BASIC OPTIMIZATION MODELS FOR WATER AND ENERGY MANAGEMENT

By
Daene C. McKinney
And
Andre G. Savitsky

June 1999
(revision 1, October 2000)
(revision 5, February 2002)
(revision 6, February 2003)

Contents

Acknowledgements

This report was originally developed for the United State Agency for International Development (USAID) Environmental Policies and Institutions for Central Asia (EPIC) Program which was a task order under the Environmental Policy and Institutional Strengthening Indefinite Quantity Contract (EPIQ) managed by International Resources Group (Task Order No. 813, USAID Contract No. PCE-I-00-96-00002-00). The first version of these materials were developed for a set of two training seminars held in Tashkent, Uzbekistan in the summer of 1999 on water and energy management modeling in Central Asia. This book was elaborated after that and some additional tasks were added. This book is intended for use by (1) beginning optimization modelers in science and engineering; and (2) engineers and scientists performing research on the optimal use of resources for water, energy and agricultural problems. The authors acknowledge the support of the International Research and Exchanges Board for support to Dr. Savitsky during his extended visit in Austin during Spring 2001. The authors would like to note the various contributions of our colleagues who have assisted in reviewing sections of the material and proposed various changes: Olga Tikhanova, Alex Meeraus, Nurlan Zhangarin, and Amirkhan Kenshimov. These materials have also been used in a graduate course on Water Resource Planning and Management at the University of Texas at Austin and the comments and feedback of the students in these classes is gratefully appreciated.

1. Introduction

1.1 Systems Approach

Planning, management and design are a critical element of sustainable economic development and expansion. In the process of planning and design there is a need to critically analyze the true economic costs, benefits and environmental consequences of projects. A lack of this analysis can often lead to a level of design quality which falls far short of optimal with respect to the utilization of scarce economic and natural resources and will not improve the ecological balance of systems in general.

More recently, population shifts and industrial changes and political changes have led political leaders and planners to the conclusion that unlimited expansion and development are no longer the primary objectives in social and economic systems planning and design. The single-objective, single-purpose, single-facility project approach to solving problems that was so common in the past is unacceptable today and has been replaced by multi-objective, multi-purpose, multi-facility solutions at a large scale which must be not only technically feasible but socially, environmentally, economically, and politically feasible as well. In most planning situations it is hard to see how all of these disparate components can be combined into system designs which meet prescribed and sometimes conflicting objectives and constraints imposed on a project.

Systems analysis can aid in identifying those likely situations where a minimum investment of funds and energies will produce maximum gains in terms of resource allocations, economic development and environmental welfare. Generally speaking, systems analysis is the art and science of disassembling complex phenomena into smaller, isolated, more readily understood, subsystems and analyzing the interactions between the subsystems and between the subsystems and the larger environment (Churchman, 1968). This is a natural human process: examining a complex process by directing attention to the component parts and the relationships between the components. Using systems analysis we can focus on the functioning of the components under the various conditions to which the system may be subjected. In many situations, by focusing on the relationships and interactions between the components of complex systems, systems analysis can provide a means of sorting through the myriad of possible solutions to a problem and narrowing the search to a few potentially optimal ones in addition to determining and illustrating the consequences of these alternatives and the tradeoffs between conflicting objectives. The central method used in water resource systems analysis is to couple the descriptions of physical and socioeconomic systems through the use of mathematical models.

A system is a collection of components and their interrelationships forming an entity (e.g., a river basin) which is acted upon by external forces, influences or inputs (precipitation) and produces a specific effect or output (streamflow). That is, a system is a set of objects which transforms an input into an output, the exact output produced depending on certain system properties or parameters (e.g., soil types, vegetation, topography). This transformation depends upon the parameters of the system and the design policies imposed on it. Figure 1.1.1 displays a systems representation of a general

groundwater system and illustrates the ideas of inputs, outputs, policy variables, and parameters.

Systems analysis involves the construction and linkage of mathematical models of the physical and economic subsystems associated with resource allocation systems. The purpose of constructing these models is to aid engineers, planners and decision makers in identifying and evaluating alternative designs and to determine which ones meet project objectives in an efficient manner. These mathematical models are able to predict a system's response to different design alternatives and conditions. The models are a set of mathematical expressions (partial or ordinary differential or algebraic equations) describing the physical, biological, chemical, and economic processes which take place in the system.

Most systems models are based on statements of basic conservation laws (mass, energy, and momentum), but they can also be empirical or statistical. Systems analysis models are generally broken down into two categories: *simulation* models and *optimization* models.

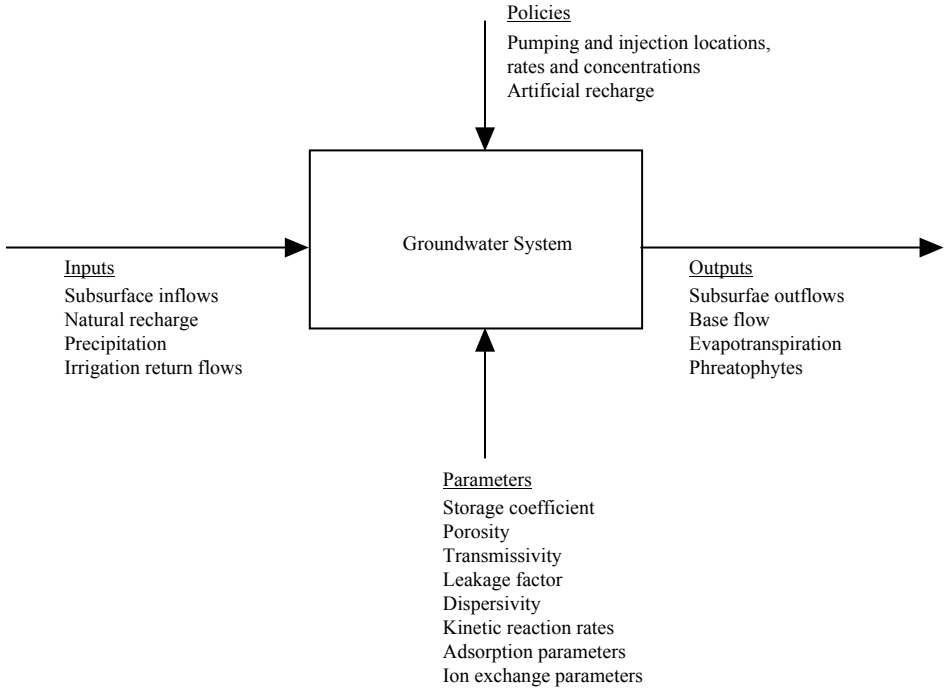


Figure 1.1.1. General diagram of a groundwater system showing inputs, outputs, parameters, and policies. (Adapted from Willis and Yeh, 1987, Fig. 1.5.)

1.2 Simulation and Optimization Models

Simulation models are used to predict a system's response to a given design configuration with great accuracy and detail, and to identify the probable costs, benefits, and impacts of a project. That is, the simulation model predicts the outcome of a single, specified set of design or policy variables. However, the space of possible design and policy variable values is, in general, infinite. Simulation

models, while important tools for managing systems, do not identify or narrow the search for optimal policies or designs for a problem; they provide only localized information regarding the response of the system to one particular design alternative at a time. Separate simulation model runs are required for each design or policy alternative considered. In many situations the number of alternative designs is sufficiently large to preclude simulating each alternative and some other method is normally used to narrow the field of search.

Optimization models provide a means of reducing the number of alternatives which need to be simulated in detail, i.e., screening them. These models search the space of possible design variable values and identify an optimal design and/or operating policy for a given system design objective and set of constraints. The sensitivity of the optimal solution to changes in the model parameters can be readily determined and tradeoffs between several conflicting objectives can also be calculated with most optimization models. These models are usually extensions of simulation models and include as unknowns the design or operating variables (decision variables) of each alternative. These models include relationships which describe the state variables and costs or benefits of each alternative as a function of the decision variables. Constraints are also included in the models to restrict the values of the design or state variables. Optimization models are generally used for preliminary evaluation or screening of alternatives and to identify important data needs prior to extensive data collection and simulation modeling activities.

1.3 Model Building Process

The process of developing the mathematical simulation and optimization models which represent the system under investigation consists of several steps. Referring to Figure 2, the first step, problem identification, is to identify the important elements of the system which pertain to the problem at hand and the interactions between the components. That is, a general outline and purpose of the model must be established. The analyst will need to identify the appropriate type of model for the system and the degree of accuracy needed given the time and resources available for modeling. Generally the simplest model with the least number of parameters which will produce reliable results in the time available is preferred. In the next step, conceptualization and development, the mathematical description of the relationships identified previously are established. In this step appropriate computational techniques are also determined and implemented for the problem. Calibration of the mathematical model is then performed to determine reliable estimates of the model parameters. In this procedure the model outputs are compared with actual historical or measured outputs of the system and the parameters are adjusted until the model predicted and the measured values agree. Then a model verification exercise is carried out in which an independent set of input data is used in the model and the predicted results are compared with measured outputs and if they are found to agree the model is considered to be verified and ready for use in simulation or optimization.

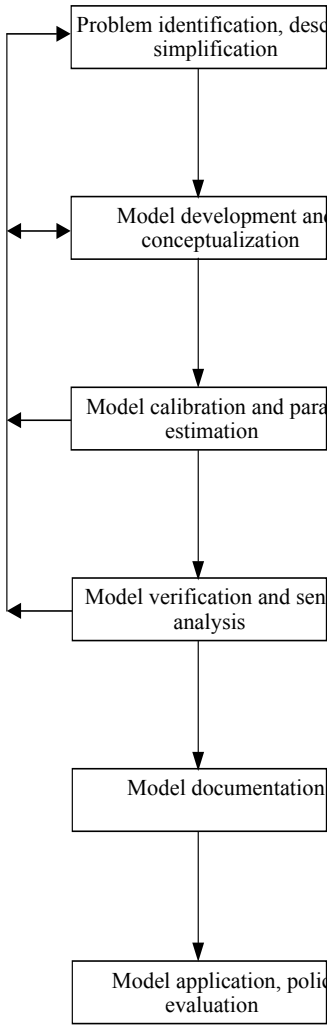


Figure 1.2.1. General diagram of the steps in the model building process. (Adapted from Loucks et al., 1981, Fig. 9.1.)

1.4 Book Organization

This book is intended for use by beginning optimization modelers in science and engineering; and engineers and scientists performing research on the optimal use of resources for water, energy and agricultural problems. Examples of optimization models from several technical areas of interest are covered in Part 1: general equations solving, water resources management, agricultural management, canal design, power system design and management, heat transfer, and fluid flow. These models are presented for the purpose of introducing the reader to the possibilities of modeling these systems using optimization techniques. There are certainly many additional application areas and techniques that could be covered, but the examples selected cover a wide enough range to introduce the reader to the topic. Part 2 of the book covers many of the basics of the modeling language used in this book. That language is the General Algebraic Modeling System (GAMS) a

high-level modeling system for mathematical programming problems (Brooke et al., 1998). This section of the book is intended to provide the reader with sufficient information to construct simple models without having to read through the full language documentation.

In our work we have found that the modeling languages are more useful than the other types of modeling packages such as LINGO (Schrage, 1999) since these products do not allow the easy construction of general modeling structures such as those needed for solving differential equations. There are other high level modeling languages available that can be used for the same purpose, including AIMMS (Bisschop and Roelofs, 2001) and AMPL (Fourer et al., 1999).

References

Bisschop, J., and M. Roelofs, AIMMS: The Users Guide, Paragon Decision Technology, 2001.

Brooke, A., D. Kendrick, A. Meeraus, and R. Raman, GAMS A Users Guide, GAMS Development Corporation, Washington D.C., 1998.

Churchman, C. W., The Systems Approach, Dell Publishing Co., New York, 1968.

Fourer, R., D. M. Gay, B. W. Kernighan, AMPL: A Modeling Language for Math Programming Package, Duxbury Press, 1999.

Loucks, D. P., J. R. Stedinger and D. A. Haith, Water Resources Systems Planning and Analysis, Prentice Hall, Englewood Cliffs, 1981

Schrage, L., Optimization Modeling with LINGO, Lindo Systems, 1999.

Willis, R. L. and W. W-G. Yeh, Groundwater Systems Planning and Management, Englewood Cliffs, Prentice Hall, 1987,

PART 1: RESOURCE MANAGEMENT MODELS

2. NONLINEAR ALGEBRAIC EQUATIONS

2.1 Solution of Algebraic Equations

(adapted from G. Kovalenko)

The problem of solving simultaneous equations (both linear and nonlinear) is often faced in science and engineering. GAMS is very well suited to solving algebraic equations $f(x) = 0$. The only difficulty is connected with defining all roots if there are multiple roots to an equation. For example, consider a system of two equations:

$$f_1(x) = Ax^2 + Bx + C = y' \quad (2.1.1)$$
$$f_2(x) = M x + N = y'' \quad (2.1.2)$$

where we would like to find the values of x, such that $y = 0$. These equations can be combined to form:

$$(Ax^2 + Bx + C) - (Mx + N) = y' - y'' \quad (2.1.3)$$

It is obvious that the absolute value of $(y' - y'')$ will 0 when we have found the sought-for roots. To solve the problem, let us minimize the value $(y' - y'')^2$ by finding the optimal value of x. Consider the concrete example

$$(10x^2 + 10x + 10) - (-10x + 100) = y' - y'' \quad (2.1.4)$$

A simple GAMS model to solve this problem is

```
VARIABLES x, y1, y2, obj;

EQUATIONS Eq1, Eq2, Objective;

Eq1.. y1 =E= 10*x*x+10*x+10;
Eq2.. y2 =E= -10*x+100;
Objective.. obj =E= (y1-y2)*(y1-y2);

MODEL Eq /ALL/

SOLVE Eq USING NLP MINIMIZING obj;

FILE res /Eq1.txt/
PUT  res;
put  "Solution x =  ", put x.l, put /;
```

In this model 4 variables are defined, *x, y1*, *y2*, and *obj*. All these variables are allowed to take on any vales, positive or negative. In addition, there are 3 equations defined in the model, *Eq1*,

Eq2 and *Objective*. *Eq1* is the same as $f_1(x)$ above and *Eq2* is the same as $f_2(x)$. The model is comprised of all the defined equations, as shown in the *model* statement. Since $f_2(x)$ is a nonlinear (quadratic) equation, the model is solved using a nonlinear programming solver, as specified in the *solve* statement.

In this model a very simple output file is generated with only a single line of text. The output file is first defined as *Eq1.txt* using the *file* command and then the model output is directed to that file using the *put* commands. In GAMS, several output file can be defined if desired and output directed to each file.

Note that in the model the exponentiation operator (**) is avoided. This is because `x**n` is calculated inside GAMS as `exp[n*log(x)]`. This operation is not defined when `x` takes on negative values (which is will, see below), and an error will result. In this case, if the exponent is known to be an integer, a function call, `power(x,n)`, can be used instead.

Figure 2.1.1 shows a plot of the solution in this case. The first two lines are functions with points of intersection to be found. The third line is the distance between the functions depending on x which is a plot of the variable *obj*. It can be seen that this system has two roots: x' =-4.162 and x'' = 2.162.

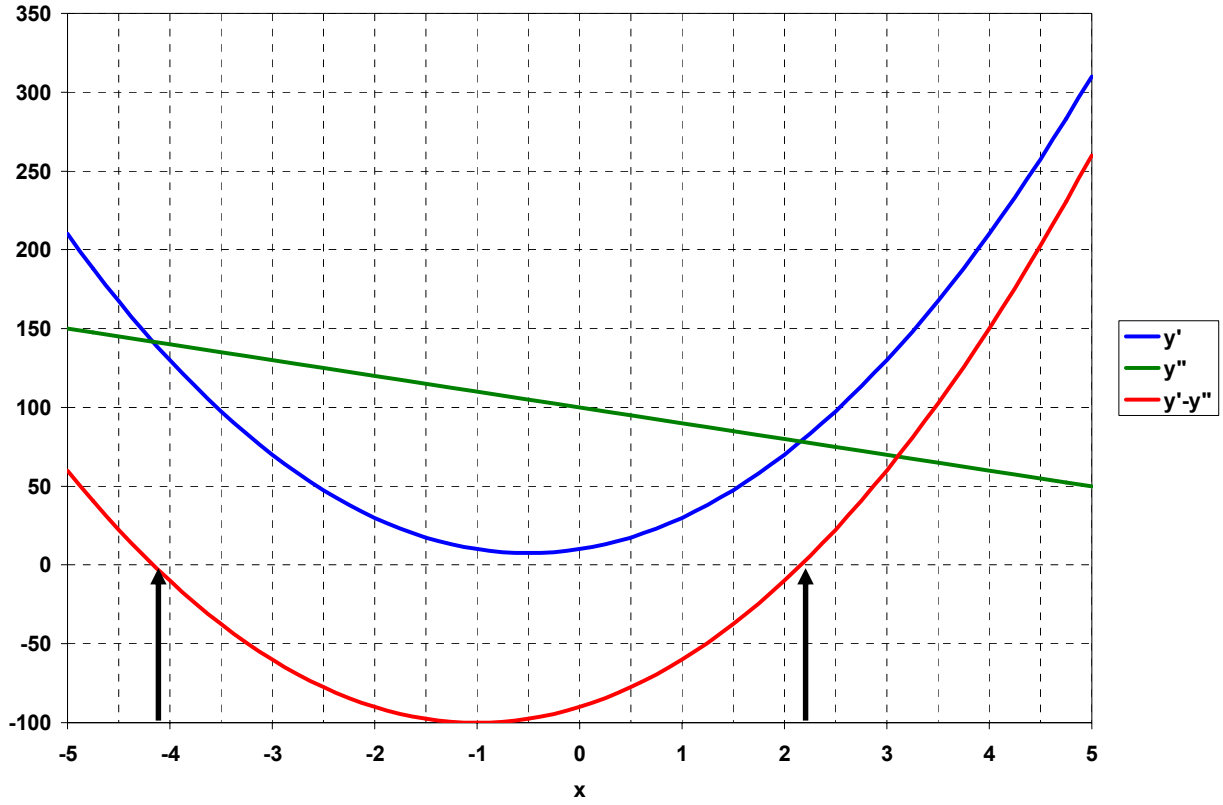


Fig. 2.1.1. Chart for the Task of Solution of Equations

When the above model is solved by GAMS the solution x' =2.162 is obtained. In order to obtain the solution x'' = -4.162 you must add the line

```
x.L  =        -3.0;
```

to the model after the *variable* statement. This will change the initial guess for x. Any initial guess of $x < -1.0$ will yield the solution $x^* = -4.162$ and any initial guess of $x > -1.0$ will yield the solution $x^*=2.162$.

The model described above can be generalized somewhat so that the data are more easily identified and available for editing. This is done using the GAMS notion of *scalars* and the coefficients of the equations are given names corresponding to those in Equations 2.1.1 and 2.1.2. Thus, the model is rewritten as

```
SCALAR A /10/, B /10/, C /10/, M /-10/, N /100/;

VARIABLES x, y1, y2, obj;

EQUATIONS Eq1, Eq2, Objective;

Eq1.. y1 =E= A*x*x+B*x+C;
Eq2.. y2 =E= M*x+N;
Objective.. obj =E= (y1-y2)*(y1-y2);

MODEL Eq /ALL/

SOLVE Eq USING NLP MINIMIZING obj;

FILE res /Eq1.txt/
PUT  res;
put  "Solution x =  ", put x.l, put /;
```

2.2 Method of Least Squares for Arbitrary Functions

(adapted from O.N. Tikhonova)

Suppose that three independent variables (x_1, x_2, x_3) are retated in a complex function with a fourth, dependent variable (y). The function relating these four variables has been defined on the basis of a physical phenomenon under study as

$$y = a(x_1)^2 - \frac{b}{x_3} - \frac{c}{x_2} + e^{-y^2} \tag{2.2.1}$$

where a, b, and c are unknown coefficients. Measurements of the four variables have been made at 8 different times. The difference (or residual) between the model value y (given in Equation 2.2.1 and which depends on the unknown parameter values a, b, and c) and the observed value $\hat{y}(t)$ can be written as

$$e(t) = y(t) - \hat{y}(t) \tag{2.2.2}$$

Numerical values of the coefficients can be determined by minimizing the squared residuals

$$Minimize \sum_{t=1}^{8}[e(t)]^2 \qquad (2.2.3)$$

In this case, suppose we have the following data

Table 2.2.1. Data for Least Squares Estimation.

	Time							
	1	2	3	4	5	6	7	8
x_1	2	3	3	3	5	5	6	7
x_2	30	60	70	60	80	90	100	100
x_3	1	6	7	3	5	9	8	17
$\hat{y}(t)$	10	20	20	20	40	50	60	70

In this case we will need to introduce the notion of *sets* and *parameters* into the GAMS model. Sets are a way of defining indices in GAMS and we need one for time in this problem. Consider the following model developed in the GAMS language:

```
SETS t  / 1, 2, 3, 4, 5, 6, 7, 8 /;

PARAMETER x1(t) /1 2, 2 3, 3 3, 4 3, 5 5, 6 5, 7 6, 8 7/;
PARAMETER x2(t) /1 30, 2 60, 3 70, 4 60, 5 80, 6 90, 7 100, 8 100/;
PARAMETER x3(t) /1 1, 2 6, 3 7, 4 3, 5 5, 6 9, 7 8, 8 17/;
PARAMETER y_hat(t) /1 10, 2 20, 3 30, 4 20, 5 40, 6 50, 7 60, 8 70/;

VARIABLES
a, b, c, y(t), e(t), obj;

EQUATION mod(t), residual(t), objective;

mod(t).. a*x1(t)*x1(t)-b/x3(t)-c/x2(t)+EXP(-y(t)*y(t)) =E= y(t);

residual(t).. e(t) =E= y(t)-y_hat(t);

objective.. obj=E=sum(t,power(e(t),2));

MODEL Leastsq / ALL /;

SOLVE Leastsq USING NLP MINIMIZING obj;

FILE res /Eq2.txt/
PUT res;
PUT "      t                    x(1,t)    x(2,t)       x(3,t)       y(t)
y_hat(t)"/;
LOOP((t),PUT t.TL, x1(t), x2(t), x3(t), y.L(t), y_hat(t)/;);
PUT /"          a          b           c"/;
PUT a.L, b.L, c.L;
```

In this model an output file is generated which prints a table of values in rows and columns. The model output is directed to that file using the *put* commands. The header for the table is printed first and then a *loop* cycles over each value of $x_1(t)$, $x_2(t)$, $x_3(t)$, $y(t)$, and $\hat{y}(t)$ printing one line for each value of t. In GAMS, several output file can be defined if desired and output directed to each file.

The results are:

```
t                x1(t)      x2(t)      x3(t)      y(t)     y_hat(t)
1                  2.0       30.0        1.0       10.2       10.0
2                  3.0       60.0        6.0       26.5       20.0
3                  3.0       70.0        7.0       24.5       30.0
4                  3.0       60.0        3.0       20.5       20.0
5                  5.0       80.0        5.0       41.5       40.0
6                  5.0       90.0        9.0       43.0       50.0
7                  6.0      100.0        8.0       55.8       60.0
8                  7.0      100.0       17.0       75.6       70.0

      a          b           c
   1.33      36.27    -1234.14
```

2.3 Exercises

1. Add the following constraint:

$y \geq 100 - 50x$

to the model of section 2.1 and resolve. Verify that you still obtain the same solution. What has happened to the "feasible" region of possible solutions by adding this constraint?

2. Modify the output of the model of section 2.2 to print the percent error in the model calculations, where the error is defined as

[y_hat(t) - y(t)]/y_hat(t)*100

3. WATER MANAGEMENT IN RIVER SYSTEMS

3.1 Optimal Allocation of Water to Users

(adapted from Loucks et al., 1981)

Consider a situation where a total quantity of water R is to be allocated to a number of different uses. Let the quantity of water allocated to each use be denoted by $x_i, i = 1,...,I$. The objective is to determine the allocations such that the total net benefits from all uses is maximized. Consider an example where there are three uses $I = 3$, as shown in Figure 3.1.

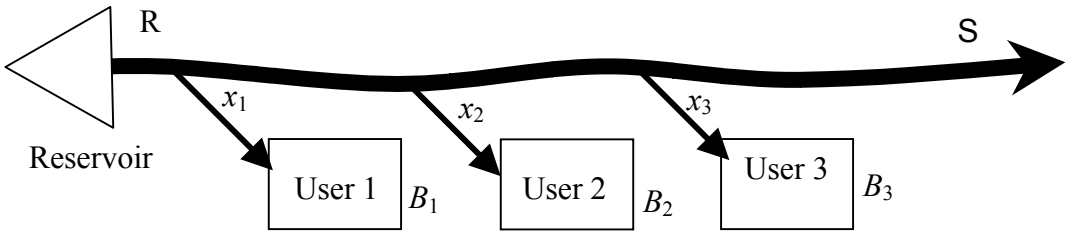


Figure 3.1. Reservoir release example.

The net-benefit resulting from an allocation of x_i to use i is (see Fig. 3.2)

$$B_i(x_i) = a_i x_i - b_i x_i^2 \qquad i = 1,2,3 \tag{3.1.1}$$

where a_i and b_i are given positive constants.

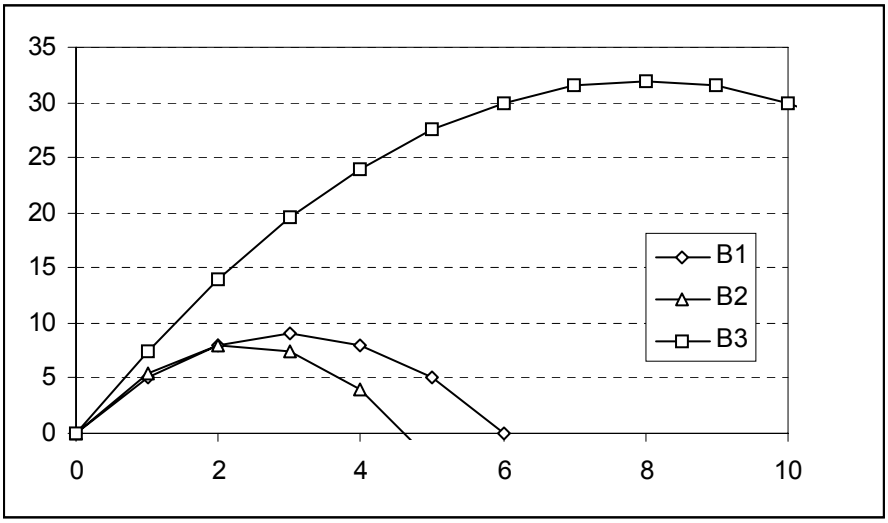


Figure 3.2. Net-benefit function for user *i*.

The three allocations x_i are unknown decision variables. The values that these variables can take on are restricted between 0 (negative allocation is meaningless) and values whose sum, $x_1 + x_2 + x_3$, does not exceed the available supply of water R minus a required instream flow S. The optimization model is

$$\underset{\vec{x}}{\textit{maximize}} \sum_{i=1}^{3}(a_i x_i - b_i x_i^2) \tag{3.1.2}$$

$$\textit{subject to} \quad \sum_{i=1}^{3} x_i + S - R = 0 \tag{3.1.3}$$

The GAMS code to solve this problem is

```
SETS      i / 1, 2, 3/

SCALAR    r RELEASE   /10.0/;

PARAMETER
a(i) /1 6.0, 2 7.0, 3 8.0/
b(i) /1 -1.0, 2 -1.5, 3 -0.5/;

VARIABLES obj OBJECTIVE;

POSITIVE VARIABLES x(i) USE, s DOWNSTREAM FLOW;

S.lo=2.0;

EQUATIONS objective, cap;

objective..obj =E= SUM(i,a(i)*x(i)+b(i)*x(i)**2);

cap..sum(i,x(i))+s-r =E= 0.0;

MODEL user /ALL/ ;

SOLVE user USING NLP MAXIMIZE obj ;

FILE res /WaterUser.txt/
PUT res
PUT 'Release         ', PUT r,     PUT /
PUT 'Downstream flow ', PUT s.l,   PUT /
PUT 'Objective       ', PUT obj.l, PUT //
PUT 'i                 x(i) ' PUT /
loop( (i), PUT i.TL,  PUT x.l(i), PUT /)
PUT //, 'd(obj)/dr = ', PUT cap.m, PUT //
```

In this model, we know that the allocations to the water users (x_i) are non-negative, so we can define them as *positive variables* which allows the use of the *exponentiation* operator (**) in the benefit function. Also, note the use of comments in the file. This makes it easier for someone else to understand the model. Two other GAMS operators are used in this model. First is the '*.lo*' operator, which is used to set a lower bound on the variable *s* of 2.0 units of flow. The same effect could be achieved by defining a new equation as a lower bound constraint

```
EQUATIONS slo;
slo..s =G= 2.0;
```

However, computational savings are realized if the '*.lo*' operator is used. Upper bounds on variables can also be defined using the '*.up*' operator with a variable. Second, the *sum* operator is used in the objective function and in the resource constraint allowing us to construct the GAMS equations in the same form as the mathematical description of the model (Equations 3.1.2 and 3.1.3).

The result for a release of $R = 10$ is

```
Release               10.00
Downstream flow        2.00
```

```
Objective                41.41

i                     x(i)
1                      1.55
2                      1.36
3                      5.09


d(obj)/dr =            2.91
```

Figure 3.3 illustrates the result of running the model for various values of the reservoir release, r. Note that the water users continue to receive additional water up to the release of $r = 15.33$. After that point users are experiencing declining revenue from the use of the water and they stop demanding additional units and the marginal value of additional release of water $\lambda = d(obj)/dr = 0$. This marginal value (Lagrange multiplier) is printed in the output file using the '*.m*' operator.

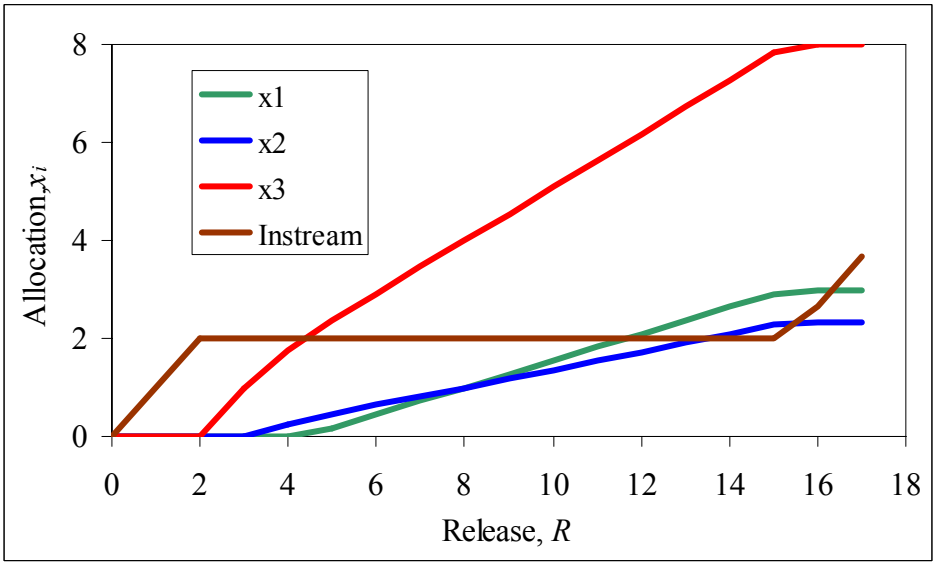


Figure 3.3. Release allocation to water users and downstream flow.

3.2 Optimal Management of a Single Reservoir

Reservoir management is an important task for water managers throughout the world. Models are often constructed to simulate or optimize the performance of reservoir operations a well as to design reservoir or the associated facilities (spillways, etc.). In this section we consider two common tasks of reservoir modeling: (1) determining various coefficients of the functions that describe reservoir characteristics; and (2) the optimal mode of reservoir operation (storage volumes, elevations and releases) while satisfying downstream water demands.

3.2.1 Estimating a Reservoir Storage – Elevation Relationship Using the Method of Lease Squares

The objective for the first task is to find the coefficients (*a*, and *b*), for the function

$$s = a(h - h_0)^b \tag{3.2.1.1}$$

where *s* is the storage volume in the reservoir (L^3), *h* is the elevation of the water surface in the reservoir (L) and h_0 is the elevation of the tailwater at the outlet of the reservoir (L). In the estimation procedure, we use the method of least squares as indicated in the Section 2.2. As in that case, the difference (or residual) between the model value *s* (given in Equation 3.2.1.1 and which depends on the unknown parameter values *a*, and *b*) and the observed value $\hat{s}$ can be written as

$$e_i = s_i - \hat{s}_i = a[\hat{h}_i - h_0]^b - \hat{s}_i, \qquad i = 1,2,...,I \tag{3.2.1.2}$$

where

i is the index of measurements (observed storage volumes and elevations);
I is the total number of measurements;
s_i is the *i*-th calculated storage volume in the reservoir;
$\hat{s}_i$ is the *i*-th observed storage volume in the reservoir; and
$\hat{h}_i$ is the *i*-th observed water elevation in the reservoir.

Numerical values of the coefficients are determined by minimizing the squared residuals

$$Minimize \sum_{i=1}^{I} (e_i)^2 \tag{3.2.1.3}$$

As an example, the volume and elevation data for Toktogul reservoir on the Naryn River in Kyrgyzstan are given the Table 3.2.1.1 and plotted in Figure 3.2.1.1.

Table 3.2.1.1 . Storage and Surface Area Versus Water Surface Elevation Data for Toktogul Reservoir.

Item	Volume (mln m^3)	Elevation (m)	Surface area (km^2)
i	*S*(*i*)	*h*(*i*)	*A*(*i*)
1	0	759	0
2	78	767	9.75
3	118	771	10
4	321	783	16.917
5	405	786	27.999
6	779	795	41.555
7	1204	802	60.713
8	1885	810	85.124
9	2297	814	102.997
10	3921	827	124.922
11	5585	838	151.271
12	5750	839	164.984
13	7717	850	178.817
14	8690	855	194.596
15	10336	863	205.747
16	11198	867	215.495
17	13003	875	225.622
18	14676	882	238.997
19	16961	891	253.886
20	19458	900	277.441

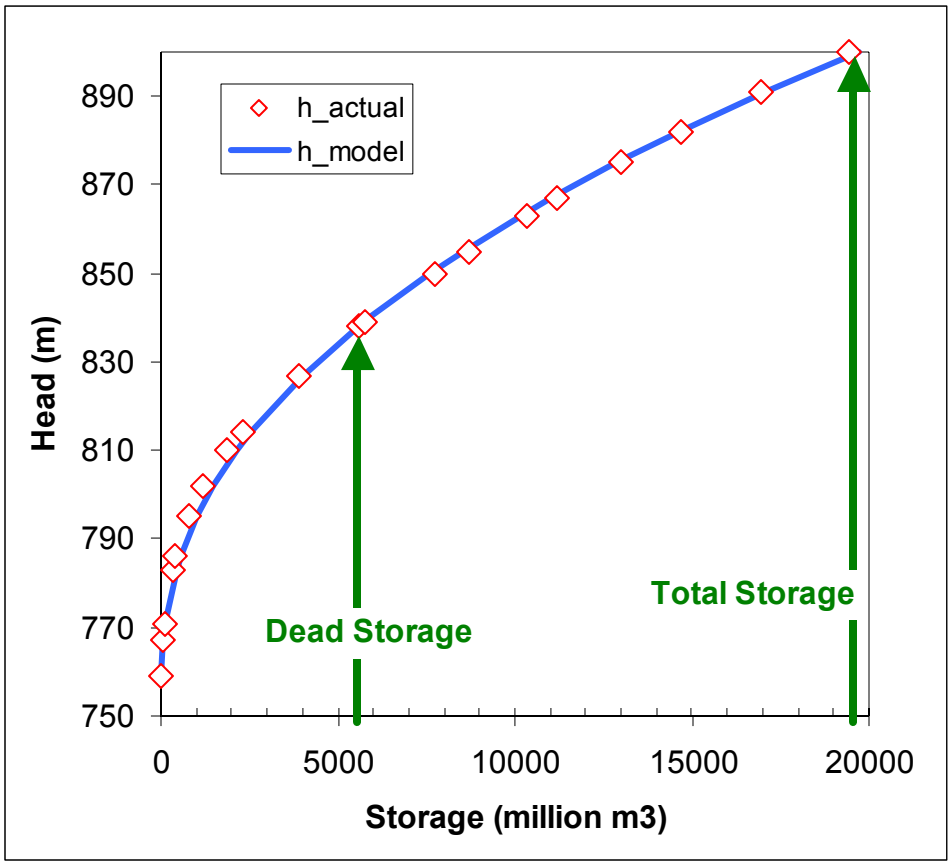


Figure 3.2.1.1. Storage versus water surface elevation relations for Toktogul reservoir.

The GAMS code for this model is shown below. Note that upper bounds on the coefficients have been added to prevent exponentiation overflow.

```
SETS
i   /1*20/

SCALAR h0  Minimum Elevation  /758/

PARAMETER
h(i) /
1          759
2          767
3          771
4          783
5          786
6          795
7          802
8          810
9          814
10         827
11         838
12         839
13         850
14         855
15         863
```

```
16        867
17        875
18        882
19        891
20        900
/;

PARAMETER
S_hat(i) /
1             0
2            78
3           118
4           321
5           405
6           779
7          1204
8          1885
9          2297
10         3921
11         5585
12         5750
13         7717
14         8690
15        10336
16        11198
17        13003
18        14676
19        16961
20        19458
/;

POSITIVE VARIABLES
a, b;

b.up=100;
a.up=100;

VARIABLES
e(i), obj;

EQUATIONS residual(i), objective;

residual(i).. e(i) =E= a*(h(i)-h0)**b - S_hat(i);

objective.. obj=E=sum(i,power(e(i),2));

MODEL hvs /ALL/;

SOLVE hvs USING  NLP MINIMIZING obj;

FILE res /River1a.txt/;
PUT res
PUT "Coefficients(a and b) in formula S = a*(h-h0)**b"/ ;
PUT "a = ", a.L, "   b = ", b.L//;
PUT "No.            Elevation   Volume(real)  Volume(calc)"/;
LOOP(i,PUT i.TL,  h(i), S_hat(i), ((a.L*(h(i)-h0)**b.L))/;);
```

The results are

```
Coefficients(a and b) in formula S = a*(h-h0)**b
```

a = 0.36 b = 2.20

No.	Elevation	Volume(real)	Volume(calc)
1	759.00	0.00	0.36
2	767.00	78.00	45.54
3	771.00	118.00	102.29
4	783.00	321.00	431.26
5	786.00	405.00	553.40
6	795.00	779.00	1021.84
7	802.00	1204.00	1496.13
8	810.00	1885.00	2160.78
9	814.00	2297.00	2543.50
10	827.00	3921.00	4026.47
11	838.00	5585.00	5575.47
12	839.00	5750.00	5729.98
13	850.00	7717.00	7583.03
14	855.00	8690.00	8519.56
15	863.00	10336.00	10142.62
16	867.00	11198.00	11012.33
17	875.00	13003.00	12869.53
18	882.00	14676.00	14624.88
19	891.00	16961.00	17062.84
20	900.00	19458.00	19707.16

A plot of the function versus the measured data are shown in Figure 3.2.1.1. The same procedure can be used to develop the function for surface area as a function of storage volume. The results are shown in Figure 3.2.1.2.

