://www.gams.com/dd/docs/tools/gdxmrw.pdf
76. M.R. Bussieck, M.C. Ferris, A. Meeraus, Grid-enabled optimization with GAMS. INFORMS J. Comput. **21**(3), 349–362 (2009)
77. S.S. Nielson, A. Consiglio, *Practical Financial Optimization: A Library of GAMS Models* (Wiley, New York, 2010)