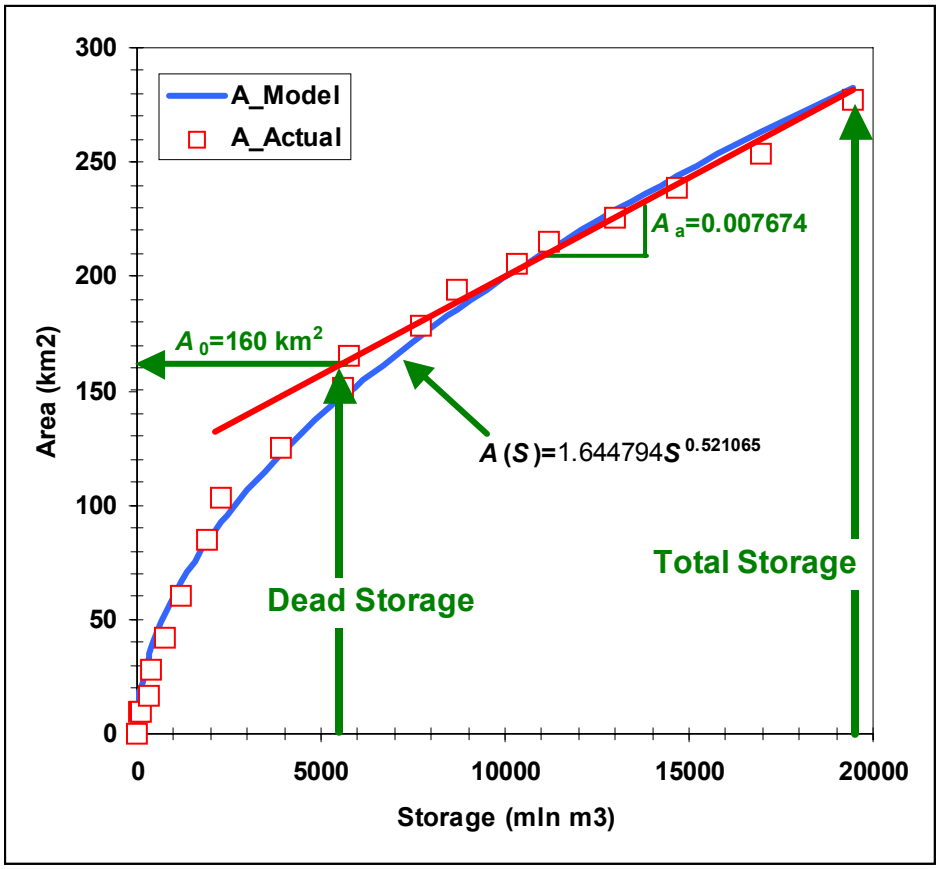


Figure 3.2.1.2. Storage versus reservoir surface area relations for Toktogul reservoir.

The model can be modified if the user would like to estimate

$$h = a(s)^b + h_0 \tag{3.2.1.4}$$

In this case, the difference (or residual) between the model value h (given in Equation 3.2.1.4 and which depends on the unknown parameter values a, and b) and the observed value $\hat{h}$ can be written as

$$e_i = h_i - \hat{h}_i = a(\hat{s}_i)^b + h_0 - \hat{h}_i, \qquad i = 1,2,...,I \tag{3.2.1.5}$$

The necessary modifications to the model are in the residual equation and the output:

```
residual(i).. e(i) =E= a*S_hat(i)**b + h0 -h(i);
.
.
.
PUT "No.            Elevation   Volume(real)  Elevation(calc)"/;
LOOP(i,PUT i.TL,  h(i), S_hat(i), ((a.L*S_hat(i)**b.L+h0))/;);
```

The results for the Toktogul reservoir example are:

```
Coefficients(a and b) in formula h = a*S**b+h0
a =         1.91   b =         0.43

No.            Elevation   Volume(real)  Elevation(calc)
1                 759.00        0.00      758.00
2                 767.00       78.00      770.70
3                 771.00      118.00      773.20
4                 783.00      321.00      781.49
5                 786.00      405.00      783.98
6                 795.00      779.00      792.52
7                 802.00     1204.00      799.72
8                 810.00     1885.00      808.69
9                 814.00     2297.00      813.24
10                827.00     3921.00      827.69
11                838.00     5585.00      839.27
12                839.00     5750.00      840.30
13                850.00     7717.00      851.53
14                855.00     8690.00      856.48
15                863.00    10336.00      864.19
16                867.00    11198.00      867.95
17                875.00    13003.00      875.33
18                882.00    14676.00      881.67
19                891.00    16961.00      889.70
20                900.00    19458.00      897.80
```

3.2.2 Reservoir Operation

In the second model, the optimal operation of the reservoir is computed given a series of inflows and downstream water demand targets. Evaporation from the reservoir surface is calculated using the relation between storage volume and surface area is needed. This function is plotted in Figure 3.2.1.2. A linear approximation to the surface area – volume relation is used between the dead storage and total storage levels (Loucks et al., 1981). The model is

$$\textit{Minimize} \quad \sum_{t=1}^{T} [r_t - d_t]^2 \tag{3.2.2.1}$$

subject to

$$(1+\alpha_t)s_t = (1-\alpha_t)s_{t-1} + q_t - r_t - \beta_t \qquad t = 1,...,T \tag{3.2.2.2}$$

$$s_t \le K \qquad t = 1,...,T \tag{3.2.2.3}$$

$$\alpha_t = \frac{A_a e_t}{2} \qquad t = 1,...,T \tag{3.2.2.4}$$

$$\beta_t = A_0 e_t \qquad t = 1,...,T \tag{3.2.2.5}$$

where:

s_t	Storage at the end of time period t, (L^3);
s_{t-1}	Storage at the beginning of time period t, (L^3);
q_t	Inflow volume during time period t, (L^3);
r_t	Release volume time period t, (L^3);
e_t	Evaporation rate, (L);
d_t	Demand, (L^3); and
K	Capacity (total storage) of reservoir, (L^3)

The GAMS code for the model is shown below.

```
SCALAR K      /19500/;
SCALAR S_min  /5500/;
SCALAR beg_S  /15000/;

SETS
t   / t1*t12/;

$include River1B_Q_Dry.inc
$include River1B_D.inc
$include River1B_Evap.inc

POSITIVE VARIABLES
S(t), R(t);

S.UP(t)=K;
S.LO(t)=S_min;

VARIABLES
obj;

EQUATIONS  objective, balance(t);
```

```
objective.. obj =E= SUM(t,power((R(t)-D(t)),2));

balance(t).. (1+a(t))*S(t) =E= (1-a(t))*beg_S $(ord(t) EQ 1)
                             + (1-a(t))*S(t-1)$(ord(t) GT 1)
                             + Q(t) - R(t)- b(t);

MODEL Reservoir / ALL /;
SOLVE Reservoir USING NLP MINIMIZING obj;

FILE res /River1b.txt/;
PUT res
PUT "                  (mln.m3)     (mln.m3) (mln.m3)    (mln.m3)"/;
PUT "                 Storage       Input   Release     Demand"/;
PUT "t0          ", beg_S/;
LOOP(t,PUT t.TL, S.L(t), Q(t), R.L(t), D(t) /;);
```

Note that in this model we have used a couple of new GAMS operators. First, the '*$include*' operator is used to copy the contents of 3 data files containing:

(1) inflow data, *River1B_Q_Ave.inc:*

```
PARAMETER
Q(t) inflow (million m3)
* normal
/
t1  426
t2  399
t3  523
t4  875
t5 2026
t6 3626
t7 2841
t8 1469
t9  821
t10 600
t11 458
t12 413
/;
```

(2) demand data, *River1B_D.inc*:

```
Parameter
D(t) demand (million m3)
/
t1  1699.5
t2  1388.2
t3  1477.6
t4  1109.4
t5   594.6
t6   636.6
t7  1126.1
t8  1092.0
t9   510.8
t10  868.5
t11 1049.8
t12 1475.5
/;
```

and (3) evaporation data, *River1B_Evap.inc*:

```
Parameter
a(t) evaporation coefficient
/
t1  0.000046044
t2  0.00007674
t3  0.000180339
t4  0.000391374
t5  0.000602409
t6  0.000648453
t7  0.000656127
t8  0.000548691
t9  0.0003837
t10 0.000145806
t11 0.000103599
t12 0.000053718
/;

Parameter
b(t) evaporation coefficient
/
t1   1.92
t2   3.2
t3   7.52
t4  16.32
t5  25.12
t6  27.04
t7  27.36
t8  22.88
t9  16.00
t10  6.08
t11  4.32
t12  2.24
/;
```

into the model at the specified locations. This allows easy input and maintenance of data with a spreadsheet and the ability to quickly change complex input conditions from one case to another. This is illustrated below where the results of running the model with the file *River1B_Q_Dry.inc*

```
PARAMETER
Q(t) inflow (million m3)
* dry
/
t1   375
t2   361
t3   448
t4   518
t5  1696
t6  2246
t7  2155
t8  1552
t9   756
t10  531
t11  438
t12  343
/;
```

as the inflow condition instead of the previously used average inflow condition.

Second, the '*$*' or '*conditional*' operator is used in the '*balance*' equation in conjunction with the '*ord*' operator to pick out the initial time period and make sure that the initial storage volume '*beg_s*' is used as the previous month's storage. The effect of this is that when the condition after the '*$*' is true then the value before the '*$*' sign is used, otherwise it is omitted. The condition is that the ordinal value of the index of the set *t* must be 1 for the condition to be true in the first case or greater than 1 in the second case.

The results from running the model using the average inflow conditions are:

	(mln.m3) Storage	(mln.m3) Input	(mln.m3) Release	(mln.m3) Demand
t0	15000.00			
t1	13723.26	426.00	1699.50	1699.50
t2	12728.83	399.00	1388.20	1388.20
t3	11762.29	523.00	1477.60	1477.60
t4	11502.47	875.00	1109.40	1109.40
t5	12894.05	2026.00	594.60	594.60
t6	15837.78	3626.00	636.60	636.60
t7	17503.44	2841.00	1126.10	1126.10
t8	17838.17	1469.00	1092.00	1092.00
t9	18118.57	821.00	510.80	510.80
t10	17838.75	600.00	868.50	868.50
t11	17239.00	458.00	1049.80	1049.80
t12	16172.46	413.00	1475.50	1475.50

and the results from running the model using the dry inflow conditions are:

	(mln.m3) Storage	(mln.m3) Input	(mln.m3) Release	(mln.m3) Demand
t0	15000.00			
t1	13672.26	375.00	1699.50	1699.50
t2	12639.84	361.00	1388.20	1388.20
t3	11598.35	448.00	1477.60	1477.60
t4	10981.79	518.00	1109.40	1109.40
t5	12044.20	1696.00	594.60	594.60
t6	13609.93	2246.00	636.60	636.60
t7	14592.96	2155.00	1126.10	1126.10
t8	15013.84	1552.00	1092.00	1092.00
t9	15231.43	756.00	510.80	510.80
t10	14883.46	531.00	868.50	868.50
t11	14264.32	438.00	1049.80	1049.80
t12	13128.11	343.00	1475.50	1475.50

A comparison of the results from using the average and dry inflow conditions is illustrated in Figure 3.2.2.1.

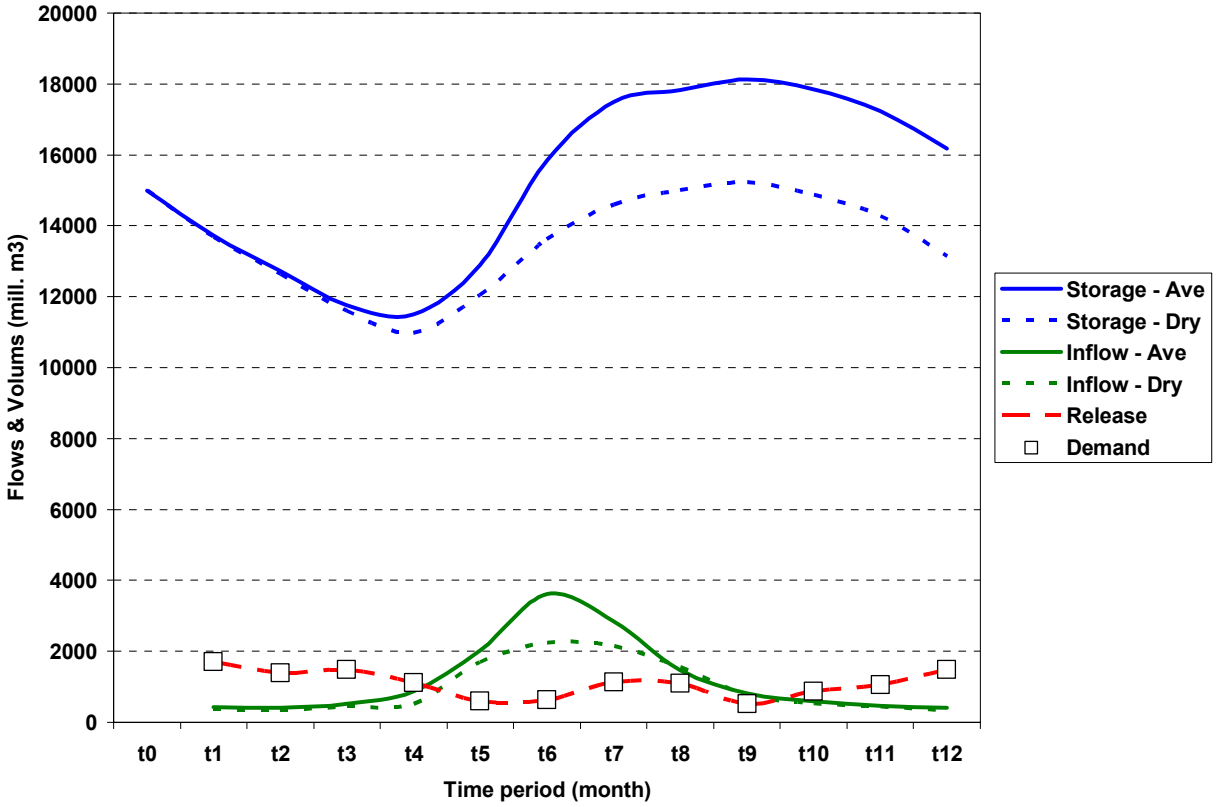


Figure 3.2.2.1. Reservoir model results using both average and dry inflow conditions.

3.3 Optimal Management of a River System

Figure 3.3.1 represents a simple river system, which includes water sources, reservoirs, consumers, and an estuary. It is desired to allocate the available water in such a way that demand is satisfied. The problem is based on graph theory, whereby conservative transport of water is carried out through a directed graph. At the nodes of the graph water resources change according to certain rules. For each type of node, the inflow (*Win*) and outflow (*Wout*) are calculated. *Win* is calculated as the sum of all releases (*Wout*) from all upstream nodes of the node under consideration. Withdrawal of water to consumers is subtracted at nodes under consideration. In this way a mass balance is maintained at each node in the system.

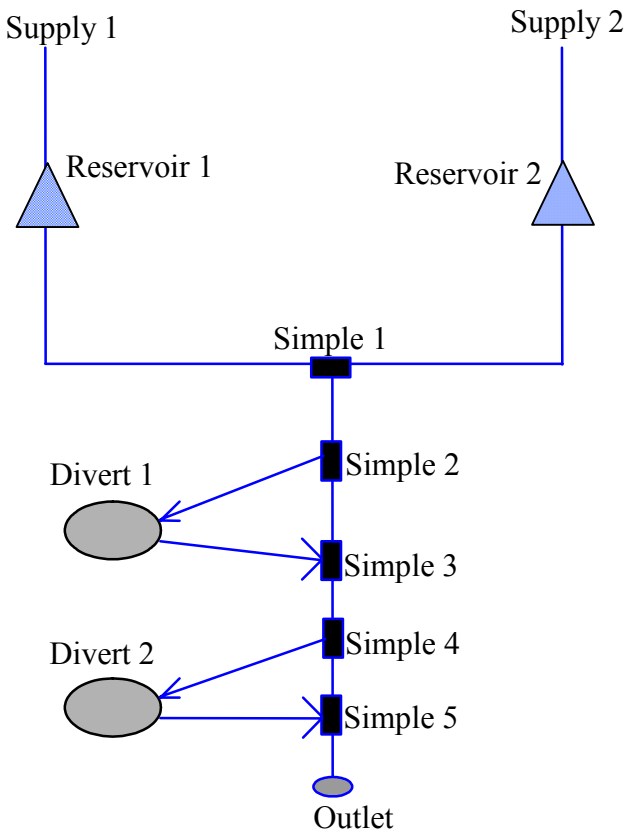


Figure 3.3.1. Structure of a River Network for the Water Management Task

Objective

$$Maximize \quad \sum_n \sum_t \frac{Divert(n,t)}{Demand(n,t)} \qquad (3.3.1a)$$

where

Divert(n,t)	Diversion of water for consumer *n* in time period *t*;
Demand(n,t)	Demand for water by consumer *n* in time period *t*; and

This objective (percentage of met demand) is a linear objective and combined with the linear constraints presented below, results in a linear program. An alternative to this objective is one in which deviations of diversions from demands are minimized, resulting in a nonlinear (quadratic) program and requiring the use of a NLP solver in GAMS.

$$Minimize \quad \sum_n \sum_t [Divert(n,t) - Demand(n,t)]^2 \qquad (3.3.1b)$$

Constraints

Simple nodes (junctions and other control nodes) $n \in nn$ and for each time period *t*, we have

$$\sum_{n \in out} R(n,t) = \sum_{n \in in} Q(n,t) + Supply(n,t) \qquad (3.3.2)$$

where

$R(n,t)$	Release from node *n* in period *t* (L^3). Releases are summed over all downstream nodes (*out*) that receive water released from the node;
$Q(n,t)$	Inflow to node *n* in period *t* (L^3). Inflows are summed over all upstream nodes (*in*) that release water to the node.
$Supply(n,t)$	Source of water at node $n \in ns$ and for each time period *t*.

Irrigation nodes $n \in nrr$, we calculate return flow as

$$R(n,t) = R_j * \sum_{n \in in} Q(n,t) \qquad (3.3.3)$$

where R_j is the return flow ratio for node *n*.

Reservoir nodes $n \in nl$, water balances in time period *t* are calculated as

$$S(n,t) = S(n,t-1) \sum_{n \in in} Q(n,t) - \sum_{n \in out} R(n,t) \qquad (3.3.4)$$

where $S(n,t)$ is the volume of water in reservoir *n* at the end of time period *t* (L^3);

The GAMS code is shown below.

```
SET n nodes
/ Supply_1, Supply_2,
  Simple_1*Simple_5,
  Divert_1, Divert_2,
  Res_1, Res_2,
  Outlet /;

ALIAS(n,n1);

SET
nn(n)     Simple nodes     /Simple_1*Simple_5/
ns(n)     Supply nodes     /Supply_1, Supply_2/
nr(n)     Water user nodes /Divert_1, Divert_2, Outlet/
nrr(n)    Irrigation nodes /Divert_1, Divert_2/
nl(n)     Reservoir nodes  /Res_1, Res_2/;

SET n_from_n(n,n1)  node n gets water from node n1 (any node)
/ Res_1.Supply_1,
  Res_2.Supply_2,
  Simple_1.Res_1,
  Simple_1.Res_2,
```

```
  Simple_2.Simple_1,
  Divert_1.Simple_2,
  Simple_3.Simple_2,
  Simple_3.Divert_1,
  Simple_4.Simple_3,
  Divert_2.Simple_4,
  Simple_5.Simple_4,
  Simple_5.Divert_2,
  Outlet.Simple_5 /;

SET n_to_nr(n,n1)  node n diverts water to node n1 (user node)
/ Simple_2.Divert_1,
  Simple_4.Divert_2,
  Simple_5.Outlet /;

SET t months /Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec /;

PARAMETER beg_S(n)    initial storage
/ Res_1  1000,
  Res_2  300 /;

PARAMETER Ret(n)    return flow coefficients
/ Divert_1  0.5,
  Divert_2  0.5,
  Outlet    0.0 /;

TABLE Supply(n,t) water supplies (m3 per sec)
          Jan   Feb   Mar   Apr  May  Jun  Jul  Aug   Sep   Oct   Nov   Dec
Supply_1  128   125   234   360  541  645  807  512   267   210   181   128
Supply_2   39    39    52   121  168  144  105   78    49    44    45    39

TABLE Demand(n,t) water demands (m3 per sec)
         Jan  Feb  Mar  Apr     May   Jun    Jul    Aug   Sep  Oct  Nov
Dec
Divert_1   0    0    0  64.5  109.8 184.4  243.7  200.9  99.5    0    0
0
Divert_2   0    0    0  13.5   15.0  22.1   26.0   24.9  13.0    0    0
0
Outlet   500  500  500   100    100   100    100    500   500  500  500
500

POSITIVE VARIABLES
Divert(n,t)   Diversions
Q(n,t)        Inflows
R(n,t)        Releases
S(n,t)        Storages;

VARIABLE obj;

* Upper bound on diversions
Divert.up(n,t) = Demand(n,t);

* Upper bound of reservoirs
S.up('Res_1' ,t    )    = 1000;
S.up('Res_2' ,t    )    = 300;

* Final storage of reservoirs
S.lo('Res_1' ,'Dec' )    = 1000;
S.lo('Res_2' ,'Dec' )    = 300;

EQUATIONS
```

```
R_no(n,t)    Simple node
R_ns(n,t)    Source node
R_nr(n,t)    Irrigation node
R_nl(n,t)    Reservoir node
R_nn(n,t)    Simple node
Objective;

* Simple node:  Release = Inflow
R_no(n,t)$(nn(n)).. R(n,t) =e= Q(n,t);

* Source node:  Release = Supply
R_ns(n,t)$(ns(n)).. R(n,t) =e= Supply(n,t);

* Irrigation node:  Release = Return Flow = Percent of Diversion
R_nr(n,t)$(nr(n)).. R(n,t) =e= ret(n)*Divert(n,t);

* Reservoir node:  Release = Mass Balance
R_nl(n,t)$(nl(n)).. S(n,t) =e= beg_S(n)$(ord(t) EQ 1)
                            +  S(n,t-1)$(ord(t) GT 1)
                            + Q(n,t)-R(n,t);

* Simple node:  Inflow = Sum of Releases from Upstream - Sum of Diversions
R_nn(n,t).. Q(n,t) =e= sum(n1$(n_from_n(n,n1)),R(n1,t))
                     - sum(n1$(n_to_nr(n,n1)),Divert(n1,t));

*Sum over Irrigation Diversions/Demands + Downstream Flow/Demand
* objective.. obj =e= sum(t,sum(n$nr(n), (Divert(n,t)/(Demand(n,t)+1e-
9))));

*Sum over Irrigation Diversions/Demands + Downstream Flow/Demand
objective.. obj=e=sum(t,sum(n$nr(n), power((Divert(n,t)-Demand(n,t)),2) ));

MODEL Lakes /all/;

*SOLVE Lakes USING LP MAXIMIZING obj;
SOLVE Lakes USING NLP MINIMIZING obj;

file res /River2.txt/
put res;
put "Node   Divert_1 Demand_1 Divert_2 Demand_2   Outlet  Demand_O"/;
loop((t),put t.TL:6, Divert.L('Divert_1',t):9.2, Demand('Divert_1',t):9.2,
                     Divert.L('Divert_2',t):9.2, Demand('Divert_2',t):9.2,
                  Divert.L('Outlet',t)  :9.2, Demand('Outlet',t)  :9.2 /;);

put /"Reservoirs  "/;
put "                Reservoir 1            Reservoir 2 "/;
put "          Inflow Storage Release  Inflow Storage Release  "/;
put "             Q-1     S-1     R-1     Q-2     S-2     R-2  "/;
loop((t),put t.TL:8, Q.L('Res_1',t):8.1, S.L('Res_1',t):8.1,
R.L('Res_1',t):8.1,
                     Q.L('Res_2',t):8.1, S.L('Res_2',t):8.1,
R.L('Res_2',t):8.1 /;);
```

Note that in this model we have used a couple of new GAMS operators:

1. *Subsets* of sets are used to partition the set of nodes into different types: simple nodes, supply nodes, water user nodes, irrigation nodes, and reservoir nodes. This facilitates the construction of a general model where different nodes serve different functions in the

network and transform water flows or other properties in different ways.

2. Alias is used to define another set of nodes exactly the same as the first set (*n* and *n1*). That way the two sets can be used to refer to two different nodes simultaneously in a two-dimensional set.
3. *Two-dimensional sets* are used to establish the topology of the network. That is, the upstream to downstream connections between nodes are identified in a two-dimensional set *n_from_n(n,n1)*. These connections are indicated by using the '*dot*' operator between two members of the two sets, e.g., *Res_1.Supply_1* where *Res_1* is from set *n* and represents the node for reservoir 1 and *Supply_1* is from set *n1* and represents the node for water supply 1.

In the example several parameters are specified that have a distinct effect on the solution:

1. The capacities of the two reservoirs in the system, 1000 and 300 units, respectively.
2. The initial volume of water stored in the reservoirs, both reservoirs at full capacity.
3. The final volume of water required to be stored in the reservoirs, both reservoirs at full capacity.
4. The instream flow requirement at the outlet, varies between 500 units in winter to 100 units in summer.

All of these factors have a distinct and significant effect on the solution and the reader can modify them and rerun the model to see the effect.

Using the nonlinear objective function, the results are:

Node	Divert_1	Demand_1	Divert_2	Demand_2	Outlet	Demand_O
Jan	0.00	0.00	0.00	0.00	500.00	500.00
Feb	0.00	0.00	0.00	0.00	500.00	500.00
Mar	0.00	0.00	0.00	0.00	500.00	500.00
Apr	64.50	64.50	13.50	13.50	100.00	100.00
May	109.80	109.80	15.00	15.00	100.00	100.00
Jun	184.40	184.40	22.10	22.10	100.00	100.00
Jul	243.70	243.70	26.00	26.00	100.00	100.00
Aug	193.27	200.90	17.27	24.90	484.73	500.00
Sep	6.00	99.50	0.00	13.00	313.00	500.00
Oct	0.00	0.00	0.00	0.00	254.00	500.00
Nov	0.00	0.00	0.00	0.00	226.00	500.00
Dec	0.00	0.00	0.00	0.00	167.00	500.00

	Reservoir 1			Reservoir 2		
	Inflow	Storage	Release	Inflow	Storage	Release
	Q-1	S-1	R-1	Q-2	S-2	R-2
Jan	128.00	382.00	746.00	39.00	168.00	171.00
Feb	125.00	107.00	400.00	39.00	107.00	100.00
Mar	234.00	0.00	341.00	52.00	0.00	159.00
Apr	360.00	0.00	360.00	121.00	0.00	121.00
May	541.00	0.00	541.00	168.00	156.00	12.00
Jun	645.00	331.70	313.30	144.00	300.00	0.00
Jul	807.00	1000.00	138.70	105.00	300.00	105.00
Aug	512.00	1000.00	512.00	78.00	300.00	78.00
Sep	267.00	1000.00	267.00	49.00	300.00	49.00
Oct	210.00	1000.00	210.00	44.00	300.00	44.00
Nov	181.00	1000.00	181.00	45.00	300.00	45.00
Dec	128.00	1000.00	128.00	39.00	300.00	39.00

The percent of the satisfied demand is illustrated in Figure 3.3.2 and the reservoir operation is shown

in Figure 3.3.3. From these figures one can see the effect of requiring that the reservoirs return to the initial condition at the end of the year, stored water is not available to meet demand in September and downstream flow demands are unmet from July to December. One interesting modification to the model would be to apply weight factors to the three different demands and see how varying the weights affects the solution.

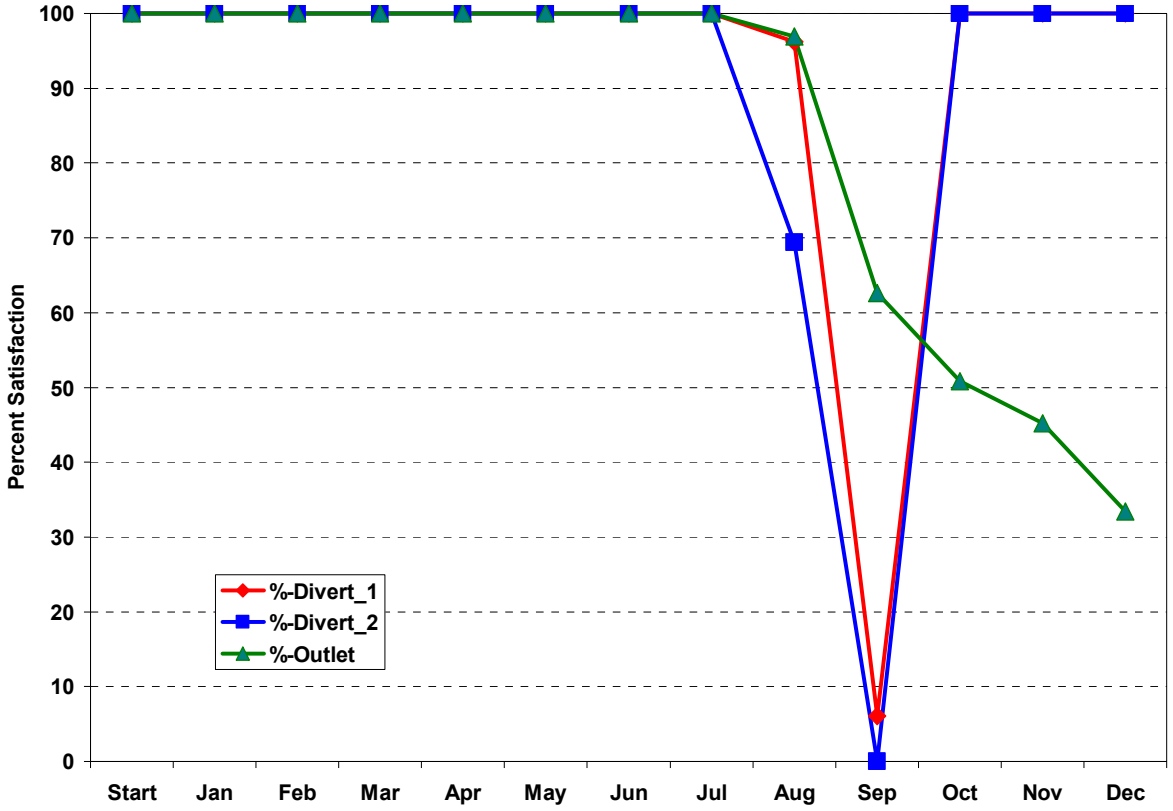


Figure 3.3.2. Demand satisfaction chart for river management example using nonlinear objective function.

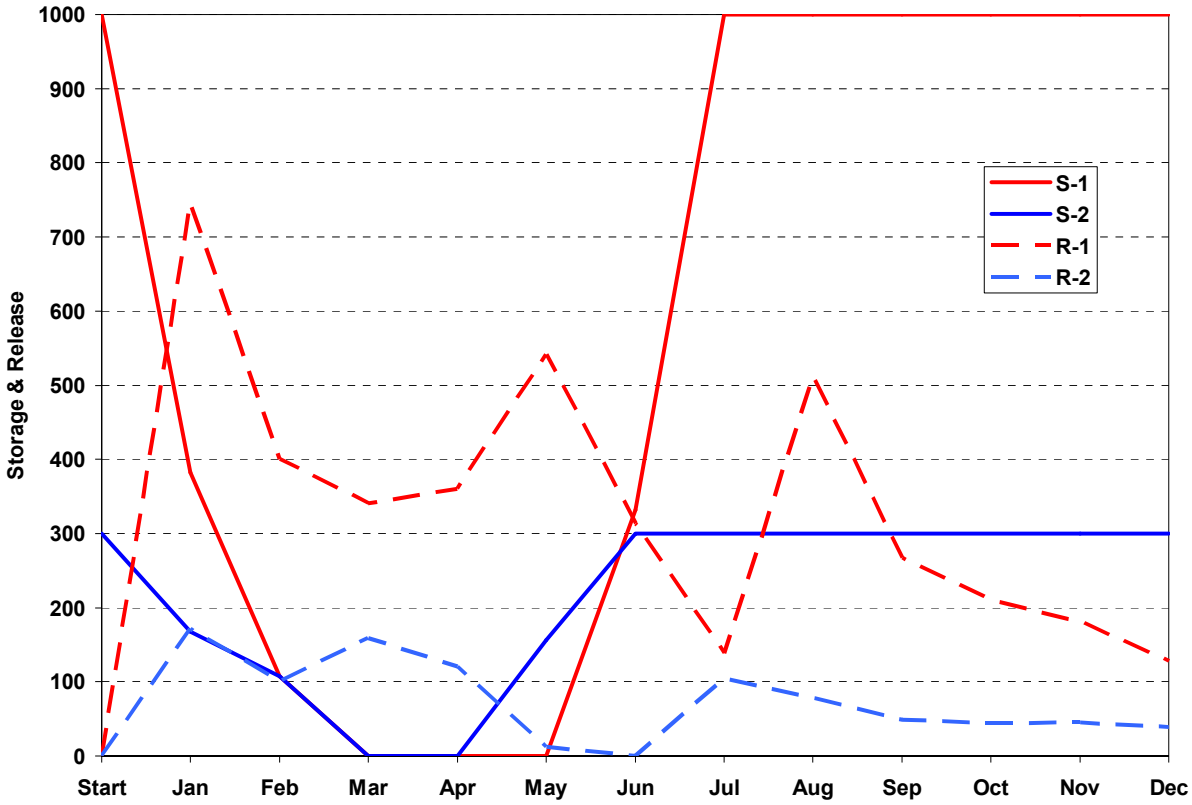


Figure 3.3.3. Reservoir storages and releases for river management example using nonlinear objective function.

This example served as the basis for the development of the models of optimal management of water resources in river systems (see N. Kipshakbaev and A. Tasybaev, Vol. 2, Section 1.1, "Optimization of the Syrdarya Water and Energy Uses under Current Conditions: Kazakhstan Part," McKinney, D.C. and A.K. Kenshimov (eds.), 2000).

3.4 Onstream and Offstream Reservoir Management

Consider the following example of solving a complex problem of using conditional operators with variables. We have two reservoirs. The first reservoir takes water from upstream (according to a hydrograph of inflows) and releases it to downstream demands (according to a schedule of demands). The second reservoir receives water from the first reservoir through a valve that opens when the volume in the first reservoir exceeds the volume in the second reservoir. The second reservoir sends water back to the first reservoir through a valve that opens when the volume in the first reservoir is insufficient to meet demands. In this case, water from the second reservoir will make up the deficit of the first reservoir through the return valve.

This is not an abstract example, but it is taken from real life. Tuyamuyun reservoir on the Amu Darya River in Central Asia is comprised of two parts: an onstream and an offstream part. Water comes to the offstream part of the reservoir by gravity flow when the water level of the onstream part exceeds the water level of the offstream part. On the other hand, water of the offstream reservoir comes to the onstream section when there is a shortage of irrigation water for agriculture. Water storage in the offstream reservoir is more preferable than in the onstream part, due to its structure. Hence, preferential storage of water is achieved in the offstream part of the reservoir.

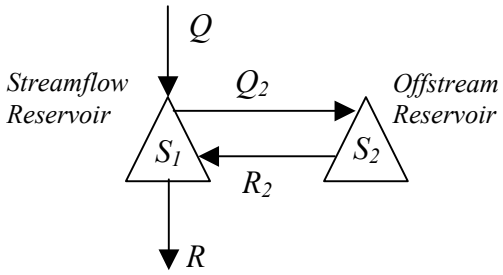


Figure 3.4.1. On- and off-stream reservoir system diagram.

The mathematical description of the problem can be written as:

1) $S_1^t = S_1^{t-1} + Q^t - R^t - Q_2^t + R_2^t$ Balance equation for onstream reservoir (1)

2) $S_2^t = S_2^{t-1} + Q_2^t - R_2^t$ Balance equation for offstream reservoir (2)

3) $S_1^t = S_2^t$ if $Q^t > 0, \text{and } R_2^t = 0$ Switch for flow from on- to off-stream

$$S_{1,\min} < S_1^t < S_{1,\max}$$
$$S_{2,\min} < S_2^t < S_{2,\max}$$

$$0 < Q^t < Q_{\max}$$
$$0 < R_2^t < Q_{\max}$$

$$Q^t = F^t, \qquad S_1 = S_1^0 \qquad \text{if } t = 0$$
$$Q_2^t = F_2^t, \qquad S_2 = S_2^0 \qquad \text{if } t = 0$$

4) $Minimize \sum R_2^t$

where

Q - inflow to the streamflow reservoir,
R - release from the streamflow reservoir
Q_2 - inflow to the streamflow reservoir,
R_2 - release from the streamflow reservoir
S_1 - storage volume in streamflow reservoir
S_2 - storage volume in streamflow reservoir

The third equation enters the calculation when the volumes in both reservoirs are equal. Equation 4 is the objective function which is to minimize the need for releases from the offstream reservoir.

How can we model this situation in GAMS? Since we can not take equation 3 out of the computation process using the conditional operator, we must eliminate its impact by isolating the elements of the equation through transforming it into an identity.

```
SETS
n reservoir  /tmgu,sult/;

SETS
t  time   / jan, feb, mar, apr, may,
            jun, jul, aug, sep, oct,
            nov, dec, eenndd /
tt(t) /jan/  ;

TABLE Q(n,t)  inflow to the onstream reservoir(mln.cub.m)
          jan feb mar apr may jun jul aug sep oct nov dec  eenndd
tmgu       128 125 234 360 541 645 807 512 267 210 981 928   250

TABLE R(n,t) required releases from the onstream reservoir (mln.cub.m);
          jan feb mar apr may jun jul aug sep oct nov dec  eenndd
tmgu       111 111 111 500 222 700 333 333 333 333 333 333   200

VARIABLES
            Q2(t),
            R2(t),
            S(n,t),
            obj;

EQUATION
balanc1(n,t),    water balance of the stream-flow reservoir;
balanc2(n,t),    water balance of the offstream reservoir;
rules1(n,t),     rules of filling the reservoirs;
ben;             objective function;

balanc1(n,t)$(not tt(t))..
           S('tmgu',t)-S('tmgu',t-1)
        =e=
           +Q('tmgu',t)-R('tmgu',t)
           -Q2(t)+R2(t);

balanc2(n,t)$(not tt(t))..
          S('sult',t)-S('sult',t-1)
       =e=
```

```
          Q2(t)-R2(t);

rules1(n,t)$(not tt(t))..
         (S('sult',t)-S('tmgu',t))     -
         (S('sult',t)-S('tmgu',t))     *
          (1.0-Q2(t) /(Q2(t)+0.00000001))
       =e=
         0.0;

ben.. obj =e= SUM(t$(not tt(t)),R2(t));

S.LO('tmgu ',t) = 1150; lower level of the stream-flow reservoir;
S.UP('tmgu ',t) = 4590; upper level of the stream-flow reservoir;
S.FX('tmgu ','jan') = 1200; initial level of the stream-flow reservoir;
S.LO('sult ',t    ) = 100 ; lower level of the offstream reservoir;
S.UP('sult ',t    ) = 4590; upper level of the offstream reservoir;
S.FX('sult ','jan') = 1200; initial level of the offstream reservoir;
R2.UP(t)= 1500;               upper limits of release from offstream;
R2.LO(t)= 0.0;                lower limits of release from offstream;
Q2.UP(t)= 1500;               upper limits of diversion to offstream;
Q2.LO(t)= 0.0;                lower limits of diversion to offstream;
Q2.L(t)= 0.00001;             initial stage of finding a solution;

OPTION ITERLIM= 20000;        not more than 20000 iterations
OPTION LIMROW =12;            list all estimated time intervals;
OPTION OPTCR  =0.000001;      accuracy of optimal solution;

OPTION nlp  =minos5;

MODEL tmgu   /all/;

SOLVE tmgu   USING  NLP MINIMIZING obj;

PARAMETER A(t);
A(t)=(1-(Q2.l(t)/(abs(Q2.l(t))+0.00001)));

FILE res /River0.txt/
PUT res;
PUT " objective = ",obj.l:10:5;
PUT /"  =========balance ====================  "/;
PUT "                                          "/;
PUT " A       R2     Q2     S_sult    S_tmgu      Q    ";
PUT "     R        in-out   dS-tmgu   dS-sult "/;
LOOP((t)$(ord(t) ne card(t)),
 PUT   A(t):5:2,R2.L(t):7:1,Q2.L(t):7:1,S.L('sult',t):10:1,S.L('tmgu',t):10:1,
       Q('tmgu',t ):10:1,R('tmgu',t ):10:1,(Q('tmgu',t )-R('tmgu',t )):10:1,
       (S.L('tmgu',t)-S.L('tmgu',t-1)):10:1,(S.L('sult',t)-S.L('sult',t-1)):10:1
/;);
 PUT " ==================================== "/;
```

The result of the calculation will be represented in text file:

```
objective =   21.50000
  =========balance ====================

 A       R2     Q2     S_sult    S_tmgu      Q         R        in-out   dS-tmgu   dS-sult
 0.50    0.0    0.0    1200.0    1200.0     128.0     111.0      17.0    1200.0    1200.0
 0.00    0.0    7.0    1207.0    1207.0     125.0     111.0      14.0       7.0       7.0
 0.00    0.0   61.5    1268.5    1268.5     234.0     111.0     123.0      61.5      61.5
 1.00   21.5    0.0    1247.0    1150.0     360.0     500.0    -140.0    -118.5     -21.5
```

```
0.00    0.0  111.0    1358.0    1358.0     541.0     222.0     319.0     208.0     111.0
1.00    0.0    0.0    1358.0    1303.0     645.0     700.0     -55.0     -55.0       0.0
0.00    0.0  209.5    1567.5    1567.5     807.0     333.0     474.0     264.5     209.5
0.00    0.0   89.5    1657.0    1657.0     512.0     333.0     179.0      89.5      89.5
1.00    0.0    0.0    1657.0    1591.0     267.0     333.0     -66.0     -66.0       0.0
1.00    0.0    0.0    1657.0    1468.0     210.0     333.0    -123.0    -123.0       0.0
0.00    0.0  229.5    1886.5    1886.5     981.0     333.0     648.0     418.5     229.5
0.00    0.0  297.5    2184.0    2184.0     928.0     333.0     595.0     297.5     297.5
=====================================
```

From the results we see that the offstream reservoir was used during the third time interval, when the stream-flow reservoir was incapable of covering the required release.

3.5 Water Rights and Markets

3.5.1 Introduction

As the costs of water supply development increase it is increasingly important that supplies be allocated more efficiently than in the past. Systems of water allocation with nontransferable water rights can lead to rigid, inflexible, and inefficient allocations of water (Gibbons, 1986; Howe et al., 1986). Several advantages of private ownership and market exchange over bureaucratic control and allocation of water often exist. Markets have been established which have been successful in transferring water from low-valued to higher-valued uses over time (Rosegrant and Binswanger, 1994), where value is defined as the maximum amount a user would be willing to pay for the use of water (Gibbons, 1986). However, the establishment of water markets are often inhibited by the presence of externalities (third-party effects) such as increased pollution or changes in return flows. These effects must be accounted for in deciding a water right transfer and losing parties must be compensated for these effects (Howe, 1996).

Water is a necessary and scarce resource for the sustenance of human society and culture. However, the allocation of water to beneficial uses is a difficult problem, quite different from other resource allocation problems commonly considered in economics (Tregarthen, 1983). The quantity of water available in a river basin will fluctuate year-to-year. The interactions between surface water and groundwater are complex and not well known in many basins. Economies of scale exist in developed storage and distribution systems, encouraging the development of large systems that may be inflexible and non-robust over the long run. Externalities often exist and may require public sector intervention. The consumptive use of water in many cases may lead to the degradation of in-stream flow values and the loss of those benefits.

Several policy options exist that may lead to increases in water use efficiency while reducing environmental degradation and releasing water for increasing demands in other economic sectors (Rosegrant and Binswanger, 1994):

1. **Technological solutions:** (a) construction of new water resource systems, and (b) rehabilitation and modernization of existing systems, e.g., canal and drainage lining, and field drainage in irrigation systems.

2. **Reform of public management of water resource systems:** (a) modification of water distribution methods, (b) implementation of water pricing policies, and (c) reform of water management bureaucracies.

3. **Communal water resource system management** which involve water users more directly in both the process of system management and improvement.

4. **Establishment of tradable property rights** in water and development of markets in these rights. Market allocation of resources may be efficient given well-defined and nonattenuated (completely specified, exclusive, transferable, and enforceable) initial property rights allocation and low transaction costs.

Options 1 - 3 have been widely used by international lending institutions and national governments. Option 4 is somewhat new and is explored below.

3.5.2 Water Rights

Water law differs from country to country and within the US it varies from state to state. Traditionally, water law has been based on common law, but more recently it has shifted a bit toward legislative law. Most common law is based on protecting the quantity, not the quality of water. Common law is comprised of traditional legal aspects laid down by court decisions and it is based on precedent set by older cases. Common law can be overturned as the needs of society change over time. In the US, common law derives its legitimacy from the constitutions (US and States). Legislative law, on the other hand, is comprised of statutory law, where the legislature passes laws that regulate water, and administrative law, where the legislature may enable administrative bodies to write rules and regulations that have the power of law.

A well defined system of water rights is a necessary condition for the development of water markets. Water rights can be vested with individual citizens or with the government (local, state or national). Water rights are essentially a bundle of entitlements defining a water right owner's:

1. rights;
2. privileges; and
3. limitations

for the use of the water. Water rights are generally treated as real property with the right holder having a usufructuary right to make use of the water but not a right to physical possession of the water (Hirshleifer, et al., 1960). These rights must be well defined and exclusive to the person or entity owning them. Such a system of water rights must completely specify the (Howe, et al., 1986):

1. quantity of water that may be diverted;
2. quantity of water that may be consumed;
3. timing of the water delivery;
4. quality of the delivered water;
5. place of diversion; and

6. place of application.

Changes in any of these characteristics will likely affect other water users in the basin or system.

An efficient water rights structure should include:

- **Universality:** Rights must be privately owned, and entitlements must be completely specified.

- **Exclusivity:** Benefits and costs accrued as a result of owning and using the rights accrue to the owner (and only the owner). This property of water rights is often violated, e.g., when a user does not bear all of the costs of a water allocation, creating an externality that is not paid for by the user. An externality may exist whenever the welfare of some user depends on its own activities and the activities of some other user(s) as well.

- **Transferability:** The transfer of rights from one user to another must be entirely voluntary. The inability to transfer the rights would prevent the owner of the right from recognizing the true opportunity cost of the water, i.e., the value that another person may place on it.

- **Enforceability:** Right owners must be secure from involuntary seizure of their rights or encroachment on their rights.

Usually, systems for water rights (implicit or explicit) fall into one of several categories (Howe, 1996):

- **Non-tradable permits for water from undeveloped (natural) supplies**
  Non-tradable permits or rights are typically specified by laws or regulations, they are for specific or defined periods and they are not tradable. Problems associated with this type of arrangement include the fact that this method of water allocation does not consider the economic efficiency or equity of the use, and allocations may be inflexible and unresponsive to changes in social values.
- **Contracts for water from developed supplies**
  Developed supplies usually provide storage and distribution facilities and water is allocated to customers by contracts (as opposed to non-tradable permits where water is distributed by water right). Contracted water supply is usually for a specific use. Problems associated with this type of arrangement include the fact that the economic efficiency or equity of the water use is not considered.

In many cases, water demand is estimated from the projected requirement or need for water, that is, the farmer's ability to put water to use (Gardner, 1983). This method of demand estimation results in a maximization of physical yield rather than profit or social benefits. This can lead to the development of new water supplies rather than using economic incentives and market mechanisms to allocate water to its best uses. Systems of this type are common in the states of California (Gardner, 1983) and Texas (TWDB, 1997).

In other cases, a fixed cost per hectare of crop is charged for water. This violates economic principles since the price is not related to the quantity of water applied or used, and there is little

incentive to conserve water. This type of system is often justified by the difficulty and expense of determining how much water is delivered to a farm.

3.5.2.1 Riparian water rights systems

Under riparian water rights systems, the owner of land bordering a stream or lake has the right to take water for use on the land. The right to use the water exists solely because of the relation of the land to the water and resides in the ownership of the land. The first riparian user acquires no priority over those who may use the stream at a later date; the rights of upstream and downstream users are viewed as being coequal (Hirshleifer et al., 1960).

Under riparian systems, the owners of lands bordering water bodies may have "reasonable use" of the waters, provided that the water is returned undiminished in quality or quantity (Howe et al., 1986). That is, the withdrawal must be reasonable with respect to the requirements of the other riparians. The determination of what constitutes reasonable use is left to the courts. A riparian right subject to the reasonable use doctrine has no guaranty to a definite quantity of water. Under the riparian system, the transfer of water rights between competing uses by a market system is severely hampered (Hirshleifer et al., 1960).

Riparian rights are most appropriate for humid regions. Where water is truly scarce and/or where water quality problems are important, the riparian doctrine simply doesn't work (Howe, 1996).

3.5.2.2 Appropriative water rights systems

The doctrine of appropriation gives no preference to the use of water by riparian landowners. Appropriative (or prior) systems tend to exist in areas of water scarcity where users are located away from water bodies. Scarcity means that each succeeding appropriation results in fewer or less valuable resources available for other users. Scarce resources come to be appropriated in their natural state according to the principles of priority of right and beneficial use (Cuzan, 1983).

The earliest water right on a given watercourse has preference over later users, "first in time means first in right." Once the appropriation is granted, it becomes senior to subsequent appropriations. In times of shortage senior or older rights have precedence over junior or newer rights. That is, senior rights have first call on available water. Appropriative rights are a right to use, not a right to own, and the beneficial use of the water is required. Beneficial use has been described as use of water in a useful industry or to supply a well-recognized want (Tregarthen, 1983). In many cases, the owner of an appropriation may lose the right as a result of failure to put it to beneficial use.

The two rules of appropriative water rights, priority and beneficial use, result in the separation of rights to water from the rights to land. Persons can mobilize capital to build water supply works and transport water to wherever it is most productively used (Cuzan, 1983). Appropriative rights may be a system in which rights are clearly defined and transferable subject to the stipulation of "no injury" (Tregarthen, 1983). It is the severability of appropriative rights that causes them to be transferable.

Often under appropriative rights systems water is owned by the public and appropriators are granted the right to use the water but ownership of the resource remains with the state (Cuzan, 1983). Often this state expropriation of water rights leads to a system of controls which makes it difficult for water to be transferred privately through sales. These systems can generate pressure for monumental water schemes by governmental agencies which subsidize low-value water uses.

Economic efficiency requires that the marginal value of water used be equal in each use, net of transport costs, and assuming that marginal values include both private and social benefits and costs (Gardner, 1983; Howe et al., 1986). Assuming that water is homogeneous, i.e., no quality variations, water prices should vary among users only by the cost of moving it from one user to another (Hirshleifer, et al., 1960).

Two main types appropriative rights systems are common: priority rights and proportional rights systems

Priority rights

Priority rights operate on the doctrine of "first in time, first in right." If the flow in a river is sufficient to provide only x% of the water appropriated, then a call for water from the senior water rights holders can shut off diversions to the lowest (100 - x)% priority rights holders. Senior water rights holders have less risk than junior rights holders, but senior rights holders may place a lower value on the last unit of water than junior rights holders. In this case, a trade should occur, the senior rights holder selling water rights to junior rights holders, thus reducing the risk to the junior appropriators (Tregarthen, 1983). Priority rights allow different degrees of water supply reliability to be purchased, but the heterogeneous nature of the rights makes it difficult to organize markets.

Proportional rights

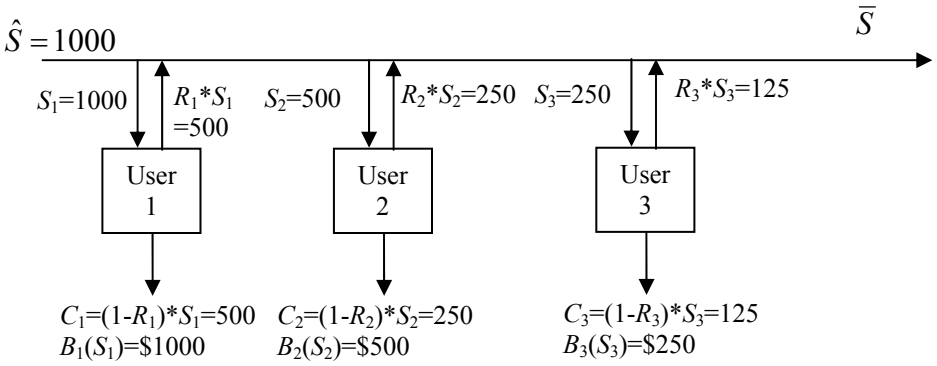
A proportional rights system shares available water among users according to a set of percentages determined by the number of rights owned, e.g., if a user owns 10 rights out of 100, then the owner is entitled to 10% of available water. Proportional rights systems require the purchase of more shares to reach any given level of assurance of water supply. The homogeneity of proportional rights makes it much easier to create markets than under the priority system.

3.5.3. Water Markets

3.5.3.1 Right to divert or consume

The right to transfer water may not be the amount of water appropriated to the use, but the "duty of water" at the point of use (Tregarthen, 1983). This concept limits the transfer of water rights, based on the consumptive use by the seller and the prospective consumptive use of the buyer. In many situations it is desirable to protect downstream users from a loss of water due to an upstream water rights trade. When rights are transferred, the use of the water may change, and with it the amount of

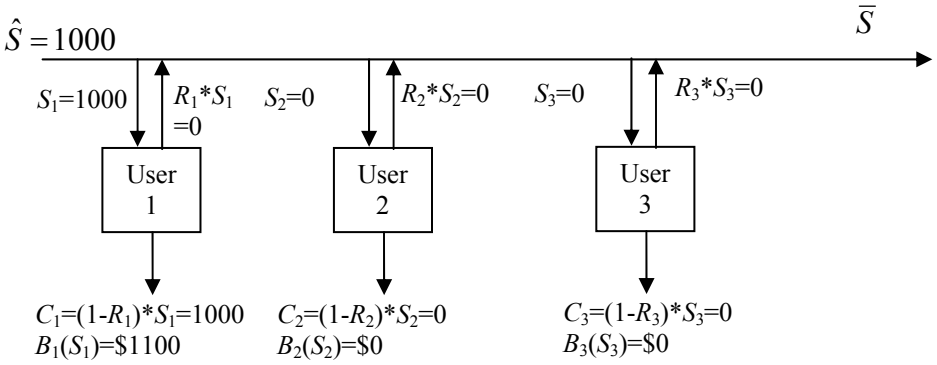
water that is consumed. The result may be a change in the amount of water available to downstream users. Often it is desirable to limit transfers so that the amount of water consumed is not changed. Consider the following example adapted from Gisser and Johnson (1983) where there are three users along a river and flow into the system is 1000 units per time period. User 1 diverts $S_1 = 1000$ units and has a return flow coefficient (R_1) of 0.5, that is, User 1 consumes $C_1 = 500$ units of water. Downstream, User 2 diverts $S_2 = 500$ units and has a return flow coefficient (R_2) of 0.5, and User 3 diverts $S_3 = 250$ units and has a return flow coefficient (R_3) of 0.5. The total diversion is 1,750 units of water, a greater amount than the initial streamflow.



Total Diversion = 1,750
Total Benefit=$1,750

Figure 3.5.3.1.1. River system for water allocation.

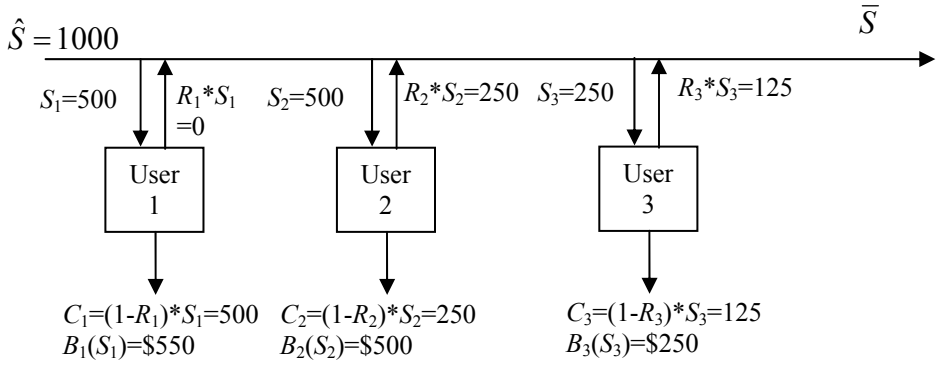
Now consider that User 1 decides to sell his/her entire diverted amount to another user outside the basin for $1.1 per unit. The net result is that User 1 is no better off than before, but Users 2 and 3 have been left without any water to divert and there is an overall net loss for the basin of $650



Total Diversion = 1,000
Total Benefit=$1100

Figure 3.5.3.1.2. River system for water allocation without consideration of consumptive use.

An appropriator (user) may own a right to divert a given quantity of water but the user can only transfer this right according to the amount of water consumed. Determining the consumptive use of water and the amount of water that returns to the river or canal can be difficult and costly. Consider again the above example, but now User 1 decides to only sell the amount of his/her previous consumptive use (500 units) for a price of $1.1 per unit. In this case the downstream users continue to receive their water and may divert as before.



Total Diversion = 1,250
Total Benefit=$1,300

Figure 3.5.3.1.3. River system for water allocation with consideration of consumptive use.

Several authors have suggested that it is better to define water rights according to the consumptive use system rather than the diversion rights system. Using the consumptive rights system the ownership of the right is clear, transfers do not require litigation, the incentive to conserve water exists, and this conservation leads to water that can be sold to downstream users (Tregarthen, 1983). However, others have suggested that the diversionary rights system is preferable (Rosegrant et al. 1995).

The definition of the tradable portion of a water right depends on the method of handling return flows. In California, the tradable portion is limited to consumptive use (consumptive right) with protection of third-party rights to return flows. This method increases transaction costs because of the difficulty in measuring consumptive use and return flows. Consumptive use is defined to be the actual evapotranspiration of crops plus any water lost to deep percolation. Thus, the water available to trade includes water that would have been consumptively used and water that would be irretrievably lost to beneficial use. In Chile and Mexico, rights are proportional to streamflow (diversionary right) and rights to return flow are retained by the water authority. Return flows are made available to users at no charge, but no rights are assigned to these flows. Changes in return flows due to water rights trades are not actionable. This method has been demonstrated to reduce transaction costs. So the tradable water is the full diversion right which is proportional to stream

flow.

What is the most appropriate method in developing countries? The transaction costs of enforcing consumptive rights increase but they protect third-parties against adverse impacts from water trades. If the lost benefits from not trading exceed the costs of adverse impact from lost return flows, then diversionary rights system is preferred. In general, the diversionary rights system will be preferred in developing countries so as to prevent high transaction costs, thus preventing the development of markets.

3.5.3.2 Tradable water rights markets

Tradable water rights are rights to use water that can be transferred all or in part, separately from the transfer of land (Rosegrant et al., 1996). Tradable water rights may be permanent, long-term, or even short-term. Tradable water rights markets may be capable of allocating water more effectively than other more restrictive and centrally controlled systems. Markets can operate most efficiently when the commodity being allocated is homogeneous (Howe et al., 1986). Heterogeneity of uses leads to difficulty in organizing a market, transmitting information to users, and matching sellers and buyers. Rights to water resources already exist in most countries (a) by custom, or (b) by law and regulation. Establishing tradable rights is a matter of reforming existing systems.

Characteristics of markets

Several desirable characteristics for water allocation mechanisms (regional, river basin or irrigation district level) have been described by Howe (Howe et al., 1986; Howe, 1996):

- **Flexibility over time**

  Water can be shifted from use to use and place to place as climate, demographics, and economic conditions change over time. Short-term (responding to climatic factors) and long-term (responding to demographic and economic factors) flexibility is necessary. It is important to note that not all water must be subject to reallocation, only a tradable margin must exist within each water-using area that is subject to low cost reallocation and this volume can be a relatively small part of the regional supply (Howe et al., 1986). Flexibility allows equating the marginal values in the water's various uses (Howe et al., 1986; Gisser and Johnson, 1983).

- **Security of tenure for established users**

  Water users must be assured of continued use or they will not invest in and maintain the water resource system. This encourages long-term investments that generate positive net benefits. In a market system, no one can be forced to sell.

- **Real opportunity costs of water**

  Valuation of water at its opportunity cost, the maximum value of outputs that could have been

produced had inputs not been used to produce the item in question (Field, 1994), provides incentives for users to shift from inefficient water uses and methods to more highly valued, less water-intensive uses and methods. Opportunity cost pricing can be implemented through the establishment of tradable water rights and development of markets in these rights (Rosegrant et al., 1996).

Water is often a scarce input to production and it is frequently priced well below its value in use (Gardner, 1983). Historically, the price of water, at most, has reflected the costs of its capture and distribution. The control of low priced water can provide access to enormous profits in many cases. A perpetual contract for the supply of water at a fixed price may fail to reflect changing opportunity costs involved in continued use. Water is one input to agricultural production, other inputs include land, capital, energy, chemicals, and labor. If production is to be profitable, all inputs used must be valued at least at their opportunity costs. A price for water established in a market and the ability to sell water (transfer or trade water rights) recognizes the real opportunity costs of the water in the use being considered. This prevents the acceptance of water uses that are less valuable than alternative uses.

It is desirable to maximize the scope of a water rights market so that transactions take place over as wide a geographical area and among as wide a variety of users as possible, subject to transaction cost limitations.

- **Differentiated risk-bearing**

  While old methods are familiar even if they are outmoded, new methods may increase uncertainty, even while they promise advantages. Predictability of the outcome of the transfer process is necessary to ensure that long-term investments that generate positive net benefits are encouraged.

- **Fairness to participants**

  Water users should not impose uncompensated costs (externalities) on other parties. Externalities occur whenever withdrawal, consumption, or quality changes by one user affect other water users. Parties giving up water should be compensated and those injured by changes in allocations should be compensated. Market transactions should guarantee fairness since no person will sell if they will not be made better off.

- **Protection of public values**

  Some values may be of little concern to individual water users and they may not be adequately reflected in the market exchange and these must be protected by social oversight, e.g., water quality and instream flows (Howe, 1996). Protection of public values will ensure that allocations will achieve the highest aggregate benefit level.

Benefits of tradable water rights markets

The benefits from establishing tradable water rights markets include (Rosegrant and Binswanger, 1994; Rosegrant et al., 1996):

- **Empowerment of water users** by requiring their consent to any reallocation of water and compensation for any water transferred;

- **Security of tenure** of water rights to the water users, which encourages investment in system efficiency improvements;

- Induces users to consider the **full opportunity cost** of water, including its value in alternative uses, providing incentives to efficiently use water and gain additional income through the sale of saved water;

- Provide incentives for users to **take account of external costs** imposed by their water use, reducing resource and environmental degradation;

- **Formalizes existing rights** to water; and

- **Provides maximum flexibility** in responding to changes in crop prices and water values.

Constraints of tradable water rights markets

Constraints to establishing tradable water rights markets leading to high transaction costs include (Rosegrant and Binswanger, 1994; Rosegrant et al., 1996):

- The unique physical, technological and economic **characteristics of water resources** systems pose problems;

- The **variable nature of water flow** makes achieving necessary certainty; and

- **Return flows** from water use can generate environmental degradation. Multiple reuse of water creates the likelihood of significant externalities imposed on third parties.

Policy considerations of tradable water rights markets

Policy considerations in developing tradable water rights markets include (Rosegrant et al., 1996):

- Definition of a method of initial allocation of water rights. This can be based on, among other things, historic water use (Chile and Mexico), fully appropriated existing rights (California);
- Type of rights, prior or proportional appropriative rights: Prior rights (California), Proportional (Chile and Mexico);
- Consumptive use or diversionary treatment of return flows;
- Indirect economic effects;

- Environmental protection;
- Water user associations;
- Infrastructure;
- Public and private institutions; and
- Regulations: Excessive regulation leads to high transaction costs, inadequate regulation leads to third-party costs or environmental degradation

3.5.3.3 Modeling tradable water rights markets

There are two fundamental strategies for dealing with water scarcity in river basins, supply management and demand management; the former involves activities to locate, develop, and exploit new sources of water, and the latter addresses the incentives and mechanisms that promote water conservation and efficient use of water (Rosegrant et al., 2000). Markets in tradable water rights can reduce information costs; increase farmer acceptance and participation; empower water users; and provide security and incentives for investments and for internalizing the external costs of water uses. Market allocation can provide flexibility in response to water demands, permitting the selling and purchasing of water across sectors, across districts, and across time by opening opportunities for exchange where they are needed. The outcomes of the exchange process reflect the water scarcity condition in the area with water flowing to the uses where its marginal value is highest (Rosegrant and Binswanger 1994; Rosegrant 1997). Markets also provide the foundation for water leasing and option contracts, which can quickly mitigate acute, short-term urban water shortages while maintaining the agricultural production base (Michelsen and Young 1993).

Water trading in a basin is constrained by the hydrologic balance in the river basin network; water may be traded taking account of physical and technical constraints of the various users, reflecting their relative profitability in trading prices; water trades reflect water scarcity in the basin that is influenced by both basin inflows and the water use plans of the users (Rosegrant et al., 2000).

The price that a water user would be willing to pay to acquire additional water must be determined for each user. This can be achieved by determining a shadow price – water withdrawal relationship can determined for each user. For this, a model must be run with varying water rights for each user as inputs and shadow prices or marginal values as output derived from the water balance equations (each user has a water balance equation in the model). If necessary, these shadow prices can be averaged over all uses for each user to obtain one shadow price for each water supply level for user. The Figure below shows the result of this for the problem of Exercise 1 below.

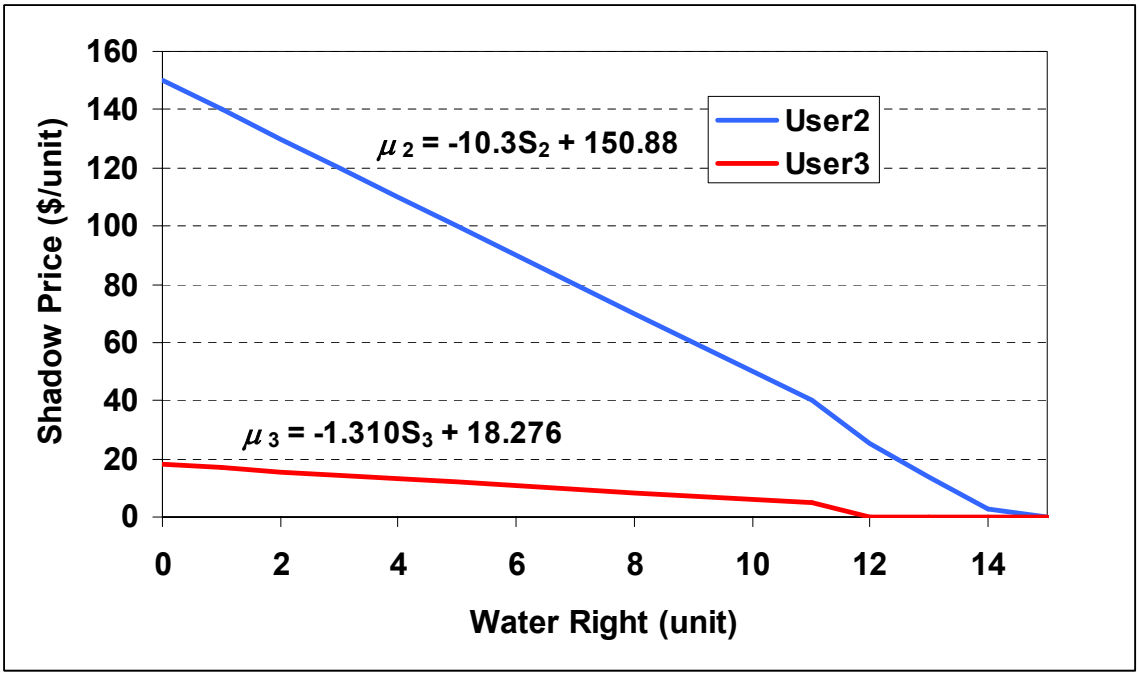


Figure 3.5.3.3.1. Shadow price for water users.

In the model of water trading, the objective is to maximize the combined benefits of all the users

$$\text{Maximize } B = \sum_i B_i = \sum_i (a_i S_i - b_i S_i^2 + \sum_j x_{i,j} \cdot wtp_j - \sum_j x_{j,i} \cdot wtp_i) \qquad (3.5.3.3.1)$$

where

B	Total benefit to all water users;
B_i	Benefit to User i, a quadratic benefit function is assumed here with coefficients a_i and b_i;
S_i	Water withdrawal by user i;
wtp_i	Water trading price for user i;
$x_{i,j}$	Water sold by user i to user j;

User i has access to water from water right or purchase

$$S_i \le wright_i - \sum_j x_{i,j} + \sum_j x_{j,i} \qquad (3.5.3.3.2)$$

where *wright* is water allocated to a user under prescribed water rights. Each user has shadow price for water which is a linear function of the amount of water demanded

$$wtp_i = am_i + bm_i S_i \qquad (3.5.3.3.3)$$

No user is allowed to sell more water than their water right

$$\sum_j x_{i,j} \le wright_i \qquad (3.5.3.3.4)$$

Trades are unidirectional, that is, if a user buys water from another user, then they can not sell water to the same user

$$x_{i,j} * x_{j,i} = 0 \qquad (3.5.3.3.5)$$

3.6 Managing Water Quality in Streams

```
SETS  i reaches /1*7/
SETS  k treatment levels /1*4/

ALIAS(i,i1);
SET i_from_i(i,i1)  reach i gets water from node i1 (any node)
/
3.1,
3.2,
5.3,
5.4,
6.5,
7.6
/;

Parameter W(i) Waste input to plant (mg per L)
/
1 248,
2 408,
3 240,
4 1440,
5 0,
6 2180,
7 279
/;

Parameter E_init(i)  Initial treatment level of plant
/
1 0.67,
2 0.30,
3 0.30,
4 0.30,
5 0.0,
6 0.30,
7 0.30
/;

Parameter DO_Sat(i) Saturate DO level in reach (mg per L)
/
1 10.20,
2 9.95,
3 9.00,
4 9.70,
```

```
5 9.00,
6 8.35,
7 8.17
/;

Table C(i,k)  Treatment plant costs for various levels of treatment ($)
       1             2             3             4
1      0         22100         77500        120600
2 630000        780000        987000       1170000
3 210000        277500        323000        378000
4 413000        523000        626000        698000
5 0                  0             0             0
6 500000        638000        790000        900000
7 840000       1072000       1232500       1350000

Parameter R(k) Possible treatment levels
/
1 0.60,
2 0.75,
3 0.85,
4 0.90
/;

Parameter Qw(i) Wastewater inflows (million m3)
/
1 19,
2 140,
3 30,
4 53,
5 0,
6 98,
7 155
/;

Parameter Dw(i) DO deficit in wastewater inflow (mg per L)
/
1 1.0,
2 1.0,
3 1.0,
4 1.0,
5 0.0,
6 1.0,
7 1.0
/;

Parameter Qi(i) Source inflow (million m3)
/
1 5129,
2 4883,
4 1120
/;

Parameter Dmax(i) Maximum allowed DO deficit on reach (mg per L)
/
1 3.2,
2 2.45,
3 2.00,
4 3.75,
5 2.50,
6 2.35,
7 4.17
```

```
/;

Parameter Di(i) DO Deficit in source flow (mg per L)
/
1 0.70,
2 1.95,
4 0.16
/;

Parameter Bi(i)  BOD level in source flow (mg per L)
/
1 1.66,
2 0.68,
4 1.0
/;

Parameter k1(i)  Deoxygenation coefficient (day-1)
/
1 0.93,
2 0.58,
3 0.68,
4 0.49,
5 0.90,
6 0.69,
7 0.16
/;

Parameter k2(i)  Coefficient (day-1)
/
1 0.06,
2 0.28,
3 0.23,
4 0.46,
5 0.09,
6 0.28,
7 0.78
/;

Parameter k3(i)  Reaeration Coefficient (day-1)
/
1 0.79,
2 0.45,
3 0.50,
4 0.83,
5 0.80,
6 0.86,
7 0.88
/;

Variables
obj        Objective value;

Positive variables
E(i)       level of treatment,
Bw(i)      BOD leaving treatment plant (mg per L),
B(i)       BOD in reach (mg per L),
D(i)       DO deficit in reach (mg per L),
B_minus(i) BOD at beginning of reach (mg per L),
D_minus(i) DO deficit at beginning of reach (mg per L),
B_plus(i)  BOD at end of reach (mg per L),
D_plus(i)  DO deficit at end of reach (mg per L),
```

```
Q(i)       Flow in reach (million m3);

Integer variables
Z(i,k)     integer variables indicating the level of treatment;

E.l(i) = E_init(i);
E.lo(i) = E_init(i);
E.fx('5') = 0.0;

Equations
Objective,
Zsum(i) Sum of integer variables,
Treatment(i) Level of treatment in terms of available levels,
BOD_Plant(i) Output BOD level of plant,
Reach_Flow(i) FLow in reach,
Reach_BOD(i) BOD in reach,
Reach_DO(i) DO deficit in reach,
BOD_Plus(i) BOD level at end of reach,
DO_Plus(i) DO deficit level at end of reach,
DO_Max(i) DO deficit constraint at begin of reach,
DO_Plus_Max(i) DO deficit constraint at end of reach
;

Objective..      obj       =E= sum(k,sum(i,C(i,k)*Z(i,k)));

Zsum(i)..        sum(k,Z(i,k)) =L= 1;

Treatment(i)..   E(i)      =E= sum(k,R(k)*Z(i,k));

BOD_Plant(i)..   Bw(i)     =E= W(i)*(1.0-E(i));

Reach_Flow(i)..  Q(i)      =E= Qi(i)      +sum(i1$(i_from_i(i,i1)),Q(i1))
+Qw(i);

Reach_BOD(i)..                                        B(i)*Q(i)            =E=
Bi(i)*Qi(i)+sum(i1$(i_from_i(i,i1)),B_plus(i1)*Q(i1))+Bw(i)*Qw(i);

Reach_DO(i)..                                           D(i)*Q(i)          =E=
Di(i)*Qi(i)+sum(i1$(i_from_i(i,i1)),D_plus(i1)*Q(i1))+Dw(i)*Qw(i);

BOD_Plus(i)..    B_plus(i) =E= B(i)*k1(i);

DO_Plus(i)..     D_plus(i) =E= B(i)*k2(i)+D(i)*k3(i);

DO_Max(i)..      D(i)      =L= Dmax(i);

DO_Plus_Max(i).. D_Plus(i) =L= Dmax(i);

Model waterquality /all/;

Solve waterquality using MINLP minimizing obj;

file WQ /WQ.txt/
put WQ
put 'Site   E(i)' put /
loop( i,  put i.tl, put E.l(i), put /)
put /
put 'Site   Q(i)' put /
loop( i,  put i.tl, put Q.l(i), put /)
put /
put 'Site   Bw(i)' put /
```

```
loop( i,  put i.tl, put Bw.l(i), put /)
put /
put 'Site                B(i)   B_Plus(i)' put /
loop( i,  put i.tl, put B.l(i), put B_Plus.l(i), put /)
put /
put 'Site                D(i)   D_Plus(i)     Dmax(i)       DO_Plus(i)'
put /
loop( i,   put  i.tl,  put  D.l(i),  put  D_Plus.l(i),  put  Dmax(i),  put
(DO_Sat(i)-D_Plus.l(i)),put /)
```

SOLUTION

```
Site   E(i)
1                  0.85
2                  0.75
3                  0.60
4                  0.90
5                  0.00
6                  0.90
7                  0.85

Site   Q(i)
1               5148.00
2               5023.00
3              10201.00
4               1173.00
5              11374.00
6              11472.00
7              11627.00

Site   Bw(i)
1                 37.20
2                102.00
3                 96.00
4                144.00
5                  0.00
6                218.00
7                 41.85

Site               B(i)   B_Plus(i)
1                  1.79        1.67
2                  3.50        2.03
3                  2.12        1.44
4                  7.46        3.66
5                  1.67        1.50
6                  3.35        2.31
7                  2.84        0.45

Site               D(i)   D_Plus(i)     Dmax(i)       DO_Plus(i)
1                  0.70        0.66        3.20        9.54
2                  1.92        1.85        2.45        8.10
3                  1.25        1.11        2.00        7.89
4                  0.20        3.60        3.75        6.10
5                  1.37        1.24        2.50        7.76
6                  1.24        2.01        2.35        6.34
7                  1.99        3.97        4.17        4.20
```

3.7 Exercises

1. a. Revise the model of Section 3.1 to solve the water user allocation problem where the objective function is given in Loucks et al., Problem 2.9. Note that problem 2.9 does not include the downstream user and this component of the model will have to be take out. The coefficients in the objective function will have to be changed to those specified in the problem (12, -1; 8, -1; and 18, -3) and the release is equal to 10 and 15 units of water. To show that the marginal values are all equal to the dual variable (Lagrange multiplier) for the constraint, note that this marginal value is printed in the solution report file. You can plot the individual objectives, mark the points where the optimal solution occurs for each decision variable (x_i) and compute the slope of the objective function there (the slopes should be the same for each decision variable).

b. Add a fourth user of water (x_4) to the model of Section 3.1 and resolve problem 2.9. The coefficients for the objective function are: $a_4 = 15$ and $b_4 = -2$.

2. (Adapted from Loucks et al., Problem 2-29(a)) A reservoir storage-yield function defines the maximum constant reservoir release that is available during a period of operation. Given a series of reservoir inflows, the yield from a reservoir is a function of the reservoir's active storage capacity. The reservoir storage-yield function can be determined by solving the following linear program:

Maximize R

subject to

$$S_{t+1} = S_t + Q_t - R_t - R \qquad t = 1,...,T;\ T+1 = 1$$

$$S_t \le K \qquad t = 1,...,T$$

where R is the yield of the reservoir (constant release in each time period), S_t is the volume of water stored in the reservoir in each period, Q_t is the inflow to the reservoir in each period, and R_t is the excess release (or spill) from the reservoir in each time period. Use linear programming to derive an annual storage-yield function for a reservoir at a site having the following record of annual flows of 5, 7, 8, 4, 3, 3, 2, 1, 3, 6, 8, 9, 3, 4, 9 units of flow. Find the values of storage required for yields of 2, 3, 3.5, 4, 4.5, and 5 units of flow.

3. Given 7 years of inflow values, the surface vs area data and relationship, and evaporation data for Toktogul reservoir in Kyrgyzstan (see Data Set 1), construct a storage-yield relationship for Toktogul. Show that the result for one point on the storage-yield relationship curve is the same for "Max R - Given K" and "Min K - Given R".

4. Consider the river network illustrated in Figure 3.5.1 below. The system contains one reservoir (Toktogul reservoir in Kyrgyzstan) and two irrigated zones (one in Uzbekistan and one in Kazakhstan). Four different crops can be grown in each irrigation zone: cotton, wheat, rice, and Lucerne. The profit per hectare for each crop is shown in Table 3.5.1 along with the maximum crop

areas and the crop water requirements. The capacity of the reservoir is 19,500 million m3, and the dead storage capacity is 5,500 million m3. The initial storage volume is 14,000 million m3. Write a GAMS model to determine the optimal mix of crops to be grown by each of the countries if the ending storage in the reservoir must be 13,000 million m3. What impact does the ending storage volume have on the solution?

Table 3.5.1. Data for Rerervoir Operation Problem.

Crop water demands (mm per month)

Crop	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Cotton	150	150	100	100	200	250	250	200	50	0	0	0
Wheat	0	0	0	100	300	300	0	0	200	0	0	0
Rice	0	0	0	450	700	800	800	700	50	0	0	0
Lucerne	0	0	0	50	200	350	350	250	50	0	0	0

Naryn River Inflow

Month	Inflow (Million m3)
Jan	386
Feb	346
Mar	416
Apr	713
May	1532
Jun	2373
Jul	2157
Aug	1431
Sep	798
Oct	582
Nov	487
Dec	420

Crop Profits

Crop	Profit ($/ha)
Cotton	350
Wheat	70
Rice	225
Lucerne	50

Maximum area per crop

Crop	Uzbekistan (ha)	Kazakhstan (ha)
Cotton	500,000	65,000
Wheat	500,000	30,000
Rice	0	500,000
Lucerne	50,000	500,000

Maximum total area

Country	Area(ha)
Uzbekistan	2,000,000
Kazakhstan	2,000,000

Return Flow Coefficient

Country	Coefficient
Uzbekistan	0.50
Kazakhstan	0.50

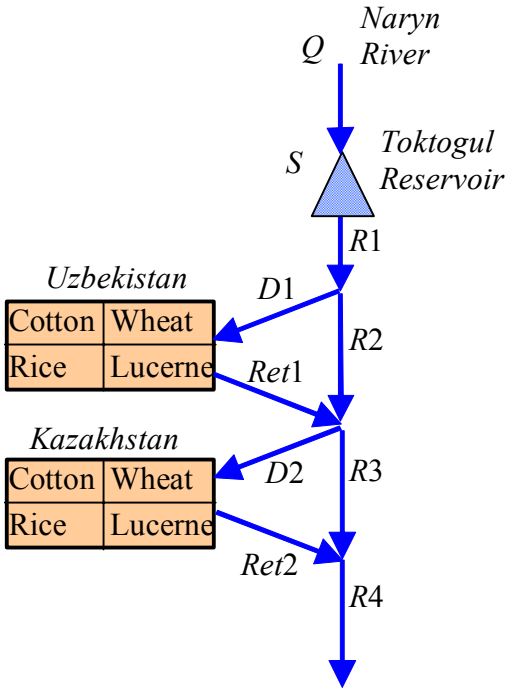


Figure 3.5.1. River Network for Reservoir Operation Task

3.5. Consider the river network illustrated in Figure 3.5.1 below. The system contains one long-term (over-year) storage reservoir (Toktogul reservoir in Kyrgyzstan), four constant volume (pass-through) reservoirs (Kurpsia, Tashkumur, Shamaldasai, and Uchkurgan, all in Kyrgyzstan), two irrigated zones (one in Uzbekistan and one in Kazakhstan). The crop information and initial and final storage information is the same as in Problem 3.4 above. Toktogul is the only reservoir in the cascade for which the volume varies over time depending on the volume of water in storage. The equation expressing the head across the turbines as a function of the reservoir water elevation and storage volume for Toktogul reservoir is

$$H(t) = H_0 + aS(t)^b - H_{tail}$$

where H_0 is the elevation at dead storage, $S(t)$ is the volume of water in storage at time t, a and b are coefficients, and H_{tail} is the elevation of the water in the tail water of the reservoir. The numerical values are given in Table 3.5.1. For the other reservoirs in the cascade, the head across the turbines is a constant, given in the table. Write a GAMS model to determine the optimal mix of crops to be grown by each of the countries if the ending storage in the reservoir must be 13,000 million m3. In addition, compute the power generated from the releases of water.

Table 3.5.1. Data for Rerervoir Operation Problem with Power.

Reservoir	Efficiency	Installed Capacity	H_0	H_{Tail}	$H_{Constant}$
Toktogul	0.874	1200	758	720	-
Kurpsai	0.874	800	-	-	100
Tashkumur	0.874	450	-	-	53
Shamaldasia	0.874	240	-	-	18.5
Uchkurgan	0.874	180	-	-	28

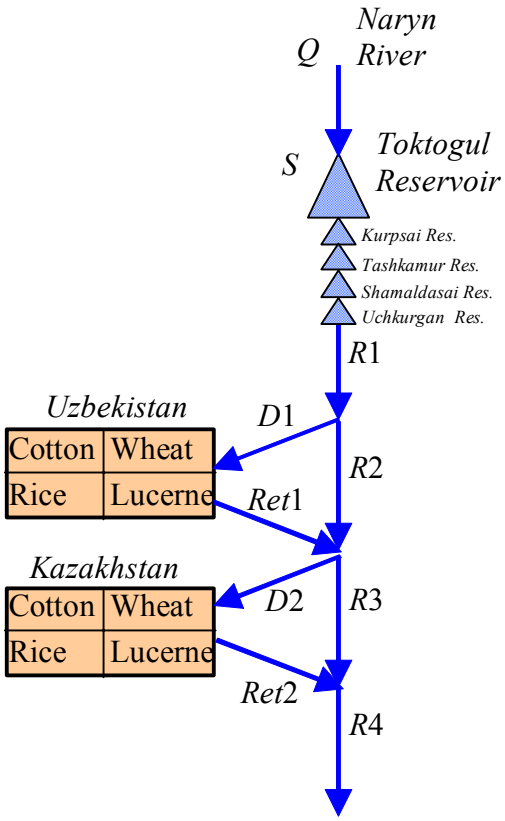


Figure 3.5.1. River network for reservoir operation fask, including power cascade

3.6. Consider the river network illustrated in Figure 3.5.1 above. Write a GAMS model to determine the optimal mix of crops to be grown by each of the countries if the ending storage in the reservoir must be 13,000 million m3. In addition, the Kyrgyz power demand must be satisfied by a combination of hydropower, thermal power, and regional grid power transfers. The total power produced is a combination of the hydropower produced, the thermal power generated by Kyrgyz power plants, and power transferred from the regional grid (see Figure 3.6.1)

$$P(t) = P_{thermal}(t) + P_{hydro}(t) + P_{transfer}(t)$$

where in any month t, $P(t)$ is the total power produced, $P_{thermal}(t)$ is the thermal power produced, $P_{hydro}(t)$ is the hydropower produced, and $P_{transfer}(t)$ is the power transferred from the regional grid.

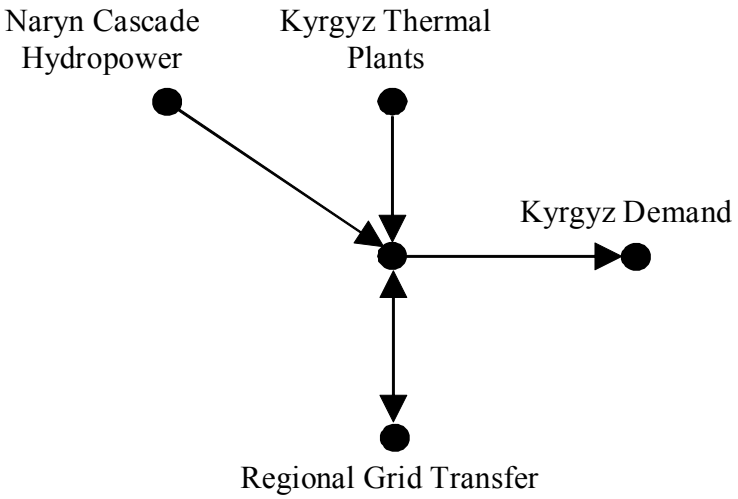


Figure 3.6.1. Power network for reservoir operation fask, including power cascade

The objective function can be formulated as

$$Z = \sum_{crops} gm(crop) * [A_2(crop) + A_2(crop)]$$
$$-\sum_{t} \{C_{thermal} * P_{thermal}(t) + C_{hydro} * P_{hydro}(t) + C_{transfer} * P_{transfer}(t) + [P(t) - E_{dem}(t)]^2\}$$

where C_{hydro} = 0.001 \$/kWh is the cost to produce hydropower, $C_{thermal}$ = 0.012 \$/kWh is the cost to produce thermal power, $C_{transfer}$ = 0.013 \$/kWh is the cost to transfer power to or from the regional grid. The Kyrgyz power demand and the Kyrgyz thermal power generation capacity is limited according to the values in Table 3.6.1.

Table 3.6.1. Electricity Data for Rerervoir Operation Problem with Power.

		Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Electricity Demand	GWh	161 4	130 1	131 8	881	609	51 6	52 9	528	50 9	69 7	119 6	152 2
Thermal Generation (Max)	GWh	345	345	345	100	45	45	45	45	45	34 5	345	345

3.7. (adapted from Loucks et al., Problem 5.14) Create and solve a GAMS model to find the solution which maximizes the total annual power production of the two reservoirs in series system shown in Figure 3.7.1.

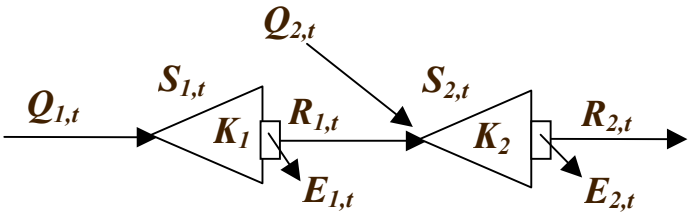


Figure 3.7.1. Two reservoirs in series.

The releases $R_{1,t}$ from the upstream reservoir (reservoir 1) plus the unregulated incremental flow $Q_{2,t}$ constitute the inflow to the downstream reservoir (reservoir 2). The flows $Q_{1,t}$ into the upstream reservoir in each of four seasons along with the incremental flows $Q_{2,t}$ and constraints on reservoir releases are given in the accompanying two tables:

Reservoir 1 (Upstream)

Season t	Inflow, $Q_{1,t}$	Minimum Release, $R_{1,t}^{\min}$	Maximum Release Through Turbines $R_{1,t}^{\max}$
1	60	20	90
2	40	30	90
3	80	20	90
4	120	20	90

Reservoir 2 (Downstream)

Season t	Incremental Flow, $Q_{2,t}$	Minimum Release, $R_{2,t}^{\min}$	Maximum Release Through Turbines, $R_{2,t}^{\max}$
1	50	30	140
2	30	40	140
3	60	30	140
4	90	30	140

Note that there is a limit on the quantity of water that can be released through the turbines for energy production in any season due to the limited capacity of the power plant and the desire to produce hydropower during periods of peak demand.

Additional information that affects the operation of the two reservoirs are limitations on the fluctuations in the pool levels (head) and the storage-head relationships:

Item	Reservoir 1 (Upstream)	Reservoir 2 (Downstream)
Maximum Head	$h_{1,t}^{\max} = 70$ m	$h_{2,t}^{\max} = 90$ m
Minimum Head	$h_{1,t}^{\min} = 30$ m	$h_{2,t}^{\min} = 60$ m
Capacity	$K_1 = 150\text{x}10^6$ m^3	$K_2 = 400\text{x}10^6$ m^3
Storage - Head Relationship	$h_{1,t}(S_{1,t}) = h_{1,t}^{\max}\left(\frac{S_{1,t}}{K_1}\right)^{0.64}$	$h_{2,t}(S_{2,t}) = h_{2,t}^{\max}\left(\frac{S_{2,t}}{K_2}\right)^{0.62}$

Assume that the efficiencies of the hydropower units are both 70%, independent of either flow through the turbines or the head on the turbines. In calculating the energy produced in any season, use the average head during that season

$$\bar{h}_{i,t} = 0.5 * [h_{i,t}(S_{i,t}) + h_{i,t}(S_{i,t+1})], \quad i = 1,2$$

3.8. Assume that stream flow ($\hat{S}$) is 26 million m^3 per unit of time and there is an interstate agreement ($\bar{S}$) calling for 14.5 million m^3 Also, initially there are two users on the river diverting S_2 and S_3 million m^3. The benefits to each user are:

$$\text{User 2:} \quad B_2(S_2) = a_2 S_2 + b_2 S_2^2 = 150 S_2 - 5 S_2^2$$
$$\text{User 3:} \quad B_3(S_3) = a_3 S_3 + b_3 S_3^2 = 18 S_3 - 0.6 S_3^2$$

In addition, both users have the same return flow coefficient, $R_2 = R_3 = 0.5$. Write an optimization model to determine an efficient allocation that results in maximizing the value of water use in the basin and respects the interstate compact. What are the water allocations and the benefits to each user?

3.9. Now, assume that an additional user wants to divert S_1 million m^3 of water from the river. User 1's benefit function is identical to User 2's benefit, i.e.,

$$\text{User 1:} \quad B_1(S_1) = a_1 S_1 + b_1 S_1^2 = 150 S_1 - 5 S_1^2$$

User 1 also has a return coefficient of $R_1 = 0.5$. Modify your optimization model to determine a new efficient allocation that results in maximizing the value of water use for all three users in the basin and respects the interstate compact. What are the water allocations and the benefits to each user?

3.10. Assume that the solution to Part (1) represents the initial water rights of Users 2 and 3, using the results obtained in Part (2), what is the minimum payment that User 1 should pay to Users 2 and 3 in order to divert water from the river? What are the resulting net benefits to each of the three users?

3.11. Assume that the system described in Problem (2) is modeled as a water market where users 1, 2, and 3 have water right allocations, 0.0, 4.0, and 7.5, respectively. Develop a model which will determine the optimal use of water by each user, assuming that the users are free to trade their water rights according to the model structure described in Section 3.2.5 above.

4. OPTIMAL MANAGEMENT OF GROUNDWATER

4.1 Finite-Difference Method

Analytical solutions are difficult for situations where you have irregular boundaries, various different boundary conditions in a region, heterogeneous and anisotropic porous media properties, or multiple phases. Often the aquifers and subsurface environments we want to model exhibit all of these characteristics. So what can we do? Numerical solutions to the governing equations of flow and mass transport in porous media provide a useful and convenient method of handling these complications. In this section, we will discuss the most common technique used to model groundwater flow and mass transport, the finite-difference method.

4.2 Numerical Differentiation

The forward Taylor series expansion of $f(x)$ in the neighborhood of a point x is

$$f(x+h) = f(x)+hf'(x)+\frac{h^2}{2!}f''(x)+\frac{h^3}{3!}f'''(x)+\cdots \tag{4.2.1}$$

where h is a small number. The backward Taylor series expansion of $f(x)$ about x can be written as

$$f(x-h) = f(x)-hf'(x)+\frac{h^2}{2!}f''(x)-\frac{h^3}{3!}f'''(x)+\frac{h^4}{4!}f^{iv}(x)+\cdots \tag{4.2.2}$$

Solving the forward equation for the first derivative, $f'(x)$, we have

$$f'(x) = \frac{f(x+h)-f(x)}{h}-\frac{h}{2}f''(x)-\frac{h^2}{6}f'''(x)+\cdots \tag{4.2.3}$$

or

$$\begin{aligned} f'(x) &= \frac{f(x+h)-f(x)}{h}+\vartheta(h) \\ &= \frac{f_{i+1}-f_i}{h}+\vartheta(h) \end{aligned} \tag{4.2.4}$$

where we will use the index notation $f(x) = f_i$ and $f(x+h) = f_{i+1}$ (see Figure 4.2.1).

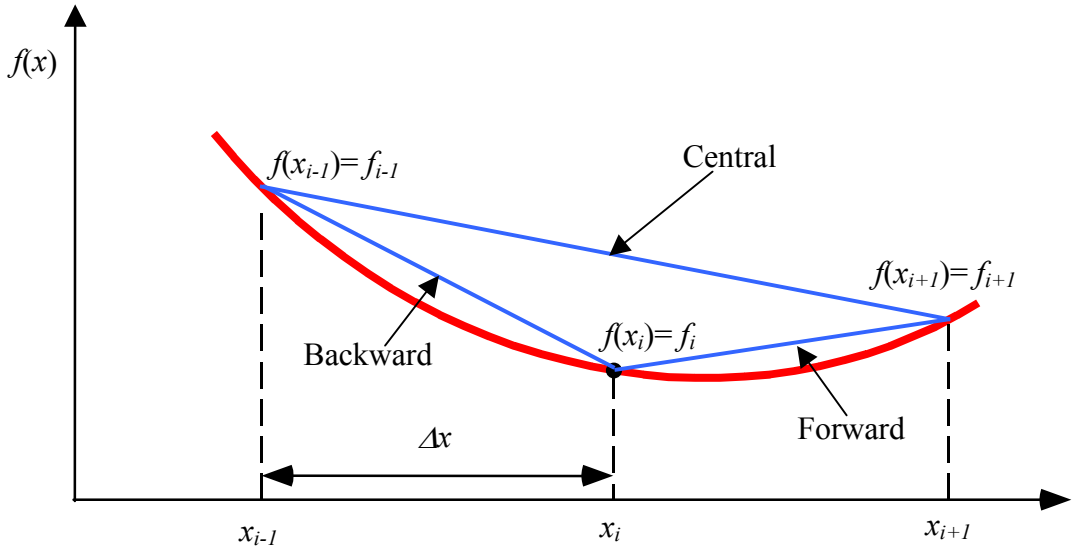


Figure 4.2.1. Forward, backward and central finite-difference approximations.

The error in this forward finite-difference approximation of the first derivative is

$$\vartheta(h) = -\frac{h}{2} f''(\bar{x}) \tag{4.2.5}$$

where $x \le \bar{x} \le x + h$. Similarly, we can solve the backward equation for the first derivative and we get

$$\begin{aligned} f'(x) &= \frac{f(x)-f(x-h)}{h} + \vartheta(h) \\ &= \frac{f_i - f_{i-1}}{h} + \vartheta(h) \end{aligned} \tag{4.2.6}$$

which is a backward finite-difference approximation of the first derivative and it has the same (first-order) error as the forward approximation.

If we subtract the forward and backward Taylor series expressions, we have

$$f(x+h)-f(x-h) = 2hf'(x) + \frac{h^3}{3!} f'''(x) + \cdots \tag{4.2.7}$$

or, solving for $f'(x)$

$$f'(x) = \frac{f(x+h)-f(x-h)}{2h} + \vartheta(h^2)$$
$$= \frac{f_{i+1}-f_{i-1}}{2h} + \vartheta(h^2) \qquad (4.2.8)$$

which is the central finite-difference approximation for the first derivative. An expression for the second derivative can be obtained by adding the forward and backward Taylor series expansions

$$f(x+h) = f(x)+hf'(x)+\frac{h^2}{2!}f''(x)+\frac{h^3}{3!}f'''(x)+\frac{h^4}{4!}f^{iv}(x)+\cdots \qquad (4.2.8)$$

and

$$f(x-h) = f(x)-hf'(x)+\frac{h^2}{2!}f''(x)-\frac{h^3}{3!}f'''(x)+\frac{h^4}{4!}f^{iv}(x)+\cdots \qquad (4.2.9)$$

to get

$$f(x+h)+f(x-h) = 2f(x)+h^2f''(x)+\frac{h^4}{12}f^{iv}(x)+\cdots \qquad (4.2.10)$$

Solving for $f''(x)$, we have

$$f''(x) = \frac{f(x+h)-2f(x)+f(x-h)}{h^2}$$
$$= \frac{f_{i+1}-2f_i+f_{i-1}}{h^2} + \vartheta(h^2) \qquad (4.2.11)$$

4.3 Grids and Discretrization

With the finite-difference method the governing partial differential equation (PDE) is replaced by a numerical approximation. In this process the continuous derivatives of the PDE are replaced by discrete approximations. The result of this discretization is a set of simultaneous algebraic equations that must be solved for the values of the unknown variables at discrete locations in the modeled domain. In order to accomplish the descretization process a mesh or grid must be defined that covers the domain. The grid consists of a series of intersecting, orthogonal, straight lines such as is illustrated in Figure 4.3.1. The unknown state variable(s) are then solved for at either the intersections of the gridlines (mesh-centered) or at the centers of the gridblocks (block-centered).

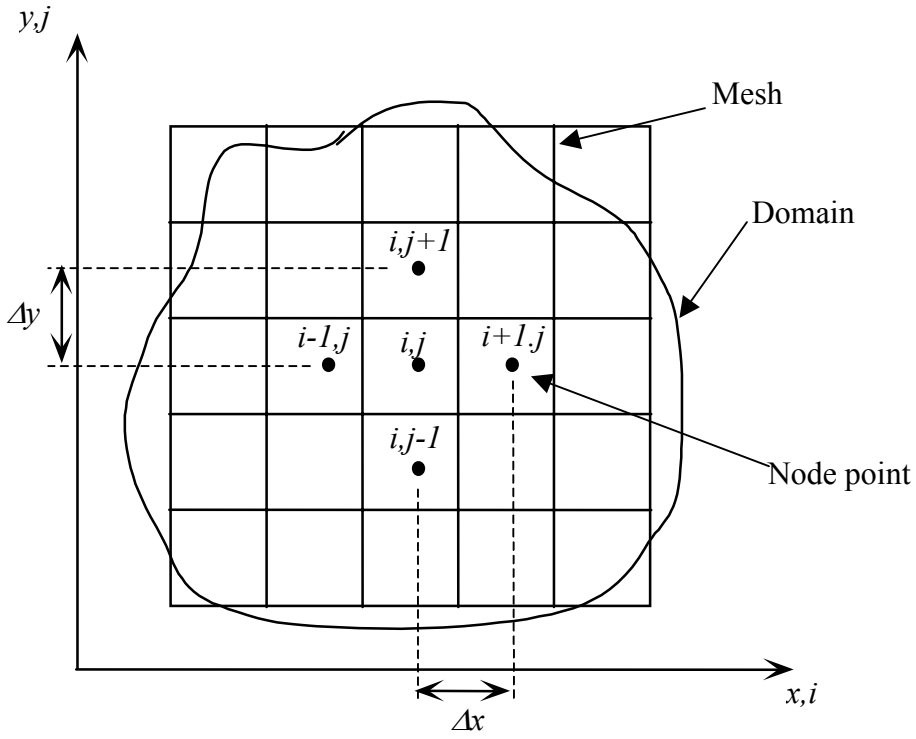


Figure 4.3.1. Finite-difference mesh.

4.4 One-Dimensional Flow

Steady-state flow

Consider steady, one-dimensional flow in a confined aquifer shown in Figure 4.4.1.

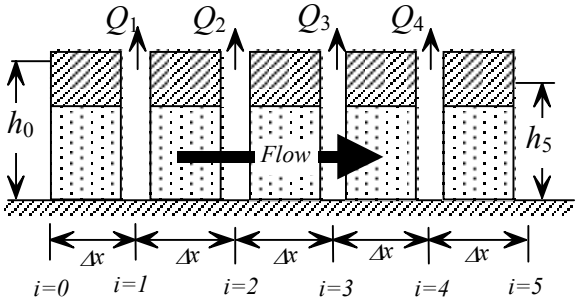


Figure 4.4.1. One-dimensional example aquifer.

The governing equation is

$$\frac{d}{dx}\left(T\frac{dh}{dx}\right) = Q \tag{4.4.1}$$

where T [L^2/T] is the transmissivity of the aquifer and $Q(x)$ [L^3/T/L^2 = L/T] is the pumping rate per unit area of aquifer. When the transmissivity is homogeneous, and we apply a finite-difference approximation to the second derivative, we have

$$\frac{\left(T\frac{\partial h}{\partial x}\right)_{i+1/2.j} - \left(T\frac{\partial h}{\partial x}\right)_{i-1/2.j}}{\Delta x} - Q_{i,j} = 0 \tag{4.4.2}$$

$$\frac{\left(T\frac{h_{i+1,j} - h_{i,j}}{\Delta x}\right) - \left(T\frac{h_{i,j} - h_{i\text{-}1,j}}{\Delta x}\right)}{\Delta x} - Q_{i,j} = 0 \tag{4.4.3}$$

$$A_{i,j}(h_{i+1,j} - h_{i,j}) + B_{i,j}(h_{i,j} - h_{i\text{-}1,j}) - Q_{i,j} = 0 \tag{4.4.4}$$

where

$$A_{i,j} = \frac{T}{\Delta x^2},\ B_{i,j} = \frac{-T}{\Delta x^2} \tag{4.4.5}$$

and Q_i is the source/sink rate for a particular node or cell. This finite-difference equation must be written for each node i where the head is unknown (nodes 1 – 4 in Figure 4.4.1).

Example - Steady Flow in a 1-D Homogeneous System

(adapted from Mays and Tung, Problem 8.3.1)

Consider the situation where the transmissivity of the aquifer is T = 1000 m^2/day, the well are Δx = 80m apart, the heads on the left- and right-hand boundaries are constant values of 40m and 35m, respectively. The objective of the optimization model is to maximize the head distribution in the aquifer while supplying a minimum pumping rate of *Demand* = 500 m^3/d/m^2. The optimization model can be written as

$$\begin{aligned}
&\textit{Maximize} \quad \sum_{i=1}^{4} h_i \\
&\textit{Subject to} \\
&\qquad \sum_{i=1}^{4} Q_i \geq \textit{Demand} \\
&\qquad A_i(h_{i+1} - h_i) + B_i(h_i - h_{i\text{-}1}) - Q_i = 0 \qquad i = 1,\ldots,4 \\
&\qquad h_i = 40 \qquad i = 0 \\
&\qquad h_i = 35 \qquad i = 5
\end{aligned} \tag{4.4.6}$$

The GAMS code for solving this model is

```
SET
         I    /1*4/;
SCALAR
         T    transmissivity      /1000/
         DX   cell width          /80/
         QMIN minimum pumping     /10/
         H0   left boundary head  /40/
         H5   right boundary head /35/
         a    constant            ;

a=T/(DX*DX);

VARIABLES
         OBJ   objective function;

POSITIVE VARIABLE
         H(I)  head at well I
         Q(I)  production rate at well I;

EQUATIONS
         FD(I)      Finite difference equation,
         PRODRATE   Sum of pumping rates,
         TOTALHEAD  Sum of heads;

FD(I)..    a*H(I-1)$(ORD(I) gt 1)
         - 2*a*H(I)
         + a*H(I+1)$(ORD(I) lt 4)
         - Q(I)
     =E= - a*H0$(ORD(I) = 1)
         - a*H5$(ORD(I) = 4);

PRODRATE..  sum(I,Q(I)) =G= QMIN;

TOTALHEAD.. sum(I,H(I)) =E= OBJ

MODEL GW1 /ALL/ ;

SOLVE GW1 USING LP MAXIMIZING OBJ ;

file GW1_Out /GW1.txt/
put GW1_Out
put '               Head  Pumping',  put /
loop( I,
        put I.tl,
        put H.l(I):8.1,
        put Q.l(I):8.1,
        put /)
```

The solution of the model is:

```
Node            Head   Pumping
0              40.00
1              11.00    3.96
2               7.33    0.00
3               3.67    0.00
4               0.00    6.04
5              35.00
```

4.5 Two-Dimensional Flow

Steady-State flow

Consider the steady, two-dimensional flow in a confined aquifer shown in Figure 4.5.1. The governing equation is

$$\frac{\partial\left(-Q^x\right)}{\partial x}+\frac{\partial\left(-Q^y\right)}{\partial y}=$$
$$\frac{\partial}{\partial x}\left(T\frac{\partial h}{\partial x}\right)+\frac{\partial}{\partial y}\left(T\frac{\partial h}{\partial y}\right)=Q \tag{4.5.1}$$

where T [L^2/T] is the transmissivity of the aquifer and Q [$L^3/T/L^2 = L/T$] is the source/sink flow rate per unit area of aquifer.

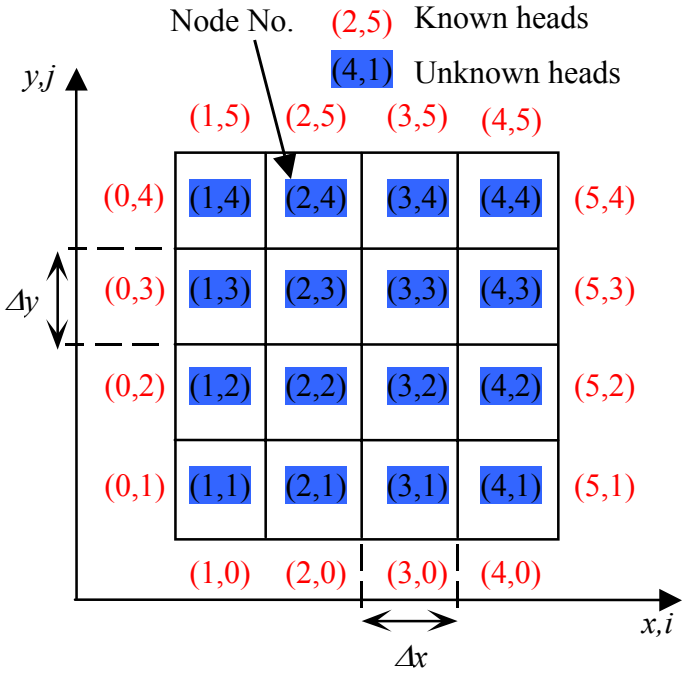


Figure 4.5.1. Two-dimensional example finite-difference grid.

Considering the cell (i,j) (see Figure 4.5.2) we can see that the flows in and out of the cell are represented by Darcy's Law written for aquifer flow. When the transmissivity is homogeneous, and we apply a finite-difference approximation to the second derivatives, we have

$$\frac{\left(T\frac{\partial h}{\partial x}\right)_{i+1/2.j}-\left(T\frac{\partial h}{\partial x}\right)_{i-1/2.j}}{\Delta x}+\frac{\left(T\frac{\partial h}{\partial y}\right)_{i.j+1/2}-\left(T\frac{\partial h}{\partial y}\right)_{i.j-1/2}}{\Delta y}-Q_{i,j}=0 \tag{4.5.2}$$

$$\frac{\left(T\frac{h_{i+1,j}-h_{i,j}}{\Delta x}\right)-\left(T\frac{h_{i,j}-h_{i-1,j}}{\Delta x}\right)}{\Delta x}+\frac{\left(T\frac{h_{i,j+1}-h_{i,j}}{\Delta y}\right)-\left(T\frac{h_{i,j}-h_{i,j-1}}{\Delta y}\right)}{\Delta y}-Q_{i,j}=0 \tag{4.5.3}$$

$$A_{i,j}(h_{i+1,j}-h_{i,j})+B_{i,j}(h_{i,j}-h_{i-1,j})+C_{i,j}(h_{i,j+1}-h_{i,j})+D_{i,j}(h_{i,j}-h_{i,j-1})-Q_{i,j}=0 \tag{4.5.4}$$

where

$$A_{i,j}=\frac{T}{\Delta x^2},\ B_{i,j}=\frac{-T}{\Delta x^2},\ C_{i,j}=\frac{T}{\Delta y^2},\ D_{i,j}=\frac{-T}{\Delta y^2} \tag{4.5.5}$$

This equation must be solved for each node (*i,j*) on the interior of the solution domain.

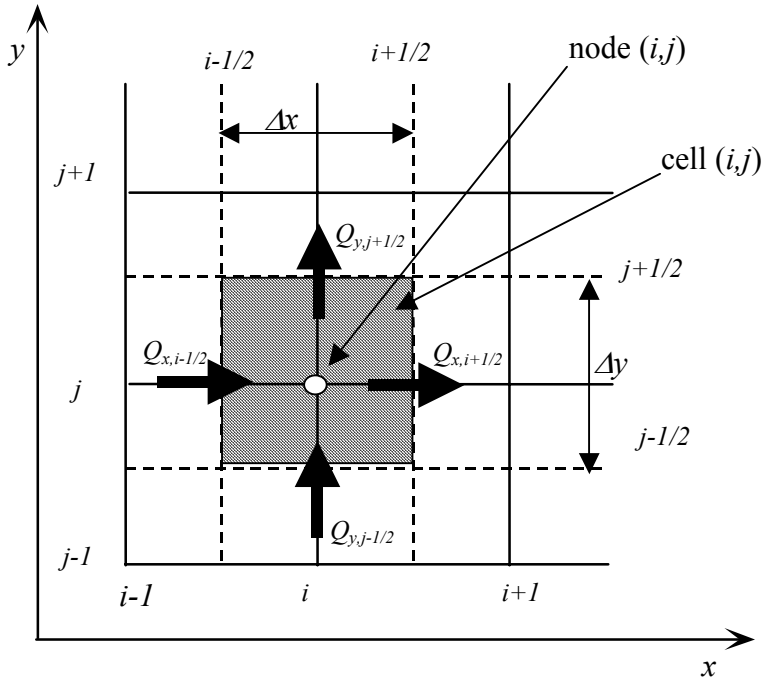


Figure 4.5.2. Finite-difference cell in two dimensional flow.

Boundary Conditions

Two types of boundary conditions are often encountered in groundwater flow problems:

(1) Constant or prescribed head (Dirichlet) conditions

$$h=f_1(x,y,t)\ on\ C_1 \tag{4.5.6}$$

In the previous example, if we write out the finite-difference equation for each node (*i,j*) where the head is unknown (nodes (1,1) - (4,4)), we have, e.g., at node (1,1)

$$A_{1,1}(h_{2,1}-h_{1,1})+B_{1,1}(h_{1,1}-h_{0.1})+C_{1,1}(h_{1,2}-h_{1,1})+D_{1,1}(h_{1,1}-h_{1,0})-Q_{1,1}=0 \tag{4.5.7}$$

where the heads $h_{0,1}$ and $h_{1,0}$ are known from the boundary conditions. For node (2,3), we have

$$A_{2,3}(h_{3,3}-h_{2,3})+B_{2,3}(h_{2,3}-h_{1,3})+C_{2,3}(h_{2,4}-h_{2,3})+D_{2,3}(h_{2,3}-h_{2,2})-Q_{2,3}=0 \tag{4.5.8}$$

(2) Prescribed flux (Neuman) conditions

$$q_n = f_2(x,y,t)\ on\ C_2 \tag{4.5.9}$$

or

$$\frac{\partial h}{\partial n} = f_2(x,y,t)\ on\ C_2 \tag{4.5.10}$$

Consider the case where we have constant head boundary conditions on the west and east sides of an aquifer and no-flow boundary conditions on the north and south sides as shown in Figure 4.5.3.

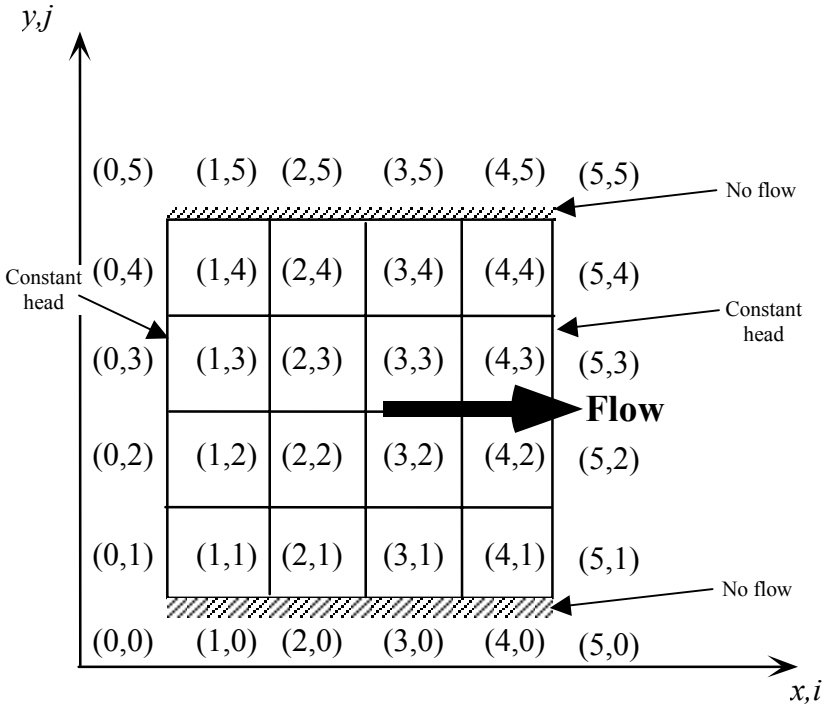


Figure 4.5.3. Two-dimensional example with no-flow boundary conditions.

The north and south side no-flow conditions can be written as

$$\left.\frac{\partial h}{dy}\right|_{i,\,j=1} = 0 \quad \text{and} \quad \left.\frac{\partial h}{dy}\right|_{i,\,j=4} = 0 \tag{4.5.11}$$

These conditions can easily be approximated by finite-differences, using a central difference approximation as

$$\frac{h_{i,0} - h_{i,1}}{\Delta y} = 0 \quad \text{and} \quad \frac{h_{i,5} - h_{i,4}}{\Delta y} = 0 \tag{4.5.12}$$

or

$$h_{i,0} = h_{i,1} \qquad \text{and} \qquad h_{i,5} = h_{i,4} \tag{4.5.13}$$

The resulting finite-difference equations for nodes along these no-flow boundaries are:

for j = 1 (south boundary):

$$A_{i,1}(h_{i+1,1} - h_{i,1}) + B_{i,1}(h_{i,1} - h_{i-1,1}) + C_{i,1}(h_{i,2} - h_{i,1}) + D_{i,1}(h_{i,1} - h_{i,0}) - Q_{i,1} = 0 \tag{4.5.14}$$

for j = 4 (north boundary):

$$A_{i,4}(h_{i+1,4} - h_{i,4}) + B_{i,4}(h_{i,4} - h_{i-1,4}) + C_{i,4}(h_{i,5} - h_{i,4}) + D_{i,4}(h_{i,4} - h_{i,3}) - Q_{i,4} = 0 \tag{4.5.15}$$

Substituting in the boundary information, we have

for j = 1 (south boundary):

$$A_{i,1}(h_{i+1,1} - h_{i,1}) + B_{i,1}(h_{i,1} - h_{i-1,1}) + C_{i,1}(h_{i,2} - h_{i,1}) + \ 0 \ - Q_{i,1} = 0 \tag{4.5.16}$$

for j = 4 (north boundary):

$$A_{i,4}(h_{i+1,4} - h_{i,4}) + B_{i,4}(h_{i,4} - h_{i-1,4}) + \ 0 \ + D_{i,4}(h_{i,4} - h_{i,3}) - Q_{i,4} = 0 \tag{4.5.17}$$

Example: Steady Flow in 2-D Homogeneous System

Consider the flow in the example aquifer shown in Figure 4.5.2, where the mesh size is 10m, western and eastern boundaries are held fixed at 100m and 50m respectively, the transmissivity is 200 m^2/day, and the maximum pumping is limited to 40 m^3/day/m^2. The optimization model for this problem can be written as

$$\textit{Minimize} \quad \sum_{i=0}^{9} \sum_{j=1}^{8} Q_{i,j}$$

Subject to

$$\sum_{i=0}^{9} \sum_{j=1}^{8} Q_{i,j} \geq Demand$$

$$\begin{aligned}
&A_{i,j}(h_{i+1,j} - h_{i,j}) + B_{i,j}(h_{i,j} - h_{i\text{-}1,j}) + \\
&\qquad C_{i,j}(h_{i,j+1} - h_{i,j}) + D_{i,j}(h_{i,j} - h_{i,j\text{-}1}) - Q_{i,j} = 0 && i = 1,...,8;\ j = 1,...8 \\
&A_{i,1}(h_{i+1,1} - h_{i,1}) + B_{i,1}(h_{i,1} - h_{i\text{-}1,1}) + C_{i,1}(h_{i,2} - h_{i,1}) = 0 && i = 0;\ j = 1,...8 \\
&A_{i,4}(h_{i+1,4} - h_{i,4}) + B_{i,4}(h_{i,4} - h_{i\text{-}1,4}) + D_{i,4}(h_{i,4} - h_{i,3}) = 0 && i = 9;\ j = 1,...,8
\end{aligned}$$

$$\begin{aligned}
&h_{i,j} = 100 && i = 0,...,9;\ j = 0 \\
&h_{i,j} = 50 && i = 0....,9;\ j = 9
\end{aligned}$$

(4.5.18)

The GAMS code to solve this linear program is given below. The reader should note that the grid orientation is rotated so that the origin is in the upper left-hand corner and the *x* axis runs down the page.

```
SETS
        I row index of model cells        / I0*I10 /
        J column index of model cells     / J0*J10 /

SCALARS
         DX    grid size x (m)                    /  10 /
         DY    grid size y (m)                    /  10 /
         HBW   boundary head west (m)             / 100 /
         HBE   boundary head east (m)             / 100 /
         HBN   boundary head north (m)            / 100 /
         HBS   boundary head south (m)            / 100 /
         DEM   water demand (1000 m3 per day)     / 400 /
         QMAX  max pumping rate (m3 per d per m2) / 10 /
         QMIN  min pumping rate (m3 per d per m2) /   0 /
         T     Transmissivity m2 per day          / 200 /
          ;

* finite-difference coefficients
PARAMETER
        A(I,J)   finite-difference coefficient
        B(I,J)   finite-difference coefficient
        C(I,J)   finite-difference coefficient
        D(I,J)   finite-difference coefficient;

        A(I,J) =  T/(DX*DX) ;
        B(I,J) = -T/(DX*DX) ;
        C(I,J) =  T/(DY*DY) ;
        D(I,J) = -T/(DY*DY) ;

* finite-difference cell types
TABLE CELL(I,J) cell type

* -2 = north no-flow bndy
* -1 = south no-flow bndy
```

```
*  0 = interior cells
*  1 = west fixed potential
*  2 = east fixed potential
*  3 = north fixed potential
*  4 = south fixed potential

    J0  J1  J2  J3  J4  J5  J6  J7  J8  J9 J10
I0   1   3   3   3   3   3   3   3   3   3   2
I1   1   0   0   0   0   0   0   0   0   0   2
I2   1   0   0   0   0   0   0   0   0   0   2
I3   1   0   0   0   0   0   0   0   0   0   2
I4   1   0   0   0   0   0   0   0   0   0   2
I5   1   0   0   0   0   0   0   0   0   0   2
I6   1   0   0   0   0   0   0   0   0   0   2
I7   1   0   0   0   0   0   0   0   0   0   2
I8   1   0   0   0   0   0   0   0   0   0   2
I9   1   0   0   0   0   0   0   0   0   0   2
I10  1   4   4   4   4   4   4   4   4   4   2;

* Use for fixed head all around
*I0   1   3   3   3   3   3   3   3   3   3   2
*I10  1   4   4   4   4   4   4   4   4   4   2;
* Use for fixed head east and west
*I0   1  -2  -2  -2  -2  -2  -2  -2  -2  -2   2
*I10  1  -1  -1  -1  -1  -1  -1  -1  -1  -1   2;

VARIABLES
        OBJ      objective value
        Q(I,J)   pumping in cell ij [m3 per day]
        ;

POSITIVE VARIABLES
        H(I,J)   head in cell ij
        ;

*        boundary conditions
* Use for fixed head east and west
         H.fx(I,J)$(CELL(I,J) = 1) = HBW ;
         H.fx(I,J)$(CELL(I,J) = 2) = HBE ;
* Use for fixed head all around
         H.fx(I,J)$(CELL(I,J) = 3) = HBN ;
         H.fx(I,J)$(CELL(I,J) = 4) = HBS ;

*        maximum/minimum pumping contrtraints
         Q.lo(I,J)$(CELL(I,J) le 0) = QMIN;
         Q.up(I,J)$(CELL(I,J) le 0) = QMAX;

EQUATIONS
        CONT(I,J)     finite-difference continuity eqn.
        DEMAND        demand constraint eqn.
        TCOST         expected total cost ;

*        continuity equation (including no-flow boundaries)
CONT(I,J) ..
     ( A(I,J)*(H(I+1,J)-H(I,J))
      +B(I,J)*(H(I,J)  -H(I-1,J))
      +C(I,J)*(H(I,J+1)-H(I,J))
      +D(I,J)*(H(I,J)  -H(I,J-1)))$(CELL(I,J) eq 0)

    +( A(I,J)*(H(I+1,J)-H(I,J))
*      +B(I,J)*(H(I,J)  -H(I,J))
```

```
          +C(I,J)*(H(I,J+1)-H(I,J))
          +D(I,J)*(H(I,J)  -H(I,J-1)))$(CELL(I,J) eq -2)

        +(
*        A(I,J)*(H(I,J)  -H(I,J))
          +B(I,J)*(H(I,J)  -H(I-1,J))
          +C(I,J)*(H(I,J+1)-H(I,J))
          +D(I,J)*(H(I,J)  -H(I,J-1)))$(CELL(I,J) eq -1)
          -Q(I,J)
        =E= 0.0;

*        demand constraint
DEMAND ..  SUM((I,J), Q(I,J)) =E= DEM ;

*        objective function
TCOST ..  OBJ =E= (SUM((I,J), H(I,J))) ;

MODEL PUMPING least-cost pumping model /ALL/ ;

SOLVE PUMPING USING LP MAXIMIZING OBJ ;

File GW2D /GW2D_output.txt/
Put GW2D;
PUT ' Objective value = '; Put OBJ.l;
Put /;
Put 'Pumping'/; Put '         ';
Loop(J, Put J.tl:6;) Put /;
Loop(I, Put I.tl:6; Loop(J, Put Q.L(I, J):6:1;); Put /;);
Put /;
Put 'Head'/; Put '         ';
Loop(J,Put J.tl:6;) Put /;
Loop(I, Put I.tl:6; Loop(J, Put H.L(I, J):6:1;); Put /;);
Put /;
```

The results of solving this model are shown in the following table and figure.

```
Objective value =     11566.11
Pumping
         J0     J1     J2     J3     J4     J5     J6     J7     J8     J9     J10
I0        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I1        0.0   10.0   10.0   10.0   10.0   10.0   10.0   10.0   10.0   10.0    0.0
I2        0.0   10.0   10.0    0.0    0.0    0.0    0.0    0.0   10.0   10.0    0.0
I3        0.0   10.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0   10.0    0.0
I4        0.0   10.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0   10.0    0.0
I5        0.0   10.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0   10.0    0.0
I6        0.0   10.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0   10.0    0.0
I7        0.0   10.0   10.0    0.0    0.0    0.0    0.0    0.0   10.0   10.0    0.0
I8        0.0   10.0   10.0   10.0    0.0    0.0    0.0   10.0   10.0   10.0    0.0
I9        0.0   10.0   10.0   10.0   10.0   10.0   10.0   10.0   10.0   10.0    0.0
I10       0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0

Head
         J0     J1     J2     J3     J4     J5     J6     J7     J8     J9     J10
I0     100.0  100.0  100.0  100.0  100.0  100.0  100.0  100.0  100.0  100.0  100.0
I1     100.0   96.0   94.6   94.5   94.5   94.5   94.5   94.5   94.6   96.0  100.0
I2     100.0   94.6   92.9   93.7   94.0   94.0   94.0   93.7   92.9   94.6  100.0
I3     100.0   94.4   93.7   93.6   93.7   93.7   93.7   93.6   93.7   94.4  100.0
I4     100.0   94.3   93.7   93.5   93.4   93.4   93.4   93.5   93.7   94.3  100.0
I5     100.0   94.1   93.4   93.1   93.0   93.0   93.0   93.1   93.4   94.1  100.0
I6     100.0   93.8   92.7   92.5   92.6   92.7   92.6   92.5   92.7   93.8  100.0
```

```
I7     100.0  93.4  90.9  91.7  92.3  92.5  92.3  91.7  90.9  93.4 100.0
I8     100.0  93.8  91.1  90.9  92.4  92.8  92.4  90.9  91.1  93.8 100.0
I9     100.0  95.6  93.8  93.3  93.6  93.7  93.6  93.3  93.8  95.6 100.0
I10    100.0 100.0 100.0 100.0 100.0 100.0 100.0 100.0 100.0 100.0 100.0
```

Heterogeneous Systems

Consider the a steady, two-dimensional flow in a heterogeneous, anisotropic, confined aquifer. The governing equation is (see Figure 4.5.3)

$$\frac{\partial\left(-Q^x\right)}{\partial x}+\frac{\partial\left(-Q^y\right)}{\partial y}=$$
$$\frac{\partial}{\partial x}\left(T^x\frac{\partial h}{\partial x}\right)+\frac{\partial}{\partial y}\left(T^y\frac{\partial h}{\partial y}\right)=Q \tag{4.5.19}$$

where T^x and T^y [L^2/T] are transmissivities in the x and y directions, respectively. When we apply a finite-difference approximation to the derivatives, we have

$$\frac{\left(T^x\frac{\partial h}{\partial x}\right)_{i+1/2,j}-\left(T^x\frac{\partial h}{\partial x}\right)_{i-1/2,j}}{\Delta x}+\frac{\left(T^y\frac{\partial h}{\partial y}\right)_{i,j+1/2}-\left(T^y\frac{\partial h}{\partial y}\right)_{i,j-1/2}}{\Delta y}-Q_{i,j}=0 \tag{4.5.20}$$

Applying a finite difference approximation to the remaining derivatives results in

$$\frac{\left(T^x_{i+1/2,j}\frac{h_{i+1,j}-h_{i,j}}{\Delta x}\right)-\left(T^x_{i\text{-}1/2,j}\frac{h_{i,j}-h_{i\text{-}1,j}}{\Delta x}\right)}{\Delta x}$$
$$+\frac{\left(T^y_{i,j+1/2}\frac{h_{i,j+1}-h_{i,j}}{\Delta y}\right)-\left(T^y_{i,j\text{-}1/2}\frac{h_{i,j}-h_{i,j\text{-}1}}{\Delta y}\right)}{\Delta y}-Q_{i,j}=0 \tag{4.5.21}$$

where

$$T^x_{i+1/2,j}=2\frac{T^x_{i+1,j}T^x_{i,j}}{T^x_{i+1,j}+T^x_{i,j}}\quad\text{and}\quad T^y_{i,j+1/2}=2\frac{T^y_{i,j+1}T^y_{i,j}}{T^y_{i,j+1}+T^y_{i,j}} \tag{4.5.22}$$

are the harmonic averages of the x- and y-direction transmissivities, respectively, between cells (i,j) and $(i+1,j)$ and between cells (i,j) and $(i,j+1)$. We can write this equation as

$$A_{i,j}(h_{i+1,j}-h_{i,j})+B_{i,j}(h_{i,j}-h_{i\text{-}1,j})+C_{i,j}(h_{i,j+1}-h_{i,j})+D_{i,j}(h_{i,j}-h_{i,j\text{-}1})-Q_{i,j}=0 \tag{4.5.23}$$

where

$$A_{i,j} = \frac{2}{\Delta x^2}\frac{T^x_{i+1,j}T^x_{i,j}}{T^x_{i+1,j}+T^x_{i,j}},\ B_{i,j} = \frac{-2}{\Delta x^2}\frac{T^x_{i-1,j}T^x_{i,j}}{T^x_{i-1,j}+T^x_{i,j}},$$
$$C_{i,j} = \frac{2}{\Delta y^2}\frac{T^y_{i,j+1}T^y_{i,j}}{T^y_{i,j+1}+T^y_{i,j}},\ D_{i,j} = \frac{-2}{\Delta y^2}\frac{T^y_{i,j11}T^y_{i,j}}{T^y_{i,j11}+T^y_{i,j}} \qquad (4.5.24)$$

If no flow boundaries exist in the system, then appropriate modifications, as described in the previous section, must be made to this system of equations.

Aside: Note that for an unequal grid spacing ($\Delta x_{i-1} \neq \Delta x_i \neq \Delta x_{i+1}$ and $\Delta y_{i-1} \neq \Delta y_i \neq \Delta y_{i+1}$), then we write

$$\frac{\left(T^x_{i+1/2,j}\frac{h_{i+1,j}-h_{i,j}}{\Delta x_{i+1/2}}\right)-\left(T^x_{i\text{-}1/2,j}\frac{h_{i,j}-h_{i\text{-}1,j}}{\Delta x_{i-1/2}}\right)}{\Delta x_i}$$

$$+\frac{\left(T^y_{i,j+1/2}\frac{h_{i,j+1}-h_{i,j}}{\Delta y_{j+1/2}}\right)-\left(T^y_{i,j\text{-}1/2}\frac{h_{i,j}-h_{i,j\text{-}1}}{\Delta y_{j-1/2}}\right)}{\Delta y_j}-Q_{i,j}=0$$

where $\Delta x_{i-1/2}$ is the distance between the centers of cells (i-1, j) and (i,j), and

$$T^x_{i+1/2,j} = 2\frac{T^x_{i,j}T^x_{i+1,j}}{T^x_{i,j}\Delta x_{i+1}+T^x_{i+1,j}\Delta x_i}\Delta x_{i+1/2}$$

Example – Steady Flow in a 2-D Heterogeneous System

```
SETS
        I row index of model cells        / I0*I10 /
        J column index of model cells     / J0*J10 /

SCALARS
         DX        grid size x (m)                / 10   /
         DY        grid size y (m)                / 10   /
         HBW       boundary head west (m)         / 100  /
         HBE       boundary head east (m)         / 50   /
```

```
         HBN       boundary head north (m)         / 100 /
         HBS       boundary head south (m)         / 100 /
         DEM       water demand (1000 m3 per day) / 1000 /
         QMAX  max pumping rate (m3 per d per m2) / 1000   /
         QMIN  min pumping rate (m3 per d per m2) /   0  /
          ;

TABLE T(I,J) isotropic transmissivity (m2 per day)

     J0  J1  J2  J3  J4  J5  J6  J7  J8  J9 J10
I0  205 185 190 175 160 185 195 190 175 160 165
I1  205 185 190 175 160 185 195 190 175 160 165
I2  260 235 215 200 175 185 195 190 175 160 165
I3  255 250 230 205 150 185 195 190 175 160 165
I4  210 225 220 195 145 185 195 190 175 160 165
I5  185 195 190 175 160 185 195 190 175 160 165
I6  160 205 215 200 165 185 195 190 175 160 165
I7  155 200 210 215 150 185 195 190 175 160 165
I8  170 215 200 195 155 185 195 190 175 160 165
I9  170 215 200 195 155 185 195 190 175 160 165
I10 170 215 200 195 155 185 195 190 175 160 165
;

* finite-difference cell types
TABLE CELL(I,J) cell type

* -2 = north no-flow bndy
* -1 = south no-flow bndy
*  0 = interior cells
*  1 = west fixed potential
*  2 = east fixed potential
*  3 = north fixed potential
*  4 = south fixed potential

    J0  J1  J2  J3  J4  J5  J6  J7  J8  J9 J10
I0   1  -2  -2  -2  -2  -2  -2  -2  -2  -2   2
I1   1   0   0   0   0   0   0   0   0   0   2
I2   1   0   0   0   0   0   0   0   0   0   2
I3   1   0   0   0   0   0   0   0   0   0   2
I4   1   0   0   0   0   0   0   0   0   0   2
I5   1   0   0   0   0   0   0   0   0   0   2
I6   1   0   0   0   0   0   0   0   0   0   2
I7   1   0   0   0   0   0   0   0   0   0   2
I8   1   0   0   0   0   0   0   0   0   0   2
I9   1   0   0   0   0   0   0   0   0   0   2
I10  1  -1  -1  -1  -1  -1  -1  -1  -1  -1   2
;

* Use for fixed head all around
*I0   1   3   3   3   3   3   3   3   3   3   2
*I10  1   4   4   4   4   4   4   4   4   4   2
* Use for fixed head east and west
*I0   1  -2  -2  -2  -2  -2  -2  -2  -2  -2   2
*I10  1  -1  -1  -1  -1  -1  -1  -1  -1  -1   2

* finite-difference coefficients
PARAMETER
        A(I,J)   finite-difference coefficient
        B(I,J)   finite-difference coefficient
        C(I,J)   finite-difference coefficient
        D(I,J)   finite-difference coefficient
```

```
         ;

         A(I,J) = 0.0;
         B(I,J) = 0.0;
         C(I,J) = 0.0;
         D(I,J) = 0.0;
A(I,J) =  (2*T(I+1,J)*T(I,J)/(T(I+1,J)+T(I,J))/(DX*DX))$(ord(I) ne card(I))
       +(   T(I,J)/(DX*DX))$(ord(I) eq card(I));
B(I,J) = (-2*T(I-1,J)*T(I,J)/(T(I-1,J)+T(I,J))/(DX*DX))$(ord(I) ne 0)
       +(   T(I,J)/(DX*DX))$(ord(I) eq 0) ;
C(I,J) =  (2*T(I,J+1)*T(I,J)/(T(I,J+1)+T(I,J))/(DY*DY))$(ord(J) ne card(J))
       +(   T(I,J)/(DY*DY))$(ord(J) eq card(J));
D(I,J) = (-2*T(I,J-1)*T(I,J)/(T(I,J-1)+T(I,J))/(DY*DY))$(ord(J) ne 0)
       +(   T(I,J)/(DY*DY))$(ord(J) eq 0);

VARIABLES
         OBJ     total cost of pumping
         ;

POSITIVE VARIABLES
         Q(I,J)   pumping in cell ij [1000 m3 per day]
         H(I,J)   head in cell ij
         ;

*         boundary conditions
* Use for fixed head east and west
          H.fx(I,J)$(CELL(I,J) = 1) = HBW ;
          H.fx(I,J)$(CELL(I,J) = 2) = HBE ;
* Use for fixed head all around
          H.fx(I,J)$(CELL(I,J) = 3) = HBN ;
          H.fx(I,J)$(CELL(I,J) = 4) = HBS ;

* maximum/minimum pumping contrtraints
          Q.lo(I,J)$(CELL(I,J) le 0) = QMIN;
          Q.up(I,J)$(CELL(I,J) le 0) = QMAX;

EQUATIONS
         CONT(I,J)      finite-difference continuity eqn.
         DEMAND         demand constraint eqn.
         OBJECTIVE      objective function ;

* continuity equation (including no-flow boundaries)
CONT(I,J) ..
      ( A(I,J)*(H(I+1,J)-H(I,J))
       +B(I,J)*(H(I,J)  -H(I-1,J))
       +C(I,J)*(H(I,J+1)-H(I,J))
       +D(I,J)*(H(I,J)  -H(I,J-1)))$(CELL(I,J) eq 0)
     +( A(I,J)*(H(I+1,J)-H(I,J))
*      +B(I,J)*(H(I,J)  -H(I,J))
       +C(I,J)*(H(I,J+1)-H(I,J))
       +D(I,J)*(H(I,J)  -H(I,J-1)))$(CELL(I,J) eq -2)
     +(
*       A(I,J)*(H(I,J)  -H(I,J))
       +B(I,J)*(H(I,J)  -H(I-1,J))
       +C(I,J)*(H(I,J+1)-H(I,J))
       +D(I,J)*(H(I,J)  -H(I,J-1)))$(CELL(I,J) eq -1)
       -Q(I,J)
     =E= 0.0;

* demand constraint
DEMAND ..  SUM((I,J), Q(I,J)) =E= DEM ;
```

```
* objective function (sum of pumping or cost of pumping)
OBJECTIVE ..  OBJ =E= (SUM((I,J), H(I,J))) ;

MODEL PUMPING least-cost pumping model /ALL/ ;

SOLVE PUMPING USING LP MAXIMIZING OBJ ;

File GW2D /GW2D_KXY_output.txt/
Put GW2D;
PUT ' Objective value = '; Put OBJ.l;
Put /;
Put 'Pumping'/; Put '         ';
Loop(J, Put J.tl:6;) Put /;
Loop(I, Put I.tl:6; Loop(J, Put Q.L(I, J):6:1;); Put /;);
Put /;
Put 'Head'/; Put '         ';
Loop(J,Put J.tl:6;) Put /;
Loop(I, Put I.tl:6; Loop(J, Put H.L(I, J):6:1;); Put /;);
Put /;
```

The results of running this model are:

```
Objective value =      7014.44
Pumping
         J0     J1     J2     J3     J4     J5     J6     J7     J8     J9     J10
I0        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I1        0.0  240.4    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I2        0.0  318.3    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I3        0.0  441.3    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I4        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I5        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I6        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I7        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I8        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I9        0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
I10       0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0

Head
         J0     J1     J2     J3     J4     J5     J6     J7     J8     J9     J10
I0     100.0   50.8   37.5   36.3   38.6   41.5   43.9   45.8   47.4   48.7   50.0
I1     100.0   12.4   25.7   32.8   38.0   41.8   44.4   46.2   47.7   48.9   50.0
I2     100.0    0.0   20.3   31.8   38.6   43.0   45.5   47.2   48.3   49.2   50.0
I3     100.0    0.0   25.4   36.3   42.2   45.8   47.6   48.6   49.3   49.7   50.0
I4     100.0   54.0   47.1   47.4   48.9   49.9   50.4   50.5   50.5   50.3   50.0
I5     100.0   74.7   63.3   58.5   56.1   54.5   53.5   52.6   51.8   50.9   50.0
I6     100.0   83.3   73.2   66.8   62.1   58.7   56.4   54.7   53.1   51.5   50.0
I7     100.0   87.6   78.6   72.1   66.5   62.0   58.9   56.5   54.3   52.1   50.0
I8     100.0   90.1   82.1   75.5   69.6   64.5   60.8   57.8   55.2   52.6   50.0
I9     100.0   91.3   84.1   77.6   71.5   66.1   62.1   58.8   55.8   52.9   50.0
I10    100.0   91.8   84.9   78.6   72.4   66.9   62.7   59.3   56.1   53.0   50.0
```

4.6 Transient Problems

In transient problems, time is added as an independent variable. Therefore, we need to consider the rate of release or uptake of water from or to storage in an aquifer. We also need to specify initial head conditions in the aquifer in order to find a solution of a problem.

If we consider the transient, two-dimensional flow in a confined aquifer. The governing equation is

$$\frac{\partial}{\partial x}\left(T^x \frac{\partial h}{\partial x}\right)+\frac{\partial}{\partial y}\left(T^y \frac{\partial h}{\partial y}\right)-Q=S\frac{\partial h}{\partial t} \tag{4.6.1}$$

where S is the aquifer storativity. When the transmissivity and storativity are heterogeneous, and we apply a finite-difference approximation to the second derivatives on the left-hand-side of this equation, we have

$$LHS_{i,j}=A_{i,j}(h_{i+1,j}-h_{i,j})+B_{i,j}(h_{i,j}-h_{i\text{-}1,j})+C_{i,j}(h_{i,j+1}-h_{i,j})+D_{i,j}(h_{i,j}-h_{i,j\text{-}1})-Q_{i,j} \tag{4.6.2}$$

Consider a backward finite difference approximation of the time difference on the right-hand-side

$$RHS_{i,j}^{t+1}=S_{i,j}\frac{h_{i,j}^{t+1}-h_{i,j}^{t}}{\Delta t} \tag{4.6.3}$$

where $h_{i,j}^{t}$ and $h_{i,j}^{t+1}$ are the heads in cell (i,j) at the time levels t, and $t+\Delta t$, respectively. Now, we have not specified the time level where we are evaluating the *LHS*. Using a weighted average of the LHS at the t and $t+\Delta t$ levels, we have

$$\theta LHS_{i,j}^{t+1}+(1-\theta)LHS_{i,j}^{t}=\frac{S_{i,j}}{\Delta t}\left(h_{i,j}^{t+1}-h_{i,j}^{t}\right) \tag{4.6.4}$$

where $0\le\theta\le 1$.

Various time-stepping schemes result from different selections of θ: e.g., **Explicit Scheme:** $\theta=0$**;** and the **Crank-Nicolson Scheme:** $\theta=1/2$**.**

One of the mst common is the **Fully Implicit Scheme:** $\theta=1$

$$\begin{aligned}A_{i,j}(h_{i+1,j}^{t+1}-h_{i,j}^{t+1})+B_{i,j}(h_{i,j}^{t+1}-h_{i-1,j}^{t+1})&\\ +C_{i,j}(h_{i,j+1}^{t+1}-h_{i,j}^{t+1})+D_{i,j}(h_{i,j}^{t+1}-h_{i,j-1}^{t+1})-Q_{i,j}^{t+1}&\\ =F_{i,j}(h_{i,j}^{t+1}-h_{i,j}^{t})&\end{aligned} \tag{4.6.5}$$

where

$$F_{i,j}=\frac{S_{i,j}}{\Delta t} \tag{4.6.7}$$

Example: Transient Flow in 2-D Homogeneous System

```
SETS
    I        row index of model cells           /I0*I10 /
    J        column index of model cells        /J0*J10 /
    L        time index                         /L0*L12/

SCALARS
    DX       grid size x (m)                    /10/
    DY       grid size y (m)                    /10/
    DT       time step size t (s) a month       /2.628E6/
    S        storage coefficient                /0.002/
    HBW      boundary head west (m)             / 100 /
    HBE      boundary head east (m)             / 50  /
    HBN      boundary head north (m)            / 100 /
    HBS      boundary head south (m)            / 100 /
    H0       initial head  (m)                  / 100 /
    QMAX     max pumping rate (m3 per d per m2) / 40 /
    QMIN     min pumping rate (m3 per d per m2) /   0 /
         ;

PARAMETER
    DEM(L)   water demand  (1000 m3 per day);
    DEM(L) = 100*ord(L);

TABLE T(I,J) isotropic transmissivity (m2 per day)

     J0  J1  J2  J3  J4  J5  J6  J7  J8  J9 J10
I0  205 185 190 175 160 185 195 190 175 160 165
I1  205 185 190 175 160 185 195 190 175 160 165
I2  260 235 215 200 175 185 195 190 175 160 165
I3  255 250 230 205 150 185 195 190 175 160 165
I4  210 225 220 195 145 185 195 190 175 160 165
I5  185 195 190 175 160 185 195 190 175 160 165
I6  160 205 215 200 165 185 195 190 175 160 165
I7  155 200 210 215 150 185 195 190 175 160 165
I8  170 215 200 195 155 185 195 190 175 160 165
I9  170 215 200 195 155 185 195 190 175 160 165
I10 170 215 200 195 155 185 195 190 175 160 165;

* finite-difference cell types
TABLE CELL(I,J) cell type

* -2 = north no-flow bndy
* -1 = south no-flow bndy
*  0 = interior cells
*  1 = west fixed potential
*  2 = east fixed potential
*  3 = north fixed potential
*  4 = south fixed potential

    J0  J1  J2  J3  J4  J5  J6  J7  J8  J9 J10
I0   1  -2  -2  -2  -2  -2  -2  -2  -2  -2   2
I1   1   0   0   0   0   0   0   0   0   0   2
I2   1   0   0   0   0   0   0   0   0   0   2
I3   1   0   0   0   0   0   0   0   0   0   2
I4   1   0   0   0   0   0   0   0   0   0   2
I5   1   0   0   0   0   0   0   0   0   0   2
I6   1   0   0   0   0   0   0   0   0   0   2
I7   1   0   0   0   0   0   0   0   0   0   2
```

```
I8   1   0   0   0   0   0   0   0   0   0   2
I9   1   0   0   0   0   0   0   0   0   0   2
I10  1  -1  -1  -1  -1  -1  -1  -1  -1  -1   2;

* Use for fixed head all around
*I0   1   3   3   3   3   3   3   3   3   3   2
*I10  1   4   4   4   4   4   4   4   4   4   2;
* Use for fixed head east and west
*I0   1  -2  -2  -2  -2  -2  -2  -2  -2  -2   2
*I10  1  -1  -1  -1  -1  -1  -1  -1  -1  -1   2;

* finite-difference coefficients
PARAMETER
        A(I,J)   finite-difference coefficient
        B(I,J)   finite-difference coefficient
        C(I,J)   finite-difference coefficient
        D(I,J)   finite-difference coefficient
         ;

        A(I,J) = 0.0;
        B(I,J) = 0.0;
        C(I,J) = 0.0;
        D(I,J) = 0.0;
A(I,J) =  (2*T(I+1,J)*T(I,J)/(T(I+1,J)+T(I,J))/(DX*DX))$(ord(I) ne card(I))
             +(   T(I,J)/(DX*DX))$(ord(I) eq card(I));
B(I,J) = (-2*T(I-1,J)*T(I,J)/(T(I-1,J)+T(I,J))/(DX*DX))$(ord(I) ne 0)
             +(   T(I,J)/(DX*DX))$(ord(I) eq 0) ;
C(I,J) =  (2*T(I,J+1)*T(I,J)/(T(I,J+1)+T(I,J))/(DY*DY))$(ord(J) ne card(J))
             +(   T(I,J)/(DY*DY))$(ord(J) eq card(J));
D(I,J) = (-2*T(I,J-1)*T(I,J)/(T(I,J-1)+T(I,J))/(DY*DY))$(ord(J) ne 0)
             +(   T(I,J)/(DY*DY))$(ord(J) eq 0);

SCALAR
        F        finite-difference coefficient;
        F = S/DT ;


VARIABLES
        OBJ     Objective value;

POSITIVE VARIABLES
        Q(I,J,L)  pumping  in  cell  ij  in  period  L  [1000  m3  per
day][decision]
        H(I,J,L) head in cell ij in period L
        ;

*        initial conditions
         H.fx(I,J,'L0')$(CELL(I,J) le 0) = H0 ;

*        boundary conditions
* Use for fixed head east and west
         H.fx(I,J,L)$(CELL(I,J) = 1) = HBW ;
         H.fx(I,J,L)$(CELL(I,J) = 2) = HBE ;
* Use for fixed head all around
         H.fx(I,J,L)$(CELL(I,J) = 3) = HBN ;
         H.fx(I,J,L)$(CELL(I,J) = 4) = HBS ;

* maximum/minimum pumping contrtraints
         Q.lo(I,J,L)$(CELL(I,J) le 0) = QMIN;
         Q.up(I,J,L)$(CELL(I,J) le 0) = QMAX;
         Q.lo(I,J,L)$(CELL(I,J) gt 0) = QMIN;
         Q.up(I,J,L)$(CELL(I,J) gt 0) = QMIN;
```

```
EQUATIONS
        CONT(I,J,L)    finite-difference continuity eqn. for period L
        DEMAND(L)      demand constraint eqn. for period L
        OBJECTIVE      objective function;

* continuity equation (including no-flow boundaries)
CONT(I,J,L) ..
     ( A(I,J)*(H(I+1,J,L+1)-H(I,J,L+1))
      +B(I,J)*(H(I,J,L+1)-H(I-1,J,L+1))
      +C(I,J)*(H(I,J+1,L+1)-H(I,J,L+1))
      +D(I,J)*(H(I,J,L+1)-H(I,J-1,L+1))
      -F*     (H(I,J,L+1))             )$(CELL(I,J) eq 0)
    +( A(I,J)*(H(I+1,J,L+1)-H(I,J,L+1))
      +C(I,J)*(H(I,J+1,L+1)-H(I,J,L+1))
      +D(I,J)*(H(I,J,L+1)-H(I,J-1,L+1))
      -F*     (H(I,J,L+1))             )$(CELL(I,J) eq -2)
    +( B(I,J)*(H(I,J,L+1)-H(I-1,J,L+1))
      +C(I,J)*(H(I,J+1,L+1)-H(I,J,L+1))
      +D(I,J)*(H(I,J,L+1)-H(I,J-1,L+1))
      -F     *(H(I,J,L+1))             )$(CELL(I,J) eq -1)
      -Q(I,J,L+1)
    =E= -F*    H(I,J,L);

* demand constraint
DEMAND(L) ..  SUM((I,J), Q(I,J,L)) =G= DEM(L);

* objective function (sum of pumping or cost of pumping)
OBJECTIVE ..  OBJ =E= (SUM((I,J,L), Q(I,J,L))) ;

MODEL PUMPING /ALL/ ;
SOLVE PUMPING USING LP MINIMIZING OBJ ;

File GW2DT /GW2DT_output.txt/
Put GW2DT;
PUT ' Objective = '; Put OBJ.l; Put /;
Put 'Pumping'/; Put '         ';
Loop(L,PUT ' Time = '; Put L.tl:6; PUT ' Demand = '; Put DEM(L):6; Put /;
PUT ' '; Loop(J, Put J.tl:6;) Put /;
Loop(I, Put I.tl:6; Loop(J, Put Q.L(I, J,L):6:0;); Put /;););
Put 'Head'/; Put '         ';
Loop(J, Put J.tl:6;) Put /;
Loop(L, PUT ' Time = '; Put L.tl:6; Put /;
PUT ' '; Loop(J, Put J.tl:6;) Put /;
Loop(I, Put I.tl:6; Loop(J, Put H.L(I, J, L):6:0;); Put /;););
```

The results of running this model are:

```
Objective =      9100.00
Pumping
         Time = L0     Demand = 100.00
 J0    J1    J2    J3    J4    J5    J6    J7    J8    J9    J10
I0         0    40    40    20     0     0     0     0     0     0
I1         0     0     0     0     0     0     0     0     0     0
I2         0     0     0     0     0     0     0     0     0     0
I3         0     0     0     0     0     0     0     0     0     0
I4         0     0     0     0     0     0     0     0     0     0
I5         0     0     0     0     0     0     0     0     0     0
I6         0     0     0     0     0     0     0     0     0     0
I7         0     0     0     0     0     0     0     0     0     0
I8         0     0     0     0     0     0     0     0     0     0
```

I9	0	0	0	0	0	0	0	0	0	0	0
I10	0	0	0	0	0	0	0	0	0	0	0

Time = L1 Demand = 200.00

	J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
I0	0	0	0	0	40	40	0	0	0	0	0
I1	0	0	0	0	0	0	0	40	0	0	0
I2	0	0	0	0	0	0	0	0	0	0	0
I3	0	0	0	0	40	0	0	0	0	0	0
I4	0	0	0	0	0	0	0	0	0	0	0
I5	0	0	0	0	0	0	0	0	0	0	0
I6	0	0	0	0	0	0	0	0	0	0	0
I7	0	0	0	0	0	0	0	0	0	0	0
I8	0	0	0	0	0	0	0	0	0	0	0
I9	0	0	0	0	0	0	0	0	0	0	0
I10	0	0	0	40	0	0	0	0	0	0	0

Time = L2 Demand = 300.00

	J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
I0	0	0	0	0	0	0	0	0	0	0	0
I1	0	0	0	0	0	0	0	0	0	0	0
I2	0	0	0	0	0	0	0	0	0	0	0
I3	0	0	0	0	0	0	0	0	0	0	0
I4	0	0	0	0	0	0	0	0	0	0	0
I5	0	0	0	0	0	20	0	0	0	0	0
I6	0	0	0	0	0	40	0	0	0	0	0
I7	0	0	0	40	0	0	0	40	0	0	0
I8	0	0	0	0	0	40	0	0	0	0	0
I9	0	0	0	0	0	0	0	0	0	0	0
I10	0	0	0	0	40	40	0	40	0	0	0

Time = L3 Demand = 400.00

	J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
I0	0	0	0	40	0	40	0	0	0	0	0
I1	0	0	0	0	0	0	0	0	0	0	0
I2	0	0	0	0	0	40	0	0	0	0	0
I3	0	0	0	0	0	40	0	0	0	0	0
I4	0	0	0	0	40	0	0	40	0	0	0
I5	0	0	0	0	0	40	0	0	0	0	0
I6	0	0	0	0	0	40	0	0	0	0	0
I7	0	0	0	40	0	40	0	0	0	0	0
I8	0	0	0	0	0	0	0	0	0	0	0
I9	0	0	0	0	0	0	0	0	0	0	0
I10	0	0	0	0	0	0	0	0	0	0	0

Time = L12 Demand = 1.3E+3

	J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
I0	0	0	0	40	15	0	0	28	0	0	0
I1	0	0	40	0	21	0	0	27	0	0	0
I2	0	0	40	40	15	0	0	25	0	0	0
I3	0	0	40	40	15	0	0	21	0	25	0
I4	0	0	40	0	23	0	0	17	0	40	0
I5	0	0	40	0	24	0	0	16	0	40	0
I6	0	0	40	0	24	0	0	15	0	40	0
I7	0	0	40	40	16	0	0	15	0	40	0
I8	0	0	40	0	23	0	0	14	0	40	0
I9	0	0	40	0	26	0	0	14	0	40	0
I10	0	0	40	0	27	0	0	14	0	40	0

Head

	J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10

Time = L0

	J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
I0	100	100	100	100	100	100	100	100	100	100	50
I1	100	100	100	100	100	100	100	100	100	100	50
I2	100	100	100	100	100	100	100	100	100	100	50
I3	100	100	100	100	100	100	100	100	100	100	50
I4	100	100	100	100	100	100	100	100	100	100	50
I5	100	100	100	100	100	100	100	100	100	100	50
I6	100	100	100	100	100	100	100	100	100	100	50
I7	100	100	100	100	100	100	100	100	100	100	50
I8	100	100	100	100	100	100	100	100	100	100	50

```
I9       100   100   100   100   100   100   100   100   100   100     50
I10      100   100   100   100   100   100   100   100   100   100     50
 Time = L1
 J0    J1    J2    J3    J4    J5    J6    J7    J8    J9    J10
I0       100    88    76    61    40    34    38    39    43    47     50
I1       100    89    77    65    51    44    42    37    43    47     50
I2       100    90    79    68    56    50    47    45    46    48     50
I3       100    91    81    70    55    54    52    50    49    50     50
I4       100    91    82    73    64    59    55    53    52    51     50
I5       100    91    83    76    68    62    59    56    53    52     50
I6       100    91    84    77    70    65    61    57    55    52     50
I7       100    91    84    78    71    66    62    58    56    53     50
I8       100    91    84    77    71    66    62    59    56    53     50
I9       100    91    82    74    69    65    62    59    56    53     50
I10      100    90    80    67    67    65    62    59    56    53     50
 Time = L2
 J0    J1    J2    J3    J4    J5    J6    J7    J8    J9    J10
I0       100    93    86    79    72    66    62    58    55    52     50
I1       100    93    86    79    72    65    61    57    55    52     50
I2       100    93    85    77    70    64    59    56    53    52     50
I3       100    92    84    76    67    60    56    53    52    51     50
I4       100    90    81    72    63    54    51    50    49    49     50
I5       100    89    78    67    56    45    45    45    46    48     50
I6       100    86    74    61    50    37    38    39    42    46     50
I7       100    85    70    54    45    35    33    30    38    44     50
I8       100    84    70    55    41    28    30    31    36    43     50
I9       100    84    69    54    38    27    26    28    34    42     50
I10      100    83    68    51    27    18    21    20    31    41     50
 Time = L3
 J0    J1    J2    J3    J4    J5    J6    J7    J8    J9    J10
I0       100    83    65    44    36    24    30    34    39    44     50
I1       100    84    68    52    39    30    31    34    39    44     50
I2       100    85    70    55    40    25    30    33    38    44     50
I3       100    86    71    56    39    25    29    32    37    44     50
I4       100    85    71    55    34    28    29    27    36    44     50
I5       100    85    71    56    40    26    30    33    39    44     50
I6       100    85    71    56    43    28    33    37    41    45     50
I7       100    85    71    54    46    33    38    40    43    47     50
I8       100    86    74    62    53    45    44    44    45    48     50
I9       100    88    77    66    58    51    48    47    47    48     50
I10      100    88    78    68    60    53    50    48    48    49     50
Time = L12
 J0    J1    J2    J3    J4    J5    J6    J7    J8    J9    J10
I0       100    68    36     9     0     0     0     0    15    32     50
I1       100    67    31    12     0     0     0     0    15    31     50
I2       100    66    30     8     0     0     0     0    14    30     50
I3       100    66    30     9     0     0     0     0    11    23     50
I4       100    65    30    14     0     0     0     0     9    19     50
I5       100    64    29    14     0     0     0     0     9    17     50
I6       100    62    28    13     0     0     0     0     8    17     50
I7       100    61    27     9     0     0     0     0     8    17     50
I8       100    62    28    13     0     0     0     0     8    17     50
I9       100    63    29    15     0     0     0     0     8    17     50
I10      100    63    30    16     0     0     0     0     8    17     50
```

4.7 Exercises

1. (Adapted from Mays and Tung, 1992, prob. 8.4.3). Consider an undeveloped, homogeneous, isotropic, confined aquifer of large areal extent in which there are L potential production wells located at points $[(x_\ell, y_\ell), \ell = 1, \ldots, L]$ and K points at which water levels are to be observed $[(x_k, y_k), k = 1, \ldots, K]$. For steady-state flow in the aquifer, the Theim equation can be used to

predict the drawdown s_k at the control points due to the pumping Q_ℓ at the production wells

$$s_k = \sum_{\ell=1}^{L} s_{k\ell} = \sum_{\ell=1}^{L} a_{k\ell} Q_\ell \qquad k = 1, \cdots, K$$

where $s_{k\ell}$ is the drawdown at point k due to pumping at well ℓ alone, and

$$a_{k\ell} = \frac{1}{2\pi T} \ln\left(\frac{R_\ell}{r_{k\ell}}\right) \quad k = 1, \cdots, K; \quad \ell = 1, \cdots, L$$

T is the transmissivity of the aquifer, R_ℓ is the radius of influence of the ℓ-th well, and

$$r_{k\ell} = \sqrt{(x_k - x_\ell)^2 + (y_k - y_\ell)^2}$$

A management model with the objective of maximizing the total withdrawal from the aquifer while meeting specified upper bound constraints on the drawdown at the control points and upper and lower bounds on the pumping rates is

Maximize $\sum_{\ell=1}^{L} Q_\ell$

Subject to

$$\sum_{\ell=1}^{L} a_{k\ell} Q_\ell = s_k \le s_k^* \qquad k = 1, \cdots, K$$

$$Q_\ell^{low} \le Q_\ell \le Q_\ell^{up} \qquad \ell = 1, \cdots, L$$

Consider the situation (shown in the Figure) of three production wells (L=3) and five control points (K=5) a transmissivity of 5,000 gal/day/ft^2, and a radius of influence for each well of 700 ft. Additional data related to the problem are listed in the Table P4.1.1 and the response coefficients $a_{k\ell}$ are listed in the Table P4.1.2.

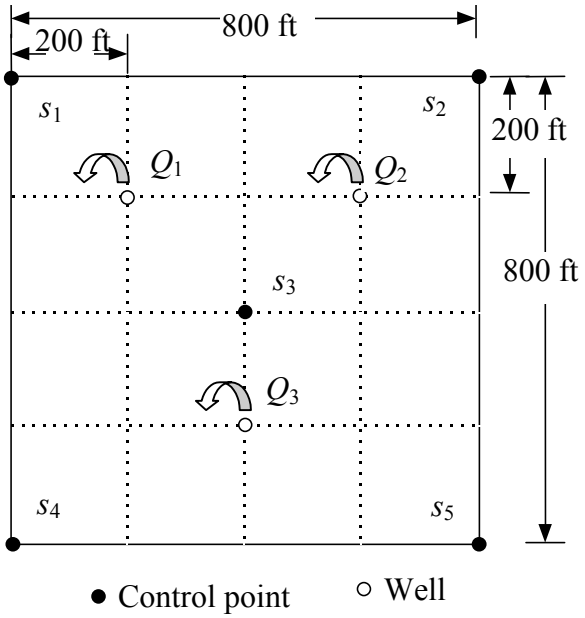


Figure P4.1.1. Aquifer for steady-state pumping optimization example.

Table P4.1.1. Distance (ft.) between well ℓ and control point k.

$r_{k,\ell}$ Well		$k=1$	Control 2	Point 3	4	5	Production Capacity, Q_ℓ^{up} (gal/day)
$\ell=$	1	282.8	632.5	282.8	848.5	632.5	200,000
	2	632.5	282.8	282.8	632.5	848.5	200,000
	3	721.1	721.1	200	565.7	565.7	200,000
Max. Allowable Drawdown, s_k* (ft)		7	7	15	7	7	

Table P4.1.2. Response matrix $\boldsymbol{A}$ ($\times 10^{-5}$) for 5 control points and 3 production wells.

$a_{k,\ell}$ Control Pt.		Well $\ell=1$	2	3
$k=$	1	2.885×10^{-5}	3.228×10^{-6}	0
	2	3.228×10^{-6}	2.885×10^{-5}	0
	3	2.885×10^{-5}	2.885×10^{-5}	3.988×10^{-5}
	4	0	3.228×10^{-6}	6.781×10^{-6}
	5	3.228×10^{-6}	0	6.781×10^{-6}

Formulate and solve this model in GAMS to find the optimal puming rates for each well.

2. One-dimensional problem

3. Two-dimensional, steady-state problem

4. Two-dimensional, transient problem

5. OPTIMAL SELECTION OF AGRICULTURAL CROPS

(Alex Meuraus)

THIS SECTION STILL NEEDS EDITING

Task:

The task of agricultural crop selection is a difficult one that can be Modeled in GAMS. Assume that the following is known:

1. Types of crops available for planting;
2. Modeling period (one year, using a time step of one month)
3. Irrigation water demands for each crop for each month
4. Stress coefficient for each crop for each month indicating the yield decrease of a given crop for a deficit of irrigation water in each time interval;
5. Price for each crop;
6. Yield for each crop;
7. Proposed area for planting crops; and
8. Available irrigation water hydrograph.

The model is to determine:

1. The amount of land to be allocated to each crop; and
2. The irrigation water required for each crop.

Objective: Maximize proceeds from the yield sales for each crop

The mathematical model is represented by the equations:

$$R = \sum_{p} C(p) * s(p) * Y(p) * \prod_{t} \left[\frac{Wr(p,t)}{Wd(p,t)} \right]^{l(p,t)}$$

$$\sum_{p} S(p) < So$$

$$\sum_{p} Wr(p,t) < Wo(t)$$

where

p	crop type
t	time index (month)
S(p)	area for crop p
$Wr(p,t)$	delivery of water for each crop in each time period;
$Wd(p,t)$	demand for water for each crop in each time period;
$l(p,t)$	stress coefficient for each crop in each time period;
So	total area of land available;
$Wo(t)$	hydrograph of available water;
$C(p)$	price for crop p;
$S(p)$	area planted with crop p; and
$Y(p)$	yield of crop p

The GAMS code for the model is:

```
SETS p  crops  / cotton, rice, wheat, maize, others /
     t  month  / jan,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,dec /

TABLE wd(p,t) water demand (mill m3 per ha)
          jan  feb  mar  apr  may   jun   jul  aug  sep  oct  nov  dec
cotton                          1.2   1.2   1.2  1.2
rice                            2.9   2.8   2.9  2.8  2.6
wheat                           0.8   0.8   0.9  0.8  0.7
maize                           1.8   1.9   1.9  1.8  1.8
others                          1.1   1.1   1.1  1.1  1.1

TABLE l(p,t) stress coefficients
          jan  feb  mar  apr   may   jun   jul  aug  sep  oct  nov  dec
cotton                          1.0   1.0   1.0  0.9
rice                            0.9   0.8   0.9  0.8  0.6
wheat                           0.8   0.8   0.9  0.8  0.7
maize                           0.8   0.9   0.9  0.8  0.8
others                          1.0   1.0   1.0  1.0  1.0

PARAMETERS
   C(p)  (price $ per c)     / cotton 3, rice 5, wheat 7, maize
11,others 60/
   Y(p)  (yield c per ha)    / cotton 2, rice 3, wheat 4, maize  5,
others 3/
   wo(t) (water available mill m3 per month);
*          computed as a uniform number between numbers 900 - 1100.
   wo(t) = UNIFORM(900,1100);

SCALAR So land available / 1000 /;

VARIABLES  obj  revenue  ($);

POSITIVE VARIABLES
   S(p) cultivated land (ha)
   wr(p,t) water delivered (mill m3 per ha);

* Upper bound on Delivered water
wr.UP(p,t) = wd(p,t);

* Initial guesses for cultivated land and delivered water
S.L(p) = So/CARD(p);
```

```
wr.L(p,t)  = wo(t);

EQUATION   ben      Objective function
           land     land balance (ha)
           water(t) water balance (ha);

ben.. obj =E=
sum(p,C(p)*S(p)*Y(p)*prod(t$wd(p,t),(wr(p,t)/wd(p,t))**l(p,t)));

* Limit on cultivated land
land.. sum(p, S(p)) =L= So;

* Limit on irrigation water
water(t).. sum(p, S(p)*wr(p,t)) =L= wo(t);

MODEL crop / ALL /;

SOLVE crop USING NLP MAXIMiZING obj;

FILE res /Crop1.txt/;
PUT res
PUT "Revenue = ", obj.L//;
PUT "Crop                Land"/;
LOOP(p,PUT p.TL,  s.l(p)/;);
```

The results of running the model are:

```
Revenue =    149626.12

Crop               Land
cotton            33.40
rice               0.00
wheat              0.00
maize              0.00
others           830.38
```

6. OPTIMAL DEVELOPMENT OF CANALS

THIS SECTION STILL NEEDS EDITING

```
*Grid points
set x /x1*x20/;
set y /y1*y20/;

SET INSIDE(X,Y);
INSIDE(X,Y)=YES$((ord(x)<card(x))and(ord(y)<card(y)));

* Topography of grid
TABLE zp(x,y)
       y1     y2     y3    y4    y5     y6      y7     y8   y9    y10   y11  y12  y13  y14  y15
y16  y17  y18  y19  y20
x1    1.1    1.1    1.1   1.1  -99.  -99.1   -99.1  -97.1  -89.  -99.  -90.  1.1  1.1  1.1  1.1
1.1  1.1  1.1  1.1  1.1
x2    1.2    1.2    1.2   1.2  -99.  -98.2   -99.2  -98.2  -69.  -89.  -90.  1.2  1.2  1.2  1.2
1.2  1.2  1.2  1.2  1.2
x3    1.3    1.3    1.3   1.3  -91.  -97.3   -99.3  -97.3  -49.  -79.  -8.3  8.3  1.3  8.3  1.3
1.3  1.3  1.3  1.3  1.3
x4    1.4    1.4    1.4   1.4  -91.  -67.4   -99.4  -67.4  -35.  -68.  -8.4  8.4  1.4  8.4  1.4
1.4  1.4  1.4  1.4  1.4
x5    1.5    1.5    8.5   1.5  -91.  -67.5   -99.5  -57.5  -25.  -57.  -8.5  8.5  1.5  8.5  1.5
```

```
1.5  1.5  1.5  1.5  1.5
x6    1.6    1.6    8.6   2.6   1.6  -37.6   -79.6  -47.6  -15.  -46.  -8.6  8.6  8.6  8.6  1.6
1.6  1.6  1.6  1.6  1.6
x7    -3     1.7    8.7   8.7   8.7  -23.7   -49.7  -33.7  -15.  -35.  -8.7  8.7  8.7  8.7  1.7
1.7  1.7  1.7  1.7  1.7
x8    -10    -9     8.8   8.8   8.8  -16.8   -22.8  -12.8  -10.  -14.  -0.8  1.8  0.8  8.8  8.8
1.8  1.8  1.8  1.8  1.8
x9    -39    -39    8.9   8.9   8.9  -10.9    -5.9   -7.9   -9.  -5.9  0.9  0.9  0.9   8.9  8.9
1.9  1.9  1.9  1.9  1.9
x10   -99    -79   -8.8   8.8   8.8  -10.8    -8.8   -8.8   -8.  -8.8  0.8  0.8  0.8   0.8  8.8
1.8  1.8  1.8  1.8  1.8
x11   -39    -39    1.7   8.7   8.7   -8.7    -8.7   -8.7   -8.  -8.7  0.7  0.7  0.7   0.7  8.7
1.7  1.7  1.7  1.7  1.7
x12   -9     9.6    1.6   1.6   8.6   -8.6     8.6    8.6   1.6  -9.6  0.6  0.6  0.6   0.6  8.6
3.6  1.6  1.6  1.6  1.6
x13   -3     9.5    1.5   1.5   8.5   8.5      8.5    8.5   1.5  -9.5  0.5  0.5  1.5   0.5  8.5
3.5  1.5  1.5  1.5  1.5
x14   1.4    1.4    1.4   1.4   8.4   1.4      1.4    8.4   1.4  -9.4  0.4  0.4  1.4   0.4  8.4
3.4  1.4  1.4  1.4  1.4
x15   1.3    1.3    1.3   1.3   8.3   1.3      1.3   -8.3   1.3  -9.3  -1.3  0.3  1.3  1.3  8.3
1.3  1.3  1.3  1.3  1.3
x16   1.2    1.2    1.2   1.2   8.2   8.2      8.2   -8.2   1.2  -9.2  -2.2  1.2  1.2  1.2  8.2
1.2  1.2  1.2  1.2  1.2
x17   1.1    1.1    1.1   1.1   8.1   8.1      8.1   -8.1  -1.1  -9.1  -3.1  1.1  1.1  1.1  1.1
1.1  1.1  1.1  1.1  1.1
x18   1.2    1.2    1.2   1.2   1.2   1.2      1.2   -6.2  -9.2  -9.2  -4.2  1.2  1.2  1.2  1.2
1.2  1.2  1.2  1.2  1.2
x19   1.3    1.3    1.3   1.3   1.3   1.3      1.3   -7.3  -9.3  -9.3  -5.3  1.3  1.3  1.3  1.3
1.3  1.3  1.3  1.3  1.3
x20   1.4    1.4    1.4   1.4   1.4   1.4      1.4   -8.4  -9.4  -9.4  -6.4  1.4  1.4  1.4  1.4
1.4  1.4  1.4  1.4  1.4
;

PARAMETER START_END(X,Y);
          START_END(X,Y)=0;
          START_END('X4','Y3')= 1;
          START_END('X13','Y18')=-1;

VARIABLE Dx(x,y), Dy(x,y), obj;

EQUATION first(x,y), ben;

first(x,y)$(INSIDE(x,y)).. START_END(X,Y) =E= (dx(x+1,y)-dx(x,y))
                                             + (dy(x,y+1)-dy(x,y));

ben.. obj =e=  1000* sum((x,y)$(INSIDE(x,y)),
                        (30-zp(x,y)) * (sqrt(dy(x,y)*dy(x,y)+0.00001))
                      + (30-zp(x,y)) * (sqrt(dx(x,y)*dx(x,y)+0.00001)));

dx.up(x,y)=1;
dy.up(x,y)=1;

dx.lo(x,y)=-1;
dy.lo(x,y)=-1;

dx.fx('x1',y)=0;   dy.fx('x1',y)=0;
dx.fx(x,'y1')=0;   dy.fx(x,'y1')=0;
dx.fx('x20',y)=0;  dy.fx('x20',y)=0;
dx.fx(x,'y20')=0;  dy.fx(x,'y20')=0;

MODEL Canal /all/;

SOLVE Canal using   NLP MINIMIZING obj;

FILE res /Canal.txt/
PUT res;
PUT " Objective function= "; PUT obj.l;
PUT /;

PARAMETER no_yes(x,y);
no_yes(x,y)=
              -START_END(X,Y)+
               SQRT(dx.l(x+1,y)*dx.l(x+1,y)+0.000001 )+
               SQRT(dx.l(x,y)  *dx.l(x,y)  +0.000001 )+
               SQRT(dy.l(x,y+1)*dy.l(x,y+1)+0.000001 )+
```

```
               SQRT(dy.l(x,y)  *dy.l(x,y)  +0.000001 )
                ;

PUT "Canal yes_no"; PUT /;
LOOP(X,
LOOP(Y, PUT no_yes(X,Y):5:1; ); PUT /;
    );

PUT "level of earth";PUT /;
LOOP(X,
LOOP(Y, PUT zp(X,Y):7:2; ); PUT /;
    );
PUT /;
```

The results of the moel are:

```
Objective function=    664152.36
Canal yes_no
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   2.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   2.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   2.0   0.1   0.1   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   1.9   0.2   0.1   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   1.8   1.8   1.8   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.1   1.9   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.1   1.9   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   1.9   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   1.9   1.9   1.9   1.9   1.9   1.9   1.9   1.9   1.9   1.9   1.9   1.9   1.9   2.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
level of earth
    1.10    1.10    1.10    1.10 -99.00 -99.10 -99.10 -97.10 -89.00 -99.00 -90.00    1.10    1.10    1.10    1.10    1.10    1.10    1.10    1.10    1.10
    1.20    1.20    1.20    1.20 -99.00 -98.20 -99.20 -98.20 -69.00 -89.00 -90.00    1.20    1.20    1.20    1.20    1.20    1.20    1.20    1.20    1.20
    1.30    1.30    1.30    1.30 -91.00 -97.30 -99.30 -97.30 -49.00 -79.00  -8.30    8.30    1.30    8.30    1.30    1.30    1.30    1.30    1.30    1.30
    1.40    1.40    1.40    1.40 -91.00 -67.40 -99.40 -67.40 -35.00 -68.00  -8.40    8.40    1.40    8.40    1.40    1.40    1.40    1.40    1.40    1.40
    1.50    1.50    8.50    1.50 -91.00 -67.50 -99.50 -57.50 -25.00 -57.00  -8.50    8.50    1.50    8.50    1.50    1.50    1.50    1.50    1.50    1.50
    1.60    1.60    8.60    2.60    1.60 -37.60 -79.60 -47.60 -15.00 -46.00  -8.60    8.60    8.60    8.60    1.60    1.60    1.60    1.60    1.60    1.60
   -3.00    1.70    8.70    8.70    8.70 -23.70 -49.70 -33.70 -15.00 -35.00  -8.70    8.70    8.70    8.70    1.70    1.70    1.70    1.70    1.70    1.70
  -10.00   -9.00    8.80    8.80    8.80 -16.80 -22.80 -12.80 -10.00 -14.00  -0.80    1.80    0.80    8.80    8.80    1.80    1.80    1.80    1.80    1.80
  -39.00  -39.00    8.90    8.90    8.90 -10.90   -5.90   -7.90   -9.00   -5.90    0.90    0.90    0.90    8.90    8.90    1.90    1.90    1.90    1.90    1.90
  -99.00  -79.00   -8.80    8.80    8.80 -10.80   -8.80   -8.80   -8.00   -8.80    0.80    0.80    0.80    0.80    8.80    1.80    1.80    1.80    1.80    1.80
  -39.00  -39.00    1.70    8.70    8.70   -8.70   -8.70   -8.70   -8.00   -8.70    0.70    0.70    0.70    0.70    8.70    1.70    1.70    1.70    1.70    1.70
   -9.00    9.60    1.60    1.60    8.60   -8.60    8.60    8.60    1.60   -9.60    0.60    0.60    0.60    0.60    8.60    3.60    1.60    1.60    1.60    1.60
   -3.00    9.50    1.50    1.50    8.50    8.50    8.50    8.50    1.50   -9.50    0.50    0.50    1.50    0.50    8.50    3.50    1.50    1.50    1.50    1.50
    1.40    1.40    1.40    1.40    8.40    1.40    1.40    8.40    1.40   -9.40    0.40    0.40    1.40    0.40    8.40    3.40    1.40    1.40    1.40    1.40
    1.30    1.30    1.30    1.30    8.30    1.30    1.30   -8.30    1.30   -9.30   -1.30    0.30    1.30    1.30    8.30    1.30    1.30    1.30    1.30    1.30
    1.20    1.20    1.20    1.20    8.20    8.20    8.20   -8.20    1.20   -9.20   -2.20    1.20    1.20    1.20    8.20    1.20    1.20    1.20    1.20    1.20
    1.10    1.10    1.10    1.10    8.10    8.10    8.10   -8.10   -1.10   -9.10   -3.10    1.10    1.10    1.10    1.10    1.10    1.10    1.10    1.10    1.10
    1.20    1.20    1.20    1.20    1.20    1.20    1.20   -6.20   -9.20   -9.20   -4.20    1.20    1.20    1.20    1.20    1.20    1.20    1.20    1.20    1.20
    1.30    1.30    1.30    1.30    1.30    1.30    1.30   -7.30   -9.30   -9.30   -5.30    1.30    1.30    1.30    1.30    1.30    1.30    1.30    1.30    1.30
    1.40    1.40    1.40    1.40    1.40    1.40    1.40   -8.40   -9.40   -9.40   -6.40    1.40    1.40    1.40    1.40    1.40    1.40    1.40    1.40    1.40
```

7. PROBLEMS OF POWER NETWORKS

7.1 Problem of Power Generation, Distribution, and Consumption

Consider a power network comprised of two groups: the first group with power generation facilities and the second group for power consumption. The power transmission lines can be used to transfer power to consumers. Figure 7.1.1 shows a network with power generation and power consumption nodes.

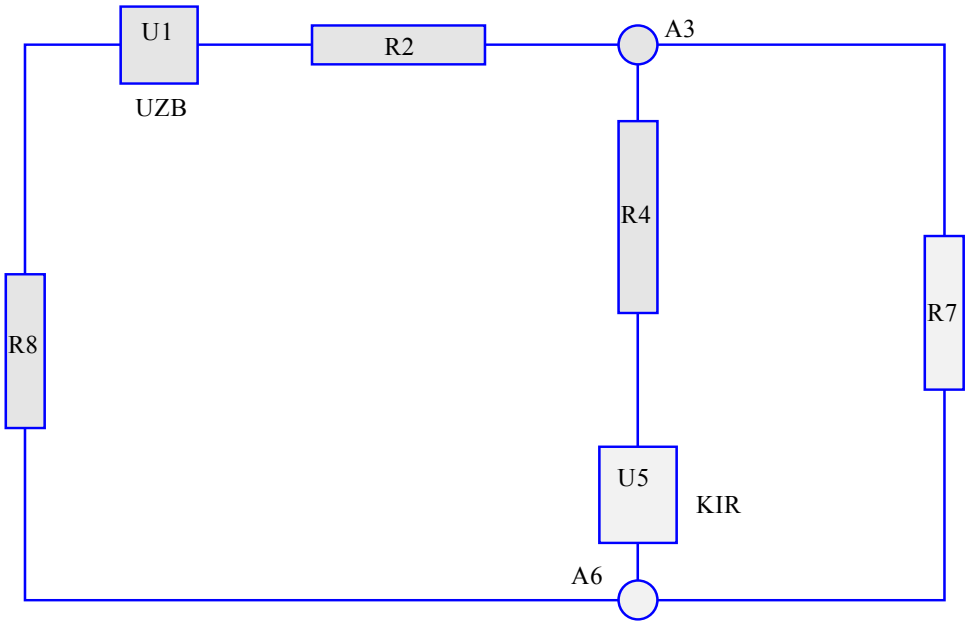


Fig. 7.1.1. Energy and Electricity Tasks Schematic

The power generation at node U5 (a5) is 100.
The power consumption at node R4 (a4) is 80.
The power consumption at node R8 (a8) is 30.
The power consumption at node R2 (a2) is 10.
The power transferred from node R8 (a8) to node a6 is 100.
The power transferred from node R2 (a2) to node a3 is 50.

The power transfer in all lines of the network and the power generation at the node U1 (a1) are to be determined. It is known that the power generation at this node is less than 900.

This problem is similar to the problem connected with power generation and power transfer between the nodes countries in the Central Asian network. The essence of the problem becomes clear assuming that "a1" refers to the set UZB (Uzbekistan), and "a5" to another set KIR (Kyrgyzstan).

Besides, there is a condition on specified power transfer through the transmission lines of the network.

Consider the laws characteristic for the model:

1) The net flow through a regular junction is 0:

$$\sum_{m\ incoming} i(m,n) = \sum_{m\ outgoing} i(n,m)$$

2) Flow from the first node in the direction of a second node of the circuit does not change. However, it has a "-" symbol for the first node, and "+" for the second node:

$$i(m,n) = -i(n,m)$$

3) Flow transferred through a power generating node is increased by the value of the generated power:

$$\sum_{m\ incoming} i(m,n) - \sum_{m\ outgoing} i(n,m) = e(n)$$

4) Flow transferred through a power consumption node is decreased by the value of the consumed power:

$$\sum_{m\ incoming} i(m,n) - \sum_{m\ outgoing} i(n,m) = -r(n)$$

In the equations:

i(m,n)	power flow from node *m* to node *n*,
e(n)	power generation at node *n*;
r(n)	power consumption at node *n*;

On the basis of these equations we can show that the total power generation is equal to the total power consumption.

```
SETS
m  /a1,a2,a3,a4,a5,a6,a7,a8   /

alias(m,m1);

SETS
mr(m)  consumers        /a2,a8,a4,a7/
me(m)  producers        /a1,A5/
m0(m)  junctions        /a3,a6/

m_and_m1(m,m1) all nodes and their connections /
a1.a2, a1.a8,
a2.a1, a2.a3,
```

```
a3.a2, a3.a4, a3.a7
a4.a3, a4.a5,
a5.a4, a5.a6,
a6.a5, a6.a8, a6.a7,
a7.a3, a7.a6,
a8.a1, a8.a6        /

VARIABLES
e(m),     power generation
r(m),     power consumption
i(m,m1),  power flow
obj;

equation
coni(m,m1),
conu0(m),
conue(m),
conur(m),
ben;

* All nodes
coni(m,m1)..          i(m,m1) =e= -i(m1,m)  ;
* Junction nodes
conu0(m)$(m0(m))..    sum(m1$(m_and_m1(m,m1)),i(m,m1)) =e=   0     ;
* Generation nodes
conue(m)$(me(m))..    sum(m1$(m_and_m1(m,m1)),i(m,m1)) =e=   e(m)  ;
* COnsumption nodes
conur(m)$(mr(m))..    sum(m1$(m_and_m1(m,m1)),i(m,m1))  =e=  -r(m)  ;

ben..           obj =E= 1;

e.up('a1')=         900;
e.lo('a1')=           0;

e.fx('a5')=         100;

i.fx('a8','a6')=100;
i.fx('a2','a3')= 50;

r.fx('a4')= 80;
r.fx('a8')= 30;
r.fx('a2')= 10;

MODEL VAN /ALL/;
SOLVE VAN USING NLP MinimaZING obj;

file res /odc1.txt/
put res;
put " I  Power Flow " /;
put "             ";
loop(m, put m.tl:10; );
        put /;
loop(m, put m.tl:10;
        loop(m1, put i.l(m,m1):10:1; );
        put /;);
put /;
put /;
put " U  Generation " /;
put "          ";
loop(m, put m.tl:10; );
put /;
```

```
put "      ";
loop(m, put e.l(m):10:1; );
put /;
put /;
put " R  Consumption" /;
put "           ";
loop(m, put m.tl:10; );
put /;
put "      ";
loop(m, put r.l(m):10:1; );
put /;
```

The results are:

```
I  Power Flow
              a1        a2        a3        a4        a5        a6        a7        a8
a1           0.0      60.0       0.0       0.0       0.0       0.0       0.0     130.0
a2         -60.0       0.0      50.0       0.0       0.0       0.0       0.0       0.0
a3           0.0     -50.0       0.0     -30.6       0.0       0.0      80.6       0.0
a4           0.0       0.0      30.6       0.0    -110.6       0.0       0.0       0.0
a5           0.0       0.0       0.0     110.6       0.0     -10.6       0.0       0.0
a6           0.0       0.0       0.0       0.0      10.6       0.0      89.4    -100.0
a7           0.0       0.0     -80.6       0.0       0.0     -89.4       0.0       0.0
a8        -130.0       0.0       0.0       0.0       0.0     100.0       0.0       0.0

 U  Generation
          a1        a2        a3        a4        a5        a6        a7        a8
       190.0       0.0       0.0       0.0     100.0       0.0       0.0       0.0

 R  Consumption
          a1        a2        a3        a4        a5        a6        a7        a8
         0.0      10.0       0.0      80.0       0.0       0.0     170.0      30.0
```

This example served as the basis for the development of the model of water-energy market designed by Kyrgyzenergo (see A. Zyryanov and E. Antipova, Vol. 2, Section 2.1, "Optimization of the Syrdarya Water and Energy Uses under Current Conditions," McKinney, D.C. and A.K. Kenshimov (eds.), 2000).

7.2 Defining Voltage and Strength of Current in Direct-Current Circuit

The problem considered here is to determine the unknown variables of a direct current circuit of any configuration comprised of power sources and resistances. Kirkoff's and Ohm's laws are used in the model:

- **Kirkoff's law:** the sum of currents in each of the nodes is equal to zero.
- **Ohm's law:** the decrease in voltage across a resistor is proportional to the resistance multiplied by the current.

Consider the scheme shown in Figure 7.2.1.

```
SETS
m   /a1,a2,a3,a4,a5,a6,a7,a8   /
alias(m,m1);
```

```
alias(m,m2);

sets
mr(m)  consumers    /a2,a8,a4,a7/
me(m)  generators   /a1,a5/
m0(m)  junctions    /a3,a6/

m_and_m1(m,m1)         connection
/
a1.a2, a1.a8,
a2.a1, a2.a3,
a3.a2, a3.a4, a3.a7
a4.a3, a4.a5,
a5.a4, a5.a6,
a6.a5, a6.a8, a6.a7,
a7.a3, a7.a6,
a8.a1, a8.a6   /

m1_m_m2(m1,m,m2);
m1_m_m2(m1,m,m2)= yes$(m_and_m1(m1,m) and m_and_m1(m,m2));
m1_m_m2(m1,m,m1)$m1_m_m2(m1,m,m1)= no;

* If there is a connection between the "m", "m1" and "m2",
* then there is a connection between "m", "i1" and "m2".
* However, if "m1" coincides with "m2",
* it is not a double connection
* but a connection based on the principle "TO- FROM".
* It is excluded from the whole set of triple connections.

VARIABLES
* Current source
e(m),

* Resistors
r(m),

* Voltages between nodes
u(m,m1),

* Currents between nodes
i(m,m1),

obj;

EQUATION
* Calculate currents
coni(m,m1),

* Calculate voltages
conu(m,m1),

* Kirchhoff ‘s Law
ii(m),

* Ohm’s Law for junction nodes
uu_0(m1,m,m2),

* Ohm’s Law for supply nodes
uu_e(m1,m,m2),
```

```
* Ohm's Law for resistor nodes
uu_r(m1,m,m2),

ben;

* Kirchhoff 's Law
ii(m)..            sum(m1$m_and_m1(m,m1),i(m,m1)) =e= 0;

* Calculate currents
coni(m,m1)..       i(m,m1) =e= -i(m1,m)   ;

* Calculate voltages
conu(m,m1)..       u(m,m1) =e=  u(m1,m)   ;

* Ohm's Law for junction nodes
uu_0(m1,m,m2)$m0(m)..
    0  =e=  (u(m,m1)-u(m,m2))$m1_m_m2(m1,m,m2);

* Ohm's Law for supply nodes
uu_e(m1,m,m2)$(me(m)$(m1_m_m2(m1,m,m2)$( ord(m1) gt  ord(m2))))..
    u(m,m1)-u(m,m2) =e= e(m);

* Ohm's Law for resistor nodes
uu_r(m1,m,m2)$(mr(m)$m1_m_m2(m1,m,m2))..
    u(m,m1)-u(m,m2) =e= -r(m)*i(m,m1);

ben..            obj =E=i('a1','a8')*i('a1','a8');

e.up(m)=          100;
e.lo(m)=         -100;
r.up(m)=          100;
r.lo(m)=         -100;
i.l(m,m1)$m_and_m1(m,m1)=    -1 ;
i.up(m,m1)=         100;
i.lo(m,m1)=        -100;
u.up(m,m1)=         100;
u.lo(m,m1)=        -100;

r.fx('a2')= 10;
r.fx('a4')= 20;
r.fx('a7')= 40;

e.fx('a1')=20;
e.fx('a5')=40;

i.fx('a1','a2')=    0.125;

MODEL VAN /ALL/;
SOLVE VAN USING NLP MINIMIZING obj;

file res /odc2.txt/
put res;
put " m_AND_m1" /;
loop(m,  put m.tl:6;
         loop(m1, put M_and_M1(M,M1):7; );
put /;);
put /;
put " m_m_M1" /;
loop(m1,
     loop(m,
```

```
         loop(m2, put M1_m_M2(m1,M,M2):7; );
put /;);
put /;);
put " I amper  " /;
put "          ";
loop(m, put m.tl:7; );
put /;
loop(m,  put m.tl:6;
         loop(m1, put i.l(M,m1):7:3; );
put /;);
put /;
put " U volt " /;
put "          ";
loop(m, put m.tl:7; );
put /;
loop(m,  put m.tl:6;
         loop(m1, put u.l(M,m1):7:1; );
put /;);
put /;
put " E volt on supplies " /;
put "          ";
loop(m, put m.tl:6; );put /;
put "      ";
loop(m, put e.l(m):6:2; );put /;
put /;
put " R - resisters  " /;
put "          ";
loop(m, put m.tl:6; );put /;
put "      ";
loop(m, put r.l(m):6:2; );put /;
```

The results are:

```
I amper
          a1     a2     a3     a4     a5     a6     a7     a8
a1     0.000  0.125  0.000  0.000  0.000  0.000  0.000 -0.125
a2    -0.125  0.000  0.125  0.000  0.000  0.000  0.000  0.000
a3     0.000 -0.125  0.000  0.750  0.000  0.000 -0.625  0.000
a4     0.000  0.000 -0.750  0.000  0.750  0.000  0.000  0.000
a5     0.000  0.000  0.000 -0.750  0.000  0.750  0.000  0.000
a6     0.000  0.000  0.000  0.000 -0.750  0.000  0.625  0.125
a7     0.000  0.000  0.625  0.000  0.000 -0.625  0.000  0.000
a8     0.125  0.000  0.000  0.000  0.000 -0.125  0.000  0.000

 U volt
          a1     a2     a3     a4     a5     a6     a7     a8
a1       0.0  -10.4    0.0    0.0    0.0    0.0    0.0    9.6
a2     -10.4    0.0  -11.6    0.0    0.0    0.0    0.0    0.0
a3       0.0  -11.6    0.0  -11.6    0.0    0.0  -11.6    0.0
a4       0.0    0.0  -11.6    0.0  -26.6    0.0    0.0    0.0
a5       0.0    0.0    0.0  -26.6    0.0   13.4    0.0    0.0
a6       0.0    0.0    0.0    0.0   13.4    0.0   13.4   13.4
a7       0.0    0.0  -11.6    0.0    0.0   13.4    0.0    0.0
a8       9.6    0.0    0.0    0.0    0.0   13.4    0.0    0.0

 E volt on supplies
         a1    a2    a3    a4    a5    a6    a7    a8
      20.00  0.00  0.00  0.00 40.00  0.00  0.00  0.00

 R - resisters
```

```
      a1     a2     a3     a4     a5     a6     a7     a8
0.00 10.00  0.00 20.00  0.00  0.00 40.00 30.00
```

This example served as the basis for the development of the models on power networks (see S. Zaitseva, Sh. Khisoriev, and A. Savitsky, Vol. 2, Section 3.2, "Optimization of Electric Mode of Energy Systems Operation," McKinney, D.C. and A.K. Kenshimov (eds.), 2000).

7.3 Passing a Complex Electrical Signal

In the previous electricity examples we considered direct and alternating current in complex electrical circuits. However, the alternating current and voltage in these examples only varied sinusoidally. Sometimes, it is necessary to deal with circuits in which the pulses of current are not sine waves. The solution of such tasks is complicated. In this section, we solve this problem using GAMS, in the hope that it is useful in construction of more complicated models consisting of capacity, inductivity and other electronic devices and possibly microchips. First we describe the governing equations which are necessary to solve the task.

Ohm's Law. The voltage drop across a resistor is proportional to the current:

$$V_i - V_j = I_{i,j} R_{i,j}$$

where

V_i, V_j	Voltage at points i and j [volt];
$I_{i,j}$	Current flowing between points i and j [amper]; and
$R_{i,j}$	Resistance between points i and j [ohm]

Kirchoff's Law. The sum of currents into or out of any point is zero:

$$\sum_j I_{i,j} = 0 \qquad \forall i \quad (\text{i not a source of current})$$

Current source. For a source of current we have a difference in potentials:

$$V_i - V_j = E_{i,j}$$

where $E_{i,j}$ voltage at points i and j on either side of the source of current.

Capacitor. Charge accumulation causes a voltage increases across a capacitor inversely proportional to the capacitance:

$$\partial V = \partial Q / C = I \partial t / C$$

where

∂	Differential,
Q	Charge [coul],

C Capacity [farad],
t Time [sec],

or

$$\frac{\partial V}{\partial t} = \frac{I}{C}$$

Here, we deal with capacitors in networks and the accumulation of charge on one plate causes accumulationof on the opposite plate and the voltages on both sides are equal. As a result of the presence of two plates in the capacitor, we may substitute a factor of 2 in the right hand side of the previous equation. However if the electrical capacity device does not contain 2 plates then the factor of 2 is not required, e.g., if there is an situation with a body capable of accumulating charge on the surface and subsequently giving it back.

Inductor. Current change produces a potential difference proportional to inductivity:

$$\frac{\partial I}{\partial t} = V_j - V_j$$

where L is inductivity [henry].

Consider the following electrical circuit

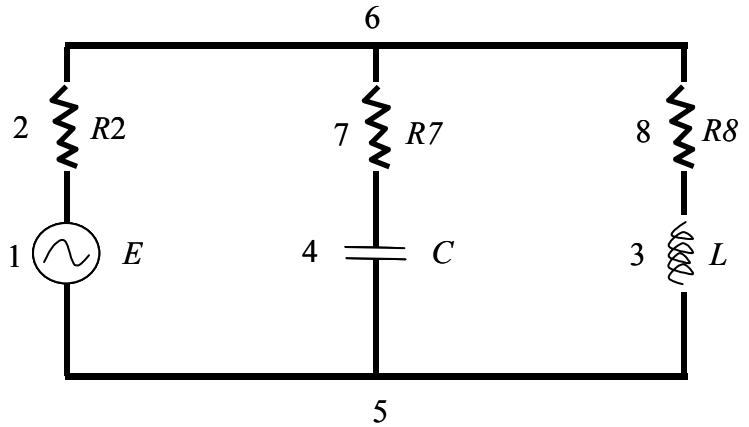


Figure 7.3.1. Simple circuit.

where

L = 0.023 henry,
C = 0.000023 faraday,
R_2 = 5 ohm,
R_7 = 7 ohm,
R_8 = 11 ohm,
ω = 18,840 hertz ,
E = $\sqrt{34}\sin(\omega t)$ volt

Or in complex form

$$E = 5 + 3j$$

The task can be solved by the method attributed to Gauss. We shall receive the system of equations

$$IR_2 + I_L(R_8 + j\omega L) = E$$
$$IR_2 + I_C(R_7 - \frac{j}{\omega C}) = E$$
$$I - I_C - I_L = 0$$

or in matrix form

$$\begin{bmatrix} 5 & 0 & 11+433.32j \\ 5 & 7-2307.7j & 0 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} I \\ I_C \\ I_L \end{bmatrix} = \begin{bmatrix} 5+3j \\ 5+3j \\ 0 \end{bmatrix}$$

with solution

$$\begin{bmatrix} I \\ I_C \\ I_L \end{bmatrix} = \begin{bmatrix} 0.006 - 0.009126j \\ -0.001313 + 0.002157j \\ 0.00713 - 0.01128j \end{bmatrix}$$

or

$$\begin{bmatrix} |I| \\ |I_C| \\ |I_L| \end{bmatrix} = \begin{bmatrix} 0.01909 \\ 0.0025 \\ 0.0134 \end{bmatrix}$$

The GAMS model for this example is:

```
SETS
T / k1*k40 /
M / a1*a8  /

Parameters Volt(T), Cap(M), Ind(M);
Volt(T)  = SIN((ord(T)-1)/card(T) * 2 * 3.14)$(ord(T) gt 0);
Cap(M)   = -23/1000000000*18840*40/6.28 * 2;
Ind(M)   =   23/1000*18840*40/6.28;

Alias(M, M1);
Alias(M, M2);

Sets
ML(M)             inductivity      /a3/
MC(M)             capacity         /a4/
MR(M)             resisters        /a2, a7, a8/
ME(M)             electro supplies /a1/
M0(M)             nodes            /a5, a6/
M_and_M1(M, M1) connection
                     /a1.a2, a1.a5, a2.a1, a2.a6, a3.a8, a3.a5,
```

```
                    a4.a7, a4.a5, a5.a4, a5.a3, a5.a1, a6.a2,
                    a6.a7, a6.a8, a7.a4, a7.a6, a8.a6, a8.a3 /
M1_M_M2(M1, M, M2);
M1_M_M2(M1, M, M2) = yes$(M_and_M1(M1, M) and M_and_M1(M, M2));
M1_M_M2(M1, M, M1)$M1_M_M2(M1, M, M1) = no;

Parameter E(M);
Parameter R(M);
E('a1') = sqrt(34);
R('a2') =  5;
R('a7') =  7;
R('a8') = 11;

VARIABLES
*         E(M),
*         R(M),
         V(M, M1, T),
         I(M, M1, T),
         OBJ;

* E.fx('a1') = sqrt(34);
* R.fx('a2') =  5;
* R.fx('a7') =  7;
* R.fx('a8') = 11;

EQUATION
         ConI(M, M1, T),
         ConU(M, M1, T),
         I_I(M, T),
         V_0(M1, M, M2, T),
         V_E(M1, M, M2, T),
         V_R(M1, M, M2, T),
         V_L(M, M1, M2, T),
         V_C(M, M1, T),
         V_C1(M1, M, M2, T),
         BEN;

I_I(M, T)..  sum(M1$M_and_M1(M, M1), I(M,M1,T)) =e= 0;
* Kirchoff law

ConI(M,M1,T)..  I(M,M1,T) =e= -I(M1,M,T);
* If there is a connection between units M and M1

ConU(M,M1,T).. V(M,M1,T) =e= V(M1,M,T);
* Potential on connection between units does not vary,
* Or, no resistance on wires connecting elements

V_E(M1,M,M2,T)$(ME(M)$(M1_M_M2(M1,M,M2)$(ord(M1) gt ord(M2)))).. 

            E(M)*Volt(T) =e= V(M,M1,T) - V(M,M2,T);
* Source units have a potential difference

V_0(M1,M,M2,T)$((M0(M)) and (M1_M_M2(M1, M, M2))).. 

            V(M, M2, T) =e= V(M, M1, T);
* For simple units there is no power failure,
* Simple units are junctions

V_R(M1,M,M2,T)$((MR(M)) and (M1_M_M2(M1,M,M2))
                        and (ord(M2) gt ord(M1))).. 
```

```
            V(M1,M,T) - V(M2,M,T) =e= -I(M,M1,T)*R(M);
* Ohm's Law: the power loss is proportional
* to product of current and resistance

V_C1(M1,M,M2,T)$((MC(M)) and (M1_M_M2(M1,M,M2))).

            V(M1,M,T) =e= -V(M2,M,T);
* the plates of the condenser always have
* equal but opposite potential on the plates

V_C(M,M1,T)$((MC(M)) and (M_and_M1(M, M1))).

        V(M,M1,T) - V(M,M1, T -- 1) =e= + I(M,M1,T)/Cap(M);
* Increase in potential on a plate
* of the condenser is proportional to current

V_L(M1,M,M2,T)$((Ml(M)) and (M1_M_M2(M1,M,M2))
                        and (ord(M2) gt ord(M1))).

   V(M1,M,T) - V(M2,M,T) =e= -Ind(M)*(I(M,M1,T)-I(M,M1,T -- 1));
* Differences in voltage across inductor is
* proportional to the change in current,

Ben.. OBJ =e= 1;

MODEL IVAN / all /;
SOLVE IVAN USING LP minimizing obj;
```

The results of solving this model are:

```
Solution from GAUSS proceedure
 I Ic Il
   0.0109   0.0025   0.0134   5.3230
 Solution from GAMS
 I Ic Il
   0.0110   0.0026   0.0134   5.2353

 I amper
 a1      a2      a3      a4      a5      a6      a7      a8
a1       0.000   0.010   0.000   0.000  -0.010   0.000   0.000   0.000
a2      -0.010   0.000   0.000   0.000   0.000   0.010   0.000   0.000
a3       0.000   0.000   0.000   0.000   0.013   0.000   0.000  -0.013
a4       0.000   0.000   0.000   0.000  -0.003   0.000   0.003   0.000
a5       0.010   0.000  -0.013   0.003   0.000   0.000   0.000   0.000
a6       0.000  -0.010   0.000   0.000   0.000   0.000  -0.003   0.013
a7       0.000   0.000   0.000  -0.003   0.000   0.003   0.000   0.000
a8       0.000   0.000   0.013   0.000   0.000  -0.013   0.000   0.000

U Volt
 a1      a2      a3      a4      a5      a6      a7      a8
a1       0.000  -0.069   0.000   0.000   0.074   0.000   0.000   0.000
a2      -0.069   0.000   0.000   0.000   0.000  -0.077   0.000   0.000
a3       0.000   0.000   0.000   0.000   0.074   0.000   0.000  -0.099
a4       0.000   0.000   0.000   0.000   0.074   0.000  -0.074   0.000
a5       0.074   0.000   0.074   0.074   0.000   0.000   0.000   0.000
a6       0.000  -0.077   0.000   0.000   0.000   0.000  -0.077  -0.077
a7       0.000   0.000   0.000  -0.074   0.000  -0.077   0.000   0.000
a8       0.000   0.000  -0.099   0.000   0.000  -0.077   0.000   0.000

E Volt on supplies
```

a1 a2 a3 a4 a5 a6 a7 a8
5.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00

R - resisters
a1 a2 a3 a4 a5 a6 a7 a8
0.00 5.00 0.00 0.00 0.00 0.00 7.00 11.00

Supply		Current			Voltage		
A1		A2-A6,	A6-A7,	A6-A8	V8-V3	V4-V7	V4-V5
Time	Voltage	Amp	Amp	Amp	Volt	Volt	Volt
1.00	0.0000	0.0108	-0.0026	0.0133	-0.1825	-0.0179	0.0179
2.00	0.9117	0.0104	-0.0025	0.0129	-0.6329	-0.4731	0.4731
3.00	1.8010	0.0098	-0.0024	0.0122	-1.0677	-0.9163	0.9163
4.00	2.6460	0.0089	-0.0023	0.0112	-1.4763	-1.3370	1.3370
5.00	3.4258	0.0078	-0.0021	0.0099	-1.8486	-1.7248	1.7248
6.00	4.1215	0.0064	-0.0019	0.0084	-2.1754	-2.0702	2.0702
7.00	4.7157	0.0050	-0.0016	0.0066	-2.4487	-2.3646	2.3646
8.00	5.1939	0.0034	-0.0013	0.0047	-2.6618	-2.6009	2.6009
9.00	5.5444	0.0017	-0.0010	0.0027	-2.8093	-2.7732	2.7732
10.00	5.7585	0.0000	-0.0006	0.0006	-2.8878	-2.8773	2.8773
11.00	5.8310	-0.0017	-0.0002	-0.0015	-2.8952	-2.9106	2.9106
12.00	5.7600	-0.0034	0.0002	-0.0036	-2.8315	-2.8723	2.8723
13.00	5.5473	-0.0050	0.0006	-0.0056	-2.6980	-2.7634	2.7634
14.00	5.1982	-0.0064	0.0010	-0.0074	-2.4982	-2.5864	2.5864
15.00	4.7212	-0.0077	0.0013	-0.0091	-2.2369	-2.3459	2.3459
16.00	4.1280	-0.0089	0.0016	-0.0105	-1.9206	-2.0476	2.0476
17.00	3.4334	-0.0098	0.0019	-0.0117	-1.5571	-1.6990	1.6990
18.00	2.6542	-0.0104	0.0022	-0.0126	-1.1553	-1.3086	1.3086
19.00	1.8098	-0.0108	0.0023	-0.0132	-0.7250	-0.8860	0.8860
20.00	0.9209	-0.0110	0.0025	-0.0134	-0.2769	-0.4416	0.4416
21.00	0.0093	-0.0108	0.0025	-0.0133	0.1780	0.0136	-0.0136
22.00	-0.9025	-0.0104	0.0025	-0.0129	0.6286	0.4686	-0.4686
23.00	-1.7921	-0.0098	0.0024	-0.0122	1.0636	0.9120	-0.9120
24.00	-2.6377	-0.0089	0.0023	-0.0112	1.4726	1.3329	-1.3329
25.00	-3.4183	-0.0078	0.0021	-0.0099	1.8453	1.7211	-1.7211
26.00	-4.1149	-0.0065	0.0019	-0.0084	2.1726	2.0670	-2.0670
27.00	-4.7102	-0.0050	0.0016	-0.0066	2.4464	2.3620	-2.3620
28.00	-5.1897	-0.0034	0.0013	-0.0047	2.6601	2.5989	-2.5989
29.00	-5.5415	-0.0018	0.0010	-0.0027	2.8084	2.7718	-2.7718
30.00	-5.7570	-0.0001	0.0006	-0.0006	2.8876	2.8766	-2.8766
31.00	-5.8309	0.0017	0.0002	0.0015	2.8957	2.9107	-2.9107
32.00	-5.7614	0.0033	-0.0002	0.0035	2.8327	2.8731	-2.8731
33.00	-5.5501	0.0049	-0.0006	0.0055	2.6999	2.7648	-2.7648
34.00	-5.2024	0.0064	-0.0010	0.0074	2.5008	2.5886	-2.5886
35.00	-4.7266	0.0077	-0.0013	0.0090	2.2401	2.3487	-2.3487
36.00	-4.1346	0.0088	-0.0016	0.0105	1.9243	2.0510	-2.0510
37.00	-3.4409	0.0097	-0.0019	0.0117	1.5612	1.7028	-1.7028
38.00	-2.6625	0.0104	-0.0022	0.0125	1.1597	1.3128	-1.3128
39.00	-1.8186	0.0108	-0.0023	0.0131	0.7297	0.8905	-0.8905
40.00	-0.9300	0.0109	-0.0025	0.0134	0.2817	0.4462	-0.4462

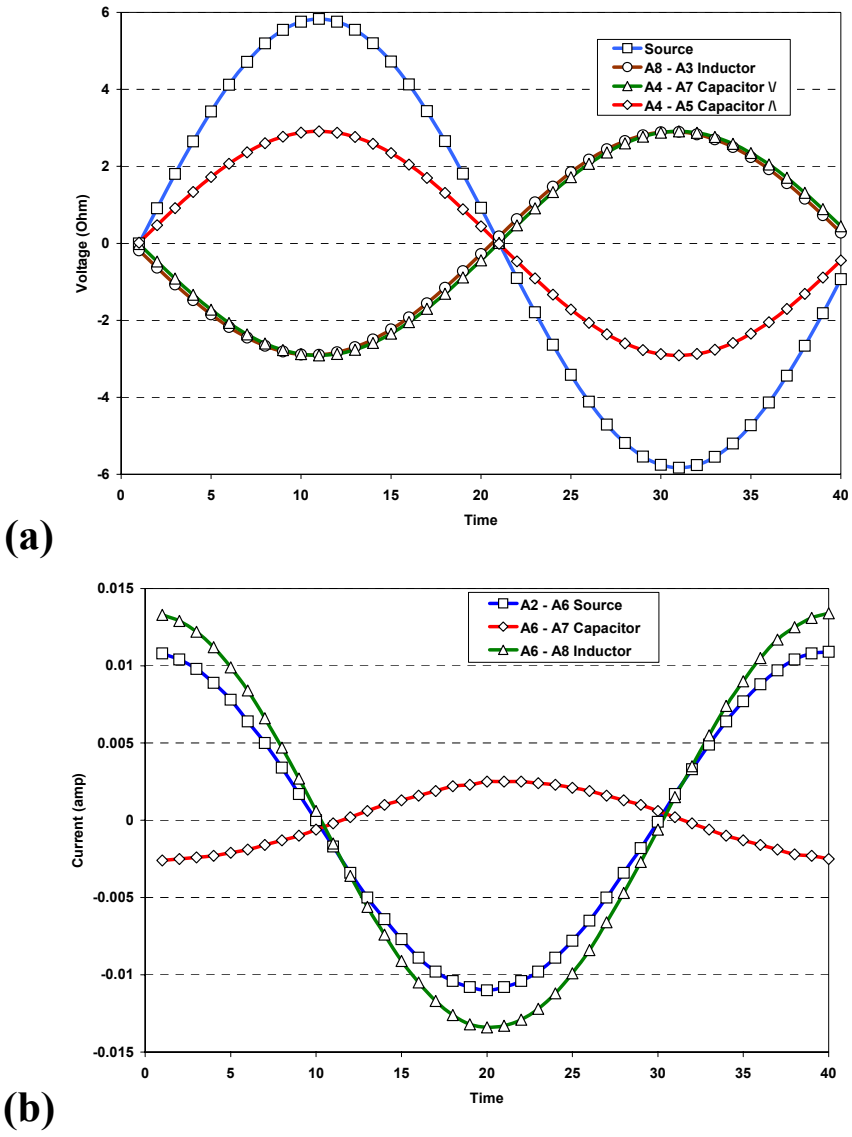


Figure 7.3.2. Results of simple circuit example: (a) voltage, (b) current.

All characteristics of the circuit can enter the model as variables. That is, the sizes and caracteristics of resistors and capacitors can be decision variables. This example is solved for a sine function input:

```
Volt(t) = SIN((ord(t)-1)/card(t)*2*3.14)$(ord(t) gt 0);
```

Actually, the model can handle a pulse of any form. Readers can change

```
(ord(t) gt 0) to (ord(t) gt 25)
```

and see how the pulse of current behaves in the circuit.

In the following example we shall show that the elementary task opens a whole area of tasks related

to electrical circuits. The equations of the resistor, the capacitor and inductor are the elementary converters of the input signal, but it is possible to include in the circuit diodes, transistors and microcircuits. For example, we shall illustrate this by adding a diode to the circuit. Diodes have the property that for current in one direction there is a resistance distinct from resistance to current in the opposite direction. As a rule these resistances differ by orders of magnitude.

To test this model we consider a "bridge" problem (Figure 3) where a sinusoidal signal is changed into a positive electricity flow.

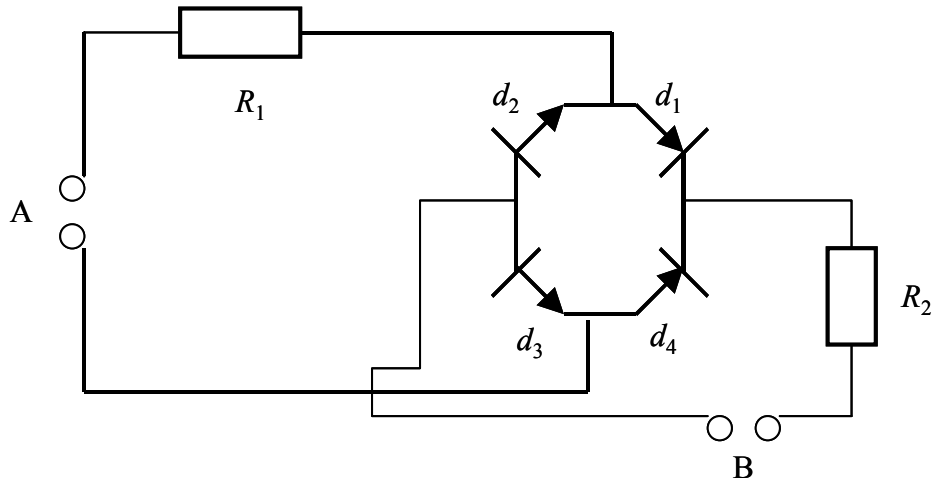


Figure 7.3.3. Bridge circuit.

From a source of alternating current, a sinusoudal wave of voltage is produced, causing a sine wave current in the circuit. Current flowing from left to right through resistor R_1 will go through diode d_1 → R_2 → d3 and come back to the source at point A. If the current moves along the bottom from the source, it will pass through d_4 → R_2 → d_1. In the model, we must enter a new set determining the direction of current passing through diodes in the circuit. We name it (see Figure 4)

```
direct(M,M1) / a3.a8, a7.a10, a11.a8, a4.a6/
```

We must also add the equation responsible for directing the current flow in the diodes:

```
V_D_V(M1,M,M2,t)$((MD(M))and(M1_M_M2(M1,M,M2))and(direct(M,M1)))

.. V(M1,M,T) - V(M2,M,T)
   =e= 2000*(sqrt(I(M,M1,T)*I(M,M1,T)) - I(M,M1,T))/2;
```

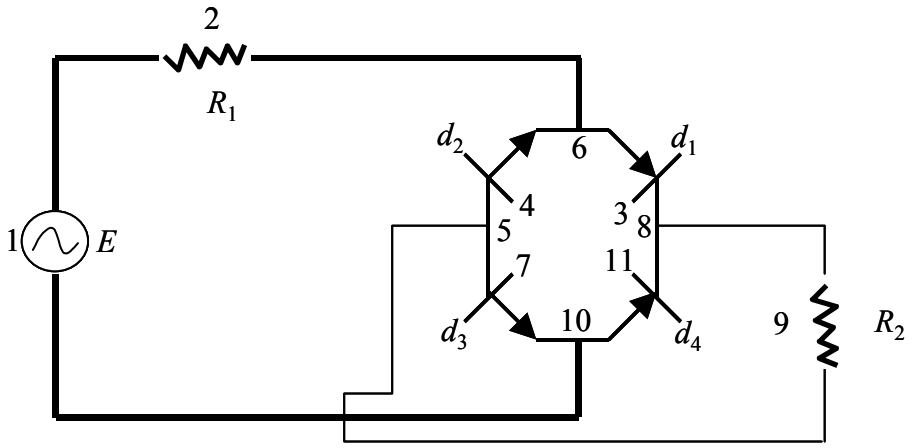


Figure 7.3.4. New circuit with diodes as bridge.

This equation has three conditions:

1. MD(M) Condition of belonging to the group of diodes;
2. M1_M_M2(M1,M,M2) Condition of existence of a connection between units M1 and M2 through unit M; and
3. direct(M,M1) Condition that M → M1 is the direction of the diode.

The left side of the equation indicates the voltage drop across the unit. The right part is zero for positive current and double its size for negative current.

In table # the currents going through the bridge through first second branch and resistance are shown. The complete agreement of known values with the decision received is visible in GAMS. From the table it is seen that the current goes and on "closed" branch of diodes reducing efficiency the straightening "bridge"-network. By other computing methods the so thin phenomenon is difficult to compute. If the user enters something other than a sine wave signal other methods will not be able to solve the problem.

In fig. # is shown sine of a source of a current and currents in branhes of the "bridge" circuit. The reader can add to the resistor of the bridge capacity and inductance thus increasing the quality of the straightened signal. In the objective function it is possible to enter a parameter of quality of the output of the signal and to find capacity and inductance needed to achieve this quality. Tasks of thia type are very difficult to solve in other languages. The model presented here provides a great opportunity for solving these problems. Once again, we emphasize that from a number of simple examples the creation of large computing systems practical experiments is possible.

7.4 Alternating Current Circuits

The elementary model of circuits with a constant current was discussed above. It is possible to consider circuits with a sine wave current. This example was elaborate by Ivan Savitsky.

For this task we use the laws Ohm and Kirchoff. The difference is that they are written in the complex form and there are imaginary parts for current, voltage and resistance. To the usual, or active, resistance is added an implicit, or reactive, resistance. The resistance is the reaction of the circuit to the varying characteristics of the current (voltage and current).

Capacitor. The implicit resistance of a capacitor is given by

$$R_C = \frac{1}{j\omega C}$$

Inductor. Implicit resistance of an inductor is given by

$$R_L = j\omega L$$

where

ω	Frequency of fluctuations of the variable power supply;
L, C	Capacitors and inductors; and
j	Imaginary unit, $\sqrt{-1}$.

Active resistance is the real part of the complex resistance, and implicit resistance is the imaginary part. Active resistance is only the real part, and the imaginary part of this type of resistance is zero. Capacitative and inductive resistance is formed only by the imaginary part of the complex numbers.

Ohm's Law:

$$V_i - V_{i-1} = I * Z(j\omega)$$

where

V_i	complex potential on one side of an object;
$V_{i\text{-}1}$	complex potential on the other side of an object;
I	complex current; and
$Z(j\omega)$	complex resistance.

Kirchoff's Law:

The sum of complex currents flowing into a unit and leaving a unit must equal zero. That is, the sum of real currents and the sum of imaginary currents on each unit must equal zero. The law reflecting invariancy and interconnection of the network as one system is

$$I_{mn} = -I_{nm}$$

The current leaving unit *m* to point *n* is the current which comes into point *n* from unit *m*. The mark (+) indicates outflow from a unit, and (-) indicates inflow to unit. For a source of current the difference of voltage is equal to the voltage of the source. For resistors, the resistance is given by Ohm's law. The voltages are not equal on each side, but the current entering is equal to that leaving. For junctions, the

voltage of all connections are equal among themselves.

Consider the electrical circuit shown in Figure 7.4.1.

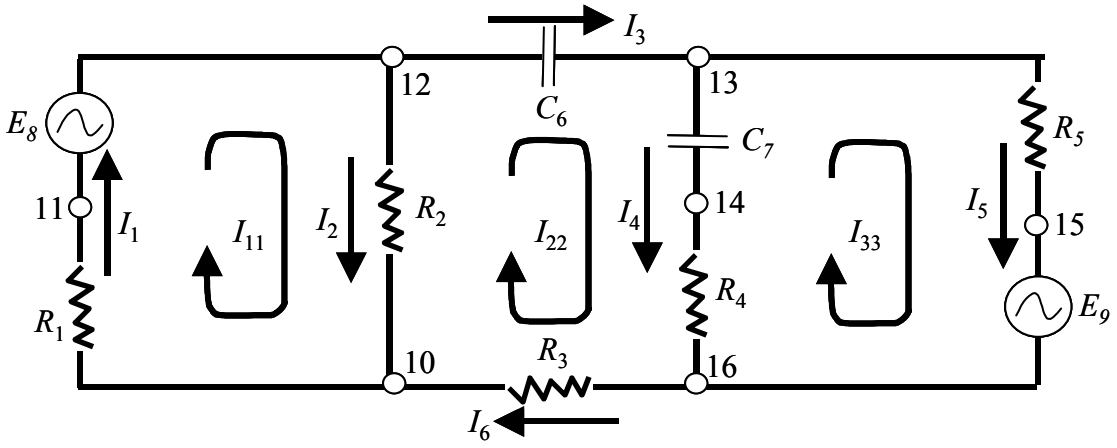


Figure 7.4.1. Electrical circuit.

In this circuit, we have the following characteristics of the components:

Sources: $E_8 = 20 + 30j$; $E_9 = 30 - 10j$ (amp);
Capacitors: $C_6 = 30$; $C_7 = 40$ (farad);
Resistors: $R_1 = 10$; $R_2 = 20$; $R_3 = 30$; $R_4 = 40$; and $R_5 = 50$ (ohm);

Let's make the equations on a method of planimetric currents for 1,2,3-circle lines:

$$I_{11}(R_1 + R_2) - I_{22}R_2 = E_8$$
$$I_{22}(R_2 + \frac{1}{j\omega C_6} + \frac{1}{j\omega C_7} + R_4 + R_3) - I_{11}R_2 - I_{33}(R_4 + \frac{1}{j\omega C_7}) = 0$$
$$I_{33}(R_4 + \frac{1}{j\omega C_7} + R_5) - I_{22}(\frac{1}{j\omega C_7} + R_4) = E_9$$

or

$$30I_{11} - 20I_{22} = 20 + 30j$$
$$-20I_{11} + (90 - 1166.66j)I_{22} - (40 - 500j)I_{33} = 0$$
$$0I_{11} - (40 - 500j)I_{22} + (90 - 500j)I_{33} = 30 - 10j$$

which can be written in the following form:

$$\begin{bmatrix} 30 & -20 & 0 \\ -20 & (90 - 1166.66j) & -(40 - 500j) \\ 0 & -(40 - 500j) & (90 - 500j) \end{bmatrix} \begin{bmatrix} I_{11} \\ I_{22} \\ I_{33} \end{bmatrix} = \begin{bmatrix} 20 + 30j \\ 0 \\ 30 - 10j \end{bmatrix}$$

Solving the given system of the equations for the frequency of an alternating current with $\omega = 50$ (Hertz) gives:

$$\begin{bmatrix} I_{11} \\ I_{22} \\ I_{33} \end{bmatrix} = \begin{bmatrix} 0.6647885 + 1.04178\,jj \\ -0.002871 + 0.06267\,j \\ 0.03313 + 0.1164806\,j \end{bmatrix}$$

Then

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \end{bmatrix} = \begin{bmatrix} I_{11} \\ I_{11} - I_{22} \\ I_6 = I_{22} \\ I_{33} - I_{22} \\ I_{33} \end{bmatrix} \begin{bmatrix} 0.6647885 + 1.04178\,j \\ 0.6675 + 0.979\,j \\ -0.0028 + 0.0626\,j \\ 0.0359 + 0.0538\,j \\ 0.0331 + 0.1164\,j \end{bmatrix}$$

The GAMS model for this example is:

```
SETS
M /A1*A16/;

Alias(M,M1);
Alias(M,M2);

Sets
         MR(M)   resistors    /a1*a7/
         ME(M)   sources      /a8*a9/
         M0(M)   nodes        /a10*a16/
         M_to_M1(M1,M2)  connections near batery
                         /a12.a11
                          a16.a15/
         M_and_M1(M,M1)  connections
/
A1.A10, A1.A11, A2.A10, A2.A12, A3.A10, A3.A16, A4.A14, A4.A16, A5.A13
A5.A15, A6.A12, A6.A13, A7.A13, A7.A14, A8.A11, A8.A12, A9.A16, A9.A15
A10.A1, A10.A2, A10.A3, A11.A1, A11.A8, A12.A2, A12.A8, A12.A6, A13.A5
A13.A6, A13.A7, A14.A4, A14.A7, A15.A5, A15.A9, A16.A3, A16.A4, A16.A9
/

         M1_M_M2(M1,M,M2);
         M1_M_M2(M1,M,M2)  =yes$(M_and_M1(M1,M) and M_and_M1(M,M2));
         M1_M_M2(M1,M,M1)$M1_M_M2(M1,M,M1)  = no;

Variables
         E_d(M)    real supplies,
         E_k(M)    implicit supplies,
         R_d(M)    real resistance,
         R_k(M)    implicit resistance,
         V_d(M,M1) real voltage,
         V_k(M,M1) implicit voltage,
         I_d(M,M1) real current,
         I_k(M,M1) implicit current,
         OBJ;
```

```
I_d.up(M,M1)  =    1000;
I_d.lo(M,M1)  =   -1000;
V_d.up(M,M1)  =    1000;
V_d.lo(M,M1)  =   -1000;

R_d.FX('a1')= 10;
R_d.FX('a2')= 20;
R_d.FX('a3')= 30;
R_d.FX('a4')= 40;
R_d.FX('a5')= 50;
R_d.FX('a6')=  0;
R_d.FX ('a7')= 0;
R_k.FX('a1')=  0;
R_k.FX('a2')=  0;
R_k.FX('a3')=  0;
R_k.FX('a4')=  0;
R_k.FX('a5')=  0;
R_k.FX('a6')= -666.6666;
R_k.FX('a7')= -500;
E_d.FX('a8')=  20;
E_d.FX('a9')=  30;
E_k.FX('a8')=  30;
E_k.FX('a9')= -10;

Equation
         ConI_d(M,M1),
         ConV_d(M,M1),
         II_d(M),
         V_0_d(M1,M,M2),
         V_E_d(M1,M,M2),
         V_R_d(M1,M,M2),
         ConI_k(M,M1),
         ConV_k(M,M1),
         II_k(M),
         V_0_k(M1,M,M2),
         V_E_k(M1,M,M2),
         V_R_k(M1,M,M2),
         BEN;
*
II_d(M)..       sum(m1$m_and_m1(M,M1),I_d(M,M1)) =e= 0;
ConI_d(M,M1)..  I_d(M,M1) =e= -I_d(M1,M);
ConV_d(M,M1)..  V_d(M,M1) =e=  V_d(M1,M);

V_0_d(M1,M,M2)$m0(M)..

     0 =e= (V_d(M,M1)-V_d(M,M2))$M1_M_M2(M1,M,M2);

V_E_d(M1,M,M2)$(ME(M)$(M1_M_M2(M1,M,M2)$(M_to_M1(M1,M2)))))..

     V_d(M,M1)-V_d(M,M2) =e= e_d(M);

V_R_d(M1,M,M2)$(MR(M)$M1_M_M2(M1,M,M2))..

     V_d(M,M1)-V_d(M,M2) =e= -(R_d(M)*I_d(M,M1)-R_k(M)*I_k(M,M1));

*
II_k(M)..       sum(m1$m_and_m1(M,M1),I_k(M,M1)) =e= 0;
ConI_k(M,M1)..  I_k(M,M1) =e= -I_k(M1,M)  ;
ConV_k(M,M1)..  V_k(M,M1) =e=  V_k(M1,M)  ;
```

```
V_0_k(M1,M,M2)$M0(M)..

     0 =e= (V_k(M,M1)-V_k(M,M2)) $M1_M_M2(M1,M,M2);

V_E_k(M1,M,M2)$(ME(M)$(M1_M_M2(M1,M,M2)$(M_to_M1(M1,M2)))).

     V_k(M,M1)-V_k(M,M2) =e= e_k(M);

V_R_k(M1,M,M2)$(MR(M) $M1_M_M2(M1,M,M2)).

     V_k(M,M1)-V_k(M,M2) =e= -(R_d(M)*I_k(M,M1)+R_k(M)*I_d(M,M1));
*
BEN..       OBJ =E=1;

MODEL VAN /ALL/;
Option nlp = Minos5;
SOLVE VAN USING NLP maximizing OBJ;
```

The results given in Tables 1 and 2 show that the GAMS model produces the same result as the Gauss method solution.

Table 1. Current I (amp, real part)

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16
A1	0	0	0	0	0	0	0	0	0	-0.66	0.66	0	0	0	0	0
A2	0	0	0	0	0	0	0	0	0	0.67	0	-0.67	0	0	0	0
A3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A4	0	0	0	0	0	0	0	0	0	0	0	0	0	0.04	0	-0.04
A5	0	0	0	0	0	0	0	0	0	0	0	0	-0.03	0	0.03	0
A6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A7	0	0	0	0	0	0	0	0	0	0	0	0	0.04	-0.04	0	0
A8	0	0	0	0	0	0	0	0	0	0	-0.66	0.66	0	0	0	0
A9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.03	0.03
A10	0.66	-0.67	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A11	-0.66	0	0	0	0	0	0	0.66	0	0	0	0	0	0	0	0
A12	0	0.67	0	0	0	0	0	-0.66	0	0	0	0	0	0	0	0
A13	0	0	0	0	0.03	0	-0.04	0	0	0	0	0	0	0	0	0
A14	0	0	0	-0.04	0	0	0.04	0	0	0	0	0	0	0	0	0
A15	0	0	0	0	-0.03	0	0	0	0.03	0	0	0	0	0	0	0
A16	0	0	0	0.04	0	0	0	0	-0.03	0	0	0	0	0	0	0

Current I (amp, implicit part)

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16
A1	0	0	0	0	0	0	0	0	0	-1.04	1.04	0	0	0	0	0
A2	0	0	0	0	0	0	0	0	0	0.98	0	-0.98	0	0	0	0
A3	0	0	0	0	0	0	0	0	0	0.06	0	0	0	0	0	-0.06
A4	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05	0	-0.05
A5	0	0	0	0	0	0	0	0	0	0	0	0	-0.12	0	0.12	0
A6	0	0	0	0	0	0	0	0	0	0	0	-0.06	0.06	0	0	0
A7	0	0	0	0	0	0	0	0	0	0	0	0	0.05	-0.05	0	0
A8	0	0	0	0	0	0	0	0	0	0	-1.04	1.04	0	0	0	0
A9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.12	0.12
A10	1.04	-0.98	-0.06	0	0	0	0	0	0	0	0	0	0	0	0	0
A11	-1.04	0	0	0	0	0	0	1.04	0	0	0	0	0	0	0	0
A12	0	0.98	0	0	0	0.06	0	-1.04	0	0	0	0	0	0	0	0
A13	0	0	0	0	0.12	-0.06	-0.05	0	0	0	0	0	0	0	0	0
A14	0	0	0	-0.05	0	0	0.05	0	0	0	0	0	0	0	0	0
A15	0	0	0	0	-0.12	0	0	0	0.12	0	0	0	0	0	0	0
A16	0	0	0.06	0.05	0	0	0	0	-0.12	0	0	0	0	0	0	0

Table 2. Voltage V (volt, real part)

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16
A1	0	0	0	0	0	0	0	0	0	0.08	-6.56	0	0	0	0	0
A2	0	0	0	0	0	0	0	0	0	0.08	0	13.44	0	0	0	0
A3	0	0	0	0	0	0	0	0	0	0.08	0	0	0	0	0	0
A4	0	0	0	0	0	0	0	0	0	0	0	0	0	-1.44	0	0
A5	0	0	0	0	0	0	0	0	0	0	0	0	-28.34	0	-30	0
A6	0	0	0	0	0	0	0	0	0	0	0	13.44	-28.34	0	0	0
A7	0	0	0	0	0	0	0	0	0	0	0	0	-28.34	-1.44	0	0
A8	0	0	0	0	0	0	0	0	0	0	-6.56	13.44	0	0	0	0
A9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-30	0
A10	0.08	0.08	0.08	0	0	0	0	0	0	0	0	0	0	0	0	0
A11	-6.56	0	0	0	0	0	0	-6.56	0	0	0	0	0	0	0	0
A12	0	13.44	0	0	0	13.44	0	13.44	0	0	0	0	0	0	0	0
A13	0	0	0	0	-28.34	-28.34	-28.34	0	0	0	0	0	0	0	0	0
A14	0	0	0	-1.44	0	0	-1.44	0	0	0	0	0	0	0	0	0
A15	0	0	0	0	-30	0	0	0	-30	0	0	0	0	0	0	0
A16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Voltage V (volt, implicit part)

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16
A1	0	0	0	0	0	0	0	0	0	-11.88	-22.3	0	0	0	0	0
A2	0	0	0	0	0	0	0	0	0	-11.88	0	7.7	0	0	0	0
A3	0	0	0	0	0	0	0	0	0	-11.88	0	0	0	0	0	-10
A4	0	0	0	0	0	0	0	0	0	0	0	0	0	-12.15	0	-10
A5	0	0	0	0	0	0	0	0	0	0	0	0	5.82	0	0	0
A6	0	0	0	0	0	0	0	0	0	0	0	7.7	5.82	0	0	0
A7	0	0	0	0	0	0	0	0	0	0	0	0	5.82	-12.15	0	0
A8	0	0	0	0	0	0	0	0	0	0	-22.3	7.7	0	0	0	0
A9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-10
A10	-11.88	-11.88	-11.88	0	0	0	0	0	0	0	0	0	0	0	0	0
A11	-22.3	0	0	0	0	0	0	-22.3	0	0	0	0	0	0	0	0
A12	0	7.7	0	0	0	7.7	0	7.7	0	0	0	0	0	0	0	0
A13	0	0	0	0	5.82	5.82	5.82	0	0	0	0	0	0	0	0	0
A14	0	0	0	-12.15	0	0	-12.15	0	0	0	0	0	0	0	0	0
A15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A16	0	0	-10	-10	0	0	0	0	-10	0	0	0	0	0	0	0

8. Optimal Solution of Heat Tranfer Problems

Many tasks, e.g., river and electrical networks, are obviously two-dimensional, but they were represented by one-dimensional systems as analogues in the equations. In this chapter we consider tasks which are not one-dimensional and some which are time dependent. For each example, along with the solution, we shall consider some additional options and operators of GAMS.

8.1 Background

The basic equation for the task is conservation of energy, as follows;

$$\frac{\partial(c\rho T)}{\partial t} = \nabla \cdot [V \nabla (c\rho T)] + I$$

where

t	time (sec),
c	specific heat capacity of substance (cal/(kg - degree),
p	density of substance of (kg/m^3),
T	temperature of substance (degree),
V	thermal conductivity (m^2/sec),
I	point source of heat (cal/(m^{3}sec).

Consider the task of solving this problem in a rectangular area in which there is a source of heat in the lower right-hand corner. On the boundary of the area the temperature is fixed

$$T = T_0(x, y) \qquad \text{for} \qquad (x, y) \text{ on the boundary}$$

If the situation is steady state, then no characteristics of the problem change with time. In this example, the thermal conductivity V is heterogeneous in the solution domain (see Figure 8.1). There is a symmetric zone at the center of the solution area where V becomes larger than in the rest of the area. In view of these assumptions the governing equation becomes

$$\nabla \cdot [V \nabla (c\rho T)] + I = 0$$

If c and ρ do not vary spatially, this equation becomes

$$\nabla \cdot (V \nabla T) + \frac{I}{c\rho} = 0$$

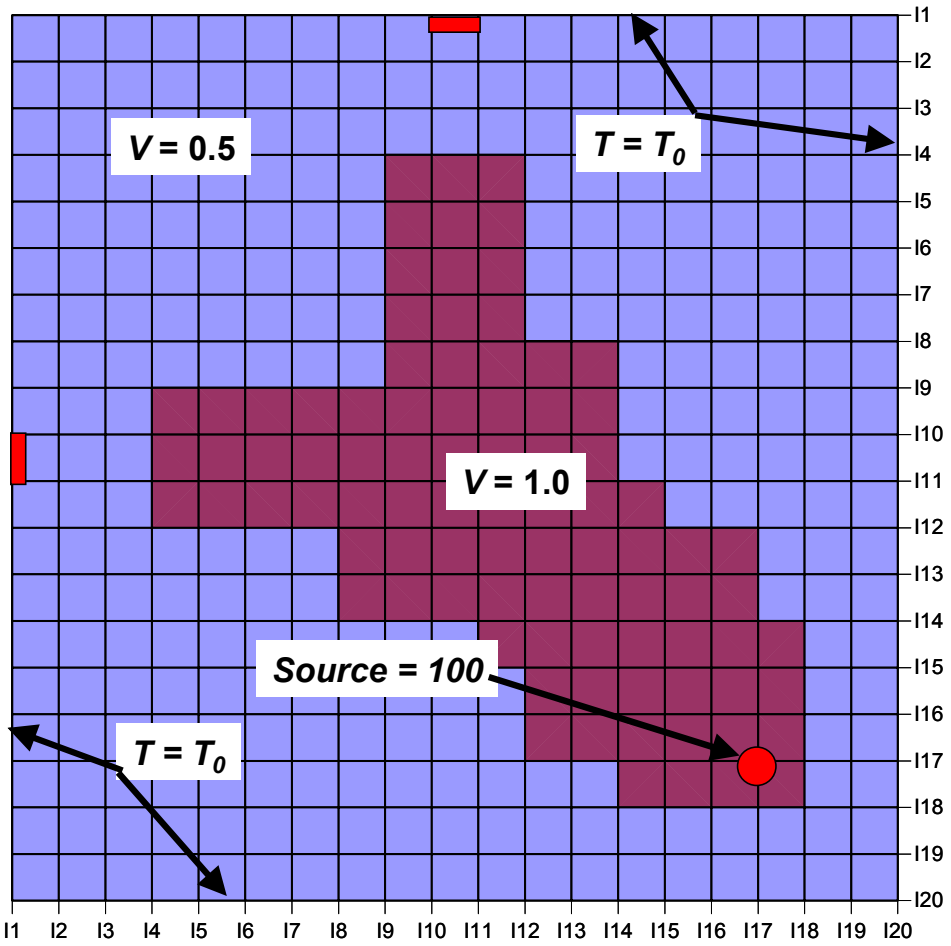


Figure 8.1.1. Solution domain for heat transfer problem.

This equation is difficult to solve due to the heterogeneity of the thermal conductivity (as opposed to $V\nabla \cdot \nabla T + I/c\rho = 0$). Also, the transient equation

$$\nabla \cdot (V\nabla T) + \frac{I}{c\rho} = \frac{\partial T}{\partial t}$$

is much easier and more commonly solved. A stationary solution to the transient equation is found by using a large number of iterations under stationary boundary conditions. After many iterations the influence of the initial conditions becomes very small and $\partial T/\partial t \to 0$. In any case, the differential operators must be approximated by algebraic analogues and a conservative scheme of solving the resulting system of algebraic equations must be used.

In this example we consider construction of the analog on a grid in more detail. In subsequent eamples we will not concentrate on this aspect as much. First, we subdivide the rectangular area into some number of small rectangles with sides parallel to the x and y axes. The result is a number of

intersection points refered to as nodes (see Figure 1 where a system of 20x20 nodes is illustrated). This system of rectangles allows to us construct algebraic analogs for the differential operators $\nabla \cdot ( )$ and $\nabla( )$.

The basic problem of constructing algebraic analogues of the differential operators is the preservation of invariancy. That is, the solution must not depend on the orientation of the coordinate axes, and symmetric areas under symmetric boundary conditions should result in symmetric solutions.

One of the variants of the differential operator in the previous equation can be derived using the well known formula from vector analysis

$$\nabla \cdot (a \nabla B) = a \nabla \cdot (\nabla B) + \nabla a \cdot \nabla B$$

which gives us the new equation:

$$V \nabla \cdot (\nabla T) + \nabla T \cdot \nabla V + \frac{I}{c\rho} = 0$$

Or using a finite difference approximation

$$V_{i,j} \frac{(T_{i+1,j} - 2T_{i,j} - T_{i-1,j})}{(\Delta x)^2} + V_{i,j} \frac{(T_{i,j+1} - 2T_{i,j} - T_{i,j-1})}{(\Delta y)^2}$$
$$\frac{(V_{i+1,j} - V_{i-1,j})}{2(\Delta x)} \frac{(T_{i+1,j} - T_{i-1,j})}{2(\Delta x)} + \frac{(V_{i,j+1} - V_{i,j-1})}{2(\Delta y)} \frac{(T_{i,j+1} - T_{i,j-1})}{2(\Delta y)}$$
$$+ \frac{I_{i,j}}{c\rho} = 0$$

8.2 Stationary temperature field in a rectangular area

Consider a stationary temperature field in a rectangular area with heterogeneous thermal conductivity and a point source of heat and fixed temperature on the boundaries of the solution domain.

```
SET X /I1*I20/;
SET Y /J1*J20/;

* boundary location determination
SET bound(X,Y);
    bound(X,Y)         =yes;
    bound('I1','J10') =no;
    bound('I20','J10')=no;

SET inside(X,Y);
```

```
    inside(X,Y)                    =yes;
    inside(X,Y)$(ord(X)=1)         =no;
    inside(X,Y)$(ord(X)=card(X))   =no;
    inside(X,Y)$(ord(Y)=1)         =no;
    inside(X,Y)$(ord(Y)=card(Y))   =no;

* temperature supply determination
PARAMETER SUPPLY(X,Y);
          SUPPLY(X,Y)           := 0;
          SUPPLY('I17','J17')   := 10000;

* parameters determination
SCALAR dx /0.1/;
SCALAR dy /0.1/;

TABLE V(X,Y)
```

	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11	J12	J13	J14	J15	J16	J17	J18	J19	J20
I1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I3	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I4	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I6	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I7	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I8	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I9	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I10	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I11	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I12	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.5
I13	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.5
I14	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.5	0.5
I15	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.5
I16	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	1.0	1.0	0.5	0.5	0.5
I17	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	1.0	1.0	0.5	0.5	0.5
I18	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I19	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
I20	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5

```
*      MODEL  DESCRIPTION
VARIABLES
       obj
       T(X,Y);

******* Variables limits *********
T.lo(X,Y) =-100.0;
T.up(X,Y) = 200;
T.l(X,Y)  = 0;

******* Boundary Conditions ******
T.fx(X,Y)$(ord(X)=1)        = 0.0;
T.fx(X,Y)$(ord(X)=card(X)) = 0.0;
T.fx(X,Y)$(ord(Y)=1)        = 0.0;
T.fx(X,Y)$(ord(Y)=card(Y)) = 0.0;

T.fx('I1','J10')  = 100;
T.fx('I1','J11')  = 100;
T.fx('I10','J1')  = 100;
T.fx('I11','J1')  = 100;

EQUATION
       TEMP(X,Y)
       ben;

TEMP(X,Y)$(inside(X,Y)).. supply(X,Y)
        +  V(X,Y)*( T(X-1,Y)-2*T(X,Y)+T(X+1,Y))/dx/dx
        +  V(X,Y)*( T(X,Y-1)-2*T(X,Y)+T(X,Y+1))/dy/dy
        + (V(X,Y+1)-V(X,Y-1))* (T(X,Y+1)-T(X,Y-1))/dy/dy/2.0
        + (V(X+1,Y)-V(X-1,Y))* (T(X+1,Y)-T(X-1,Y))/dx/dx/2.0
        =e= 0;
```

```
ben.. obj =e= sum(X,T(X,'J2'));

*********************************
MODEL temper /all/;
SOLVE temper using nlp minimazing obj;

************************************************************
file res1 /heat_2.txt/
file res2 /heat_2.dat/
put res1;
put obj.l; put /;
loop(X,  put X.tl:6;loop(Y, put T.l(X,Y):6:2; );put /;);put /;
          put " inside 20  " /;
 loop(X,  put X.tl:6;loop(Y, put inside(X,Y):6; );put /;);
          put " inside 20  " /;
 loop(X,  put X.tl:6;loop(Y, put supply(X,Y):6; );put /;);
put res2;
loop(X,loop(Y, put T.l(X,Y):4:0; );put /;);put /;
```

The solution of the model is:

```
   0   0   0   0   0   0   0   0   0 100 100   0   0   0   0   0   0   0   0   0
   0   1   1   2   3   4   6  11  21  51  51  21  11   6   4   3   2   1   1   0
   0   1   3   4   6   8  11  15  22  32  32  22  15  10   7   5   4   2   1   0
   0   2   4   6   8  11  13  17  21  23  23  21  17  13  10   8   5   4   2   0
   0   3   6   8  11  13  15  18  21  22  22  21  18  15  12   9   7   5   2   0
   0   4   8  11  13  15  17  19  21  22  22  21  19  16  14  11   8   5   3   0
   0   6  11  13  15  17  19  20  22  22  22  22  20  18  15  12  10   6   3   0
   0  11  15  17  18  19  20  21  22  22  23  23  22  20  17  14  11   8   4   0
   0  21  22  21  21  21  22  22  23  23  23  23  23  22  19  16  13   9   5   0
 100  51  32  23  22  22  22  22  23  23  24  24  24  24  22  19  15  11   5   0
 100  51  32  23  22  22  22  23  23  24  25  25  26  27  25  22  18  13   7   0
   0  21  22  21  21  21  22  23  23  24  25  26  28  29  29  27  22  16   8   0
   0  11  15  17  18  19  20  22  23  24  26  28  29  31  33  33  28  21  11   0
   0   6  10  13  15  16  18  20  22  24  27  29  31  33  36  37  37  27  14   0
   0   4   7  10  12  14  15  17  19  22  25  29  33  36  39  42  44  37  18   0
   0   3   5   8   9  11  12  14  16  19  22  27  33  37  42  48  54  45  21   0
   0   2   4   5   7   8  10  11  13  15  18  22  28  37  44  54  78  52  22   0
   0   1   2   4   5   5   6   8   9  11  13  16  21  27  37  45  52  34  16   0
   0   1   1   2   2   3   3   4   5   5   7   8  11  14  18  21  22  16   8   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

The symmetry of the problem and the boundary conditions results in the symmetric solution. The solution of the model is written to the file "*.dat", which serves as an input file for the program which graphically displays the solution (see Figure 8.2.1). In this figure all isolines of temperature are symmetric, a necessary condition of the solution.

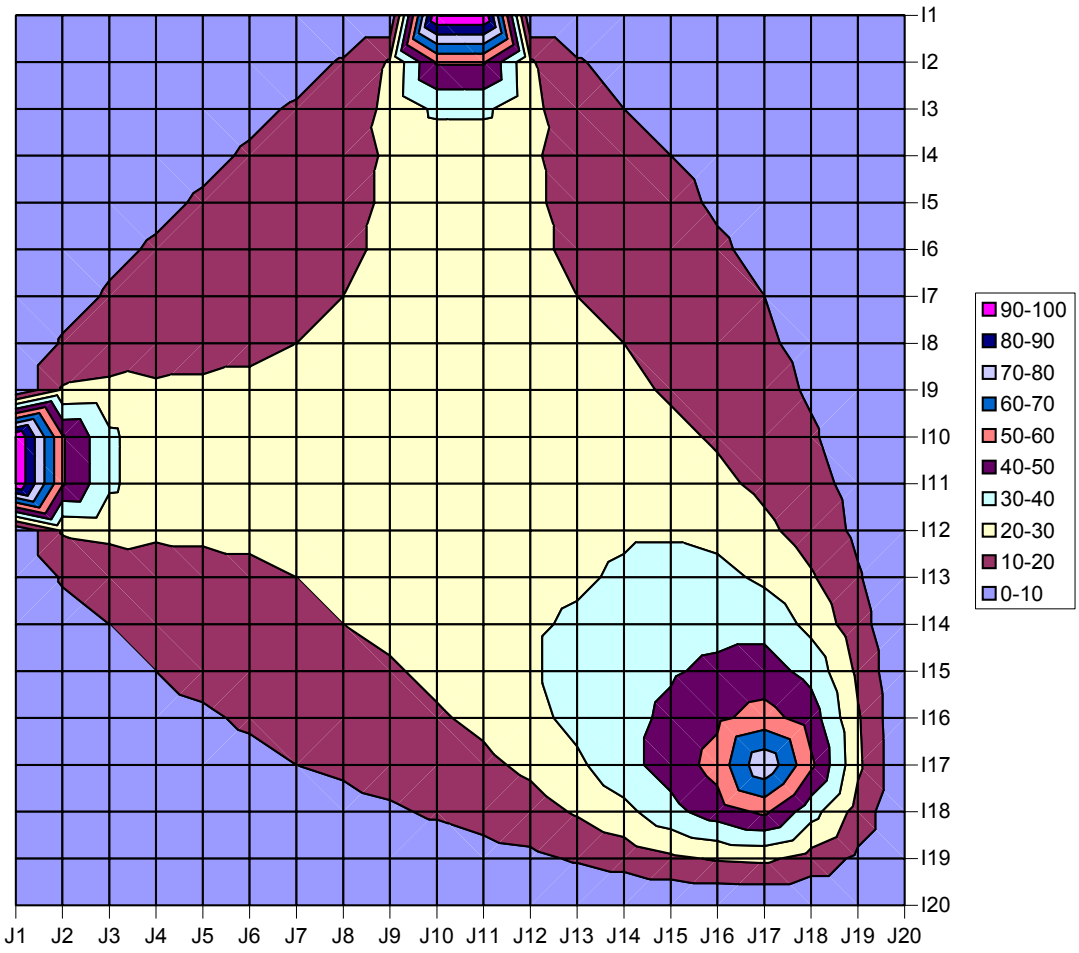


Figure 8.2.1. Solution to heat transport problem with fixed boundary conditions.

Next, we consider the possibility of finding a solution for other types of boundary conditions. Consider a problem where heat flow, not temperature, is given on the borders of the solution area, e.g., if the borders are made from insulating material. In this case

$$\frac{\partial T(x,y)}{\partial n} = 0 \quad on\,the\,boundary$$

where n is the unit normal vector to border of the solution area. The governing equation is

$$\nabla \cdot (V \nabla T) + \frac{I}{c\rho} = 0$$

Several difficulties present themselves immediately in this case:

1) The previous algebraic analogue of the differential equation is not conservative;

2) The existence of insulating boundaries contradicts the heat conductivity equation; and
3) The internal source of heat should balance the flow of heat through the borders.

That is, a stationary solution can not exist within insulated borders around an area containing an internal source of heat. The temperature in the region should constantly grow and its stabilization will not occur. To solve this problem it is necessary:

1) To reconstruct algebraic analog of the differential operators;
2) To alter the objective function; and
3) To make a boundary condition which is more realistic.

The boundary condition is modified as

$$\frac{\partial T(x,y)}{\partial n} = s \quad on\,the\,boundary$$

where s is a given heat flux on the boundary.

Assume that outside the solution domain the environment has large thermal conductivity. As a result of this, at any point on the border there will be a larger temperature than that of the nearby external points, this will result in heat currents in the external area moving parallel to the borders. From the external area there will be sufficient hear capacity to maintain the temperature on the border.

Let's copy the algebraic approximation of the differential operators in the conservative form. The algebraic approximation of the conservative form allows no opportunity for sources or sinks of heat resulting from the apoximation of the differential operators. Thus, considering the governing equation

$$\nabla \cdot (V \nabla T) + \frac{I}{c\rho} = 0$$

or, in its algrbraic representation

$$\frac{(V_{i+1,j} + V_{i,j})(T_{i+1,j} - T_{i,j}) - (V_{i,j} + V_{i-1,j})(T_{i,j} - T_{i-1,j})}{(\Delta x)^2} +$$
$$\frac{(V_{i,j+1} + V_{i,j})(T_{i,j+1} - T_{i,j}) - (V_{i,j-1} + V_{i,j})(T_{i,j} - T_{i,j-1})}{(\Delta y)^2}$$
$$+ \frac{I_{i,j}}{c\rho} = 0$$

The given temperature should be minimized during the search for the solution

$$Minimize\ T_0(x,y)$$

To guarantee conservation of energy in the solution it is necessary to calculate heat flows through the borders of the solution domain and to compare them with the source of heat inside the domain. Conservative methods for solving the differential equation should conserve the source of heat inside the solution domain by heat flow through the borders of the area. In the model, inequalities enter the solution when the flow of heat is directed from the solution domain to the external environment.

8.3 Stationary Temperature Field in a Rectangular Area with Border Heat Flow

Consider a ttationary temperature field in a rectangular area with heterogeneous thermal conductivity and a point source of heat and heat flows through the borders of the solution domain.

```
SET X /I1*I20/;

ALIAS (X,Y);
*****************************************************
*   Determine boundary location
*
*   Determine type of boundary conditions

SET bound(X,Y);
    bound(X,Y)        =yes;
    bound('I1','I10')=no;
    bound('I1','I11')=no;
    bound('I10','I1')=no;
    bound('I11','I1')=no;

*   Determine zone for heat transport equation
SET inside(X,Y);
 inside(X,Y)                    =yes;
 inside(X,Y)$(ord(X)=1)         =no;
 inside(X,Y)$(ord(X)=card(X))   =no;
 inside(X,Y)$(ord(Y)=1)         =no;
 inside(X,Y)$(ord(Y)=card(Y))   =no;

*   Determine parameters
SCALAR dx     step  in X direction  /0.1/
       dy     step  in Y direction  /0.1/ ;

*   Determine heat supply
PARAMETER SUPPLY(X,Y);
          SUPPLY(X,Y)         :=0;
          SUPPLY('I17','I17'):=100/dx/dy;

TABLE V(X,Y)     thermal conductivity distribution
    I1   I2   I3   I4   I5   I6   I7   I8   I9   I10  I11  I12 I13  I14  I15  I16  I17  I18  I19  I20
I1  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I2  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I3  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I4  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I6  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I7  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I8  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I9  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   1.0  1.0  1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I10 0.5  0.5  0.5  0.5  1.0  1.0  1.0  1.0  1.0   1.0  1.0  1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I11 0.5  0.5  0.5  0.5  1.0  1.0  1.0  1.0  1.0   1.0  1.0  1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I12 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  1.0   1.0  1.0  1.0  1.0  1.0  0.5  0.5  0.5  0.5  0.5  0.5
I13 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  1.0   1.0  1.0  1.0  1.0  1.0  1.0  1.0  0.5  0.5  0.5  0.5
```

```
I14 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  1.0  1.0  1.0  1.0  1.0  0.5  0.5  0.5  0.5
I15 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  1.0  1.0  1.0  1.0  1.0  0.5  0.5  0.5
I16 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  1.0  1.0  1.0  1.0  1.0  0.5  0.5  0.5
I17 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  1.0  1.0  1.0  0.5  0.5  0.5
I18 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I19 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I20 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5 ;

*********************************
*       MODEL  DESCRIPTION
*********************************

VARIABLES
          obj
          Q
          T(X,Y);

******* limitation *********
T.lo(X,Y)  =-100.0;
T.up(X,Y)  = 200;
T.l(X,Y)  = 0;
Q.lo  = 0;

******* boundary conditions ******
T.fx('I1','I10')  = 100;
T.fx('I1','I11')  = 100;
T.fx('I10','I1')  = 100;
T.fx('I11','I1')  = 100;

EQUATION
          TEMP(X,Y)  main equation

          f1(X,Y)    boundary condition dt:dn
          f2(X,Y)    boundary condition dt:dn
          f3(X,Y)    boundary condition dt:dn
          f4(X,Y)    boundary condition dt:dn

          fp1(X,Y)   boundary condition t
          fp2(X,Y)   boundary condition t
          fp3(X,Y)   boundary condition t
          fp4(X,Y)   boundary condition t

          ben        benefit equation
          ;

***************main equation***********************************
TEMP(X,Y)$(inside(X,Y))..

             (( V(X+1,Y)+V(X,Y))*(T(X+1,Y)-T(X,Y))
            -(  V(X,Y)+V(X-1,Y))*(T(X,Y)-T(X-1,Y)))/dx/dx/2.0
            +(( V(X,Y+1)+V(X,Y))*(T(X,Y+1)-T(X,Y))
            -(  V(X,Y-1)+V(X,Y))*(T(X,Y)-T(X,Y-1)))/dy/dy/2.0
            +   supply(X,Y)
            =e= 0.0
             ;

**************equation which determines dT/dn********
f1(X,Y)$((ord(X)=1      $bound(X,Y))).. T(X+1,Y)-T(X,Y)   =G= 0;
f2(X,Y)$((ord(X)=card(X)$bound(X,Y))).. T(X,Y)  -T(X-1,Y) =L= 0;
f3(X,Y)$((ord(Y)=1      $bound(X,Y))).. T(X,Y+1)-T(X,Y)   =G= 0;
f4(X,Y)$((ord(Y)=card(Y)$bound(X,Y))).. T(X,Y)  -T(X,Y-1) =L= 0;

**************equation which determines T on outside bndy******
fp1(X,Y)$((ord(X)=1      $bound(X,Y))).. T(X,Y) =e= Q;
```

```
fp2(X,Y)$((ord(X)=card(X)$bound(X,Y))).. T(X,Y) =e= Q;
fp3(X,Y)$((ord(Y)=1      $bound(X,Y))).. T(X,Y) =e= Q;
fp4(X,Y)$((ord(Y)=card(Y)$bound(X,Y))).. T(X,Y) =e= Q;

*****************************************************************
ben.. obj=e=Q;

*********************************
MODEL heat4 /all/;
SOLVE heat4 using nlp minimizing obj;

*****************************************************************
FILE res1 /heat_4.txt/
FILE res2 /heat_4.dat/

*****************************************************************
*     You can see the flow through the boundaries
*     from equations in the model

PARAMETER
         x_left(X), x_right(X),
         y_top(Y),  y_bottom(Y),
         total,
         total_x1, total_x2,
         total_y1, total_y2,
         t_supply;

x_left(X)   =  (T.l('I2',X)  -T.l('I1',X) )*(V('I1',X) +V('I2',X) )/dx/2;
x_right(X)  = -(T.l('I20',X) -T.l('I19',X))*(V('I20',X)+V('I19',X))/dx/2;
y_top(Y)    =  (T.l(Y,'I2')  -T.l(Y,'I1') )*(V(Y,'I1') +V(Y,'I2') )/dy/2;
y_bottom(Y) = -(T.l(Y,'I20') -T.l(Y,'I19'))*(V(Y,'I20')+V(Y,'I19'))/dy/2;

total_x1 = sum(X,x_left(X))*dy;
total_x2 = sum(X,x_right(X))*dy;
total_y1 = sum(Y,y_top(Y))*dx;
total_y2 = sum(Y,y_bottom(Y))*dx;
total    = sum(Y,y_top(Y))*dx  + sum(Y,y_bottom(Y))*dx
          +sum(X,x_left(X))*dy + sum(X,x_right(X))*dy;
t_supply = supply('I17','I17')*dx*dy;
```

The results are plotted in the following figure.

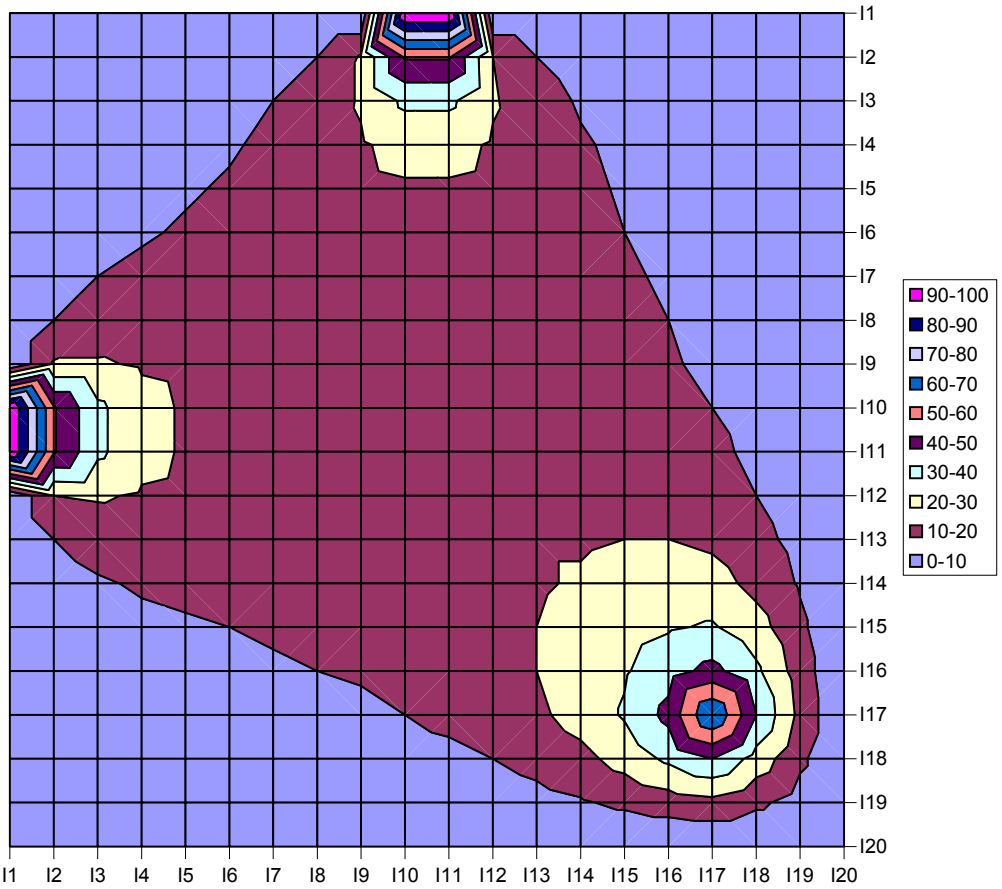


Figure 8.3.1. Solution of the heat transport problem minimizing the boundary temperature.

Part of the results file are

```
obj function                          =          0.00
 temperature on boundary              =          0.00
 inside heat supply                   =        100.00
 total heat transport                 =        100.00
 total heat transport x1 boundary     =         -2.98
 total heat transport x2 boundary     =         52.98
 total heat transport y1 boundary     =         -2.98
 total heat transport y2 boundary     =         52.98
heat transport  left   direction
  0.0   3.3   6.6  10.0  14.1  19.9  29.9  50.9 102.9 -246.4 -246.6
102.2  49.8  28.4  18.0  12.0   8.1   5.0   2.4   0.0
heat transport  top    direction
  0.0   3.3   6.6  10.0  14.1  19.9  29.9  50.9 102.9 -246.4 -246.6
102.2  49.8  28.4  18.0  12.0   8.1   5.0   2.4   0.0
heat transport  right  direction
  0.0   2.4   4.7   6.6   8.3   9.8  11.2  12.9  14.9  17.6  21.3
 26.7  34.5  45.4  60.3  76.2  84.6  61.6  30.8   0.0
 heat transport  botton direction
  0.0   2.4   4.7   6.6   8.3   9.8  11.2  12.9  14.9  17.6  21.3
 26.7  34.5  45.4  60.3  76.2  84.6  61.6  30.8   0.0
```

The symmetry of the results, is due to the invariancy of the equation, its algebraic analogue, the

symmetry of the solution domain, the field of conductivity and the symmetry of the boundary conditions. The equality of the internal source of heat and the total flow of heat through borders of the area demonstrate the conservatism of the method.

If we change the problem from one of minimization to one of maximization, we will receive a new solution strongly distinguished from the previous one. Part of that solution file is reproduced below and the temperature solution is shown in Figure 8.3.2.

In this case the temperature of external environment has appeared equal 103.38. But in this case, through all borders the thermal flow is directed to the outside. In the previous case on two sides we saw two powerful concentrated flows of heat directed into the area and compensatory currents of heat on other sides outside. In both cases, the flow of heat is balanced with the internal source of heat.

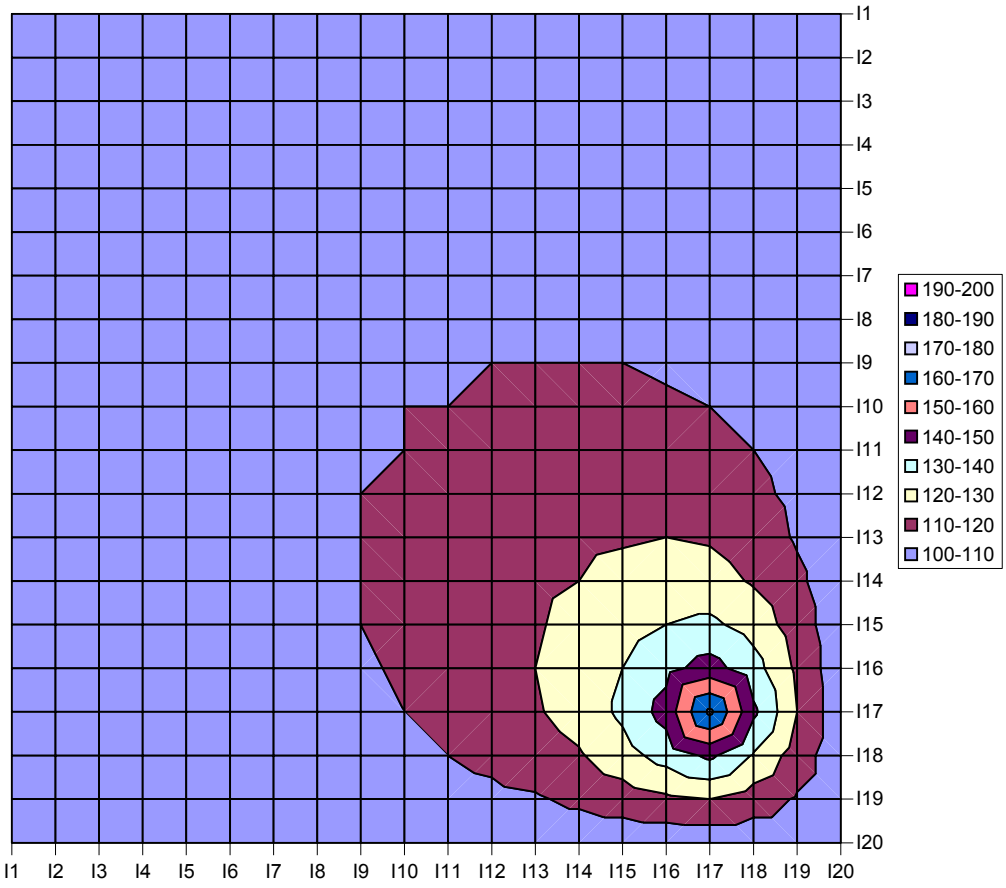


Figure 8.3.2. Solution of the heat transport problem maximizing the boundary temperature.

```
obj function                        =       103.38
 temperature on boundary            =       103.38
 inside heat supply                 =       100.00
 total heat transport               =       100.00
 total heat transport x1 boundary   =         4.53
 total heat transport x2 boundary   =        45.47
 total heat transport y1 boundary   =         4.53
```

```
total heat transport y2 boundary  =        45.47
heat transport left direction
 0.0   0.3   0.7   1.0   1.3   1.5   1.6   1.4   0.0
12.1  12.3   0.4   2.0   2.4   2.4   2.2   1.8   1.2   0.6   0.0
heat transport top direction
 0.0   0.3   0.7   1.0   1.3   1.5   1.6   1.4   0.0
12.1  12.3   0.4   2.0   2.4   2.4   2.2   1.8   1.2   0.6   0.0
heat transport right direction
 0.0   0.6   1.3   2.1   2.9   4.0   5.3   7.0   9.1
12.1  16.2  21.9  29.9  41.3  56.4  72.8  82.0  59.9  29.9   0.0
heat transport botton direction
 0.0   0.6   1.3   2.1   2.9   4.0   5.3   7.0   9.1  12.1
16.2  21.9  29.9  41.3  56.4  72.8  82.0  59.9  29.9   0.0
```

8.4 Time Dependent Temperature Field

Consider a time dependent temperature field in a rectangular area with heterogeneous thermal conductivity and a point source of heat and heat flows through the borders of the solution domain.

In GAMS we can search for the solution in a cyclic, or time marching, process, in which each sulution is used as the beginning of the search for the following solution and strongly influences it. For this purpose the **LOOP** operator can be used. Inside the body of the **LOOP** operator the **SOLVE** command is executed repreatedly until a maximum number of cycles ahs been completed. In the following example, this method is applied to the problem of finding time dependent temperature fields.

Let a point source of heat begins to work in the solution domain at the initial time. Heat from the source is distributed over the solution domain according to the transient heat transport equation

$$\nabla \cdot (V \nabla T) + \frac{I}{c\rho} = \frac{\partial T}{\partial t}$$

The solution method consists of calculating changes in the temperature field inside the solution domain.

We shall compute the termperature solution for only one time step; however, it will be computed on the basis of the previous temperature field. After the current period temperature field is computed from the previous time period it is saved for use in the next time step.

```
SET  X             /I1*I20 /;
SET  time          /t1*t1 /;
Alias(X,Y);

* Determination of solution domain

SET      Inside(X,Y);

         Inside(X,Y)                    = yes;
         Inside(X,Y)$(ord(X) = 1)       = no;
```

```
          Inside(X,Y)$(ord(X) = card(X)) = no;
          Inside(X,Y)$(ord(Y) = 1)       = no;
          Inside(X,Y)$(ord(Y) = card(Y)) = no;

* Parameter determination
Scalar heat accumulation in one time step   /0.0/;
Scalar dx step for space in X directions    /0.1/;
Scalar dy step for space in Y directions    /0.1/;
Scalar dt step for time                     /0.1/;

* Temperature supply determination
Parameter past_T(X,Y);
          past_T(X,Y)        = 0;
Parameter SUPPLY(X,Y);
          SUPPLY(X,Y)        = 0;
          SUPPLY('I10','I11'):= 10/dx/dy;

* Thermal conductivity
Table V(X,Y) temperature conductivity distribution

     I1  I2  I3  I4  I5  I6  I7  I8  I9  I10  I11  I12  I13  I14  I15  I16  I17  I18  I19  I20
I1  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I2  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I3  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I4  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I5  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I6  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I7  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I8  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I9  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I10 0.5 0.5 0.5 1.5 1.5 1.5 1.5 1.5 1.5  1.5  1.5  1.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I11 0.5 0.5 0.5 1.5 1.5 1.5 1.5 1.5 1.5  1.5  1.5  1.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I12 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I13 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I14 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I15 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I16 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I17 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I18 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I19 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
I20 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5;

* MODEL DESCRIPTION
VARIABLES
          T(X,Y) field of temperature
          Q      Temperature on boundaries
          Obj    Objective function;

EQUATION
          TEMP(X,Y) Main equation of heat transport
          F1(X,Y)  Boundary computation dt:dn
          F2(X,Y)  Boundary computation dt:dn
          F3(X,Y)  Boundary computation dt:dn
          F4(X,Y)  Boundary computation dt:dn
          Fp1(X,Y) Boundary computation t
          Fp2(X,Y) Boundary computation t
          Fp3(X,Y) Boundary computation t
          Fp4(X,Y) Boundary computation t
          Ben      Benefit equation;

* Main equation
TEMP(X,Y)$(inside(X,Y))..
          T(X,Y)-Past_T(X,Y) =e=
          dt*(Supply(X,Y)
              +( (V(X+1,Y)+V(X,Y))  *(T(X+1,Y)-T(X,Y))
              -(V(X,Y)  +V(X-1,Y))*(T(X,Y)  -T(X-1,Y)))
               /dx/dx/2.0
              +( (V(X,Y+1)+V(X,Y))*(T(X,Y+1)-T(X,Y))
              -(V(X,Y-1)+V(X,Y))*(T(X,Y)  -T(X,Y-1)))
```

```
              /dy/dy/2.0);

* Equation which determines dT/dn
F1(X,Y)$((ord(X) = 1))..        T(X+1,Y)-T(X,Y) =g= 0;
F2(X,Y)$((ord(X) = card(X)))..  T(X,Y)-T(X-1,Y) =l= 0;
F3(X,Y)$((ord(Y) = 1))..        T(X,Y+1)-T(X,Y) =g= 0;
F4(X,Y)$((ord(Y) = card(Y)))..  T(X,Y)-T(X,Y-1) =l= 0;

* Equation which determines temperature on boundary
Fp1(X,Y)$((ord(X) = 1 ))..      T(X,Y) =e= Q;
Fp2(X,Y)$((ord(X) = card(X))).. T(X,Y) =e= Q;
Fp3(X,Y)$((ord(Y) = 1 ))..      T(X,Y) =e= Q;
Fp4(X,Y)$((ord(Y) = card(Y))).. T(X,Y) =e= Q;

* Objective function
Ben.. Obj =e= Q;

* Limit of temperature inside and on boundaries
T.lo(X,Y) =  0.0;
T.up(X,Y) = 1000;
Q.l       =  0.0;

*****************************************************************
MODEL flow /all /;

* Output file
File res1 /flow.txt/
Put res1;

* Below is the cycle of computation
*****************************************************************
LOOP (time,
      SOLVE flow using nlp minimizing obj;

      PUT " Time interval = "; PUT time.tl:20; PUT /;
      LOOP (X,PUT X.tl:6;
           LOOP (Y,PUT T.l(X,Y):6:1;);PUT /;
      ); PUT /;
      Heat = Heat + sum( (X,Y),(T.l(X,Y)-Past_T(X,Y)))*dx*dy;
      Past_T(X,Y) = T.l(X,Y);
     );
```

Let's consider more in detail the structure of the main **LOOP** over *time* in the model. Each pass through the loop executes the GAMS **SOLVER** defining a new temperature field for the moment *t*. The algorithm uses an implicit approach where all variables belonging to the current time period are unknown. Considering the model further, the temperature field *T*(*X,Y*) is determined based on the values from the previous time step *Past_T*(*X,Y*). In the loop the current period temperature field is transferred to the previous temperature field. Then, the change in heat quantity in the domain is computed for the current time step. The amount of heat leaving the solution domain plus the amount of heat accumulated in the domain must be equal to the amount generated by the heat source.

Below is a fragment of the output file containing an account of the balance of heat for one time step:

```
Obj function                    = 0.00
Temperature on boundary         = 0.00
Heat storage                    = 0.93
 Inside heat supply             = 1.00
```

```
Total heat transport              = 0.068
Total heat transport x1 boundary = 0.01696
Total heat transport x2 boundary = 0.01277
Total heat transport y1 boundary = 0.02165
Total heat transport y2 boundary = 0.01766
```

The resulting temperature field for the third time period is illustrated in Figure. 8.4.1.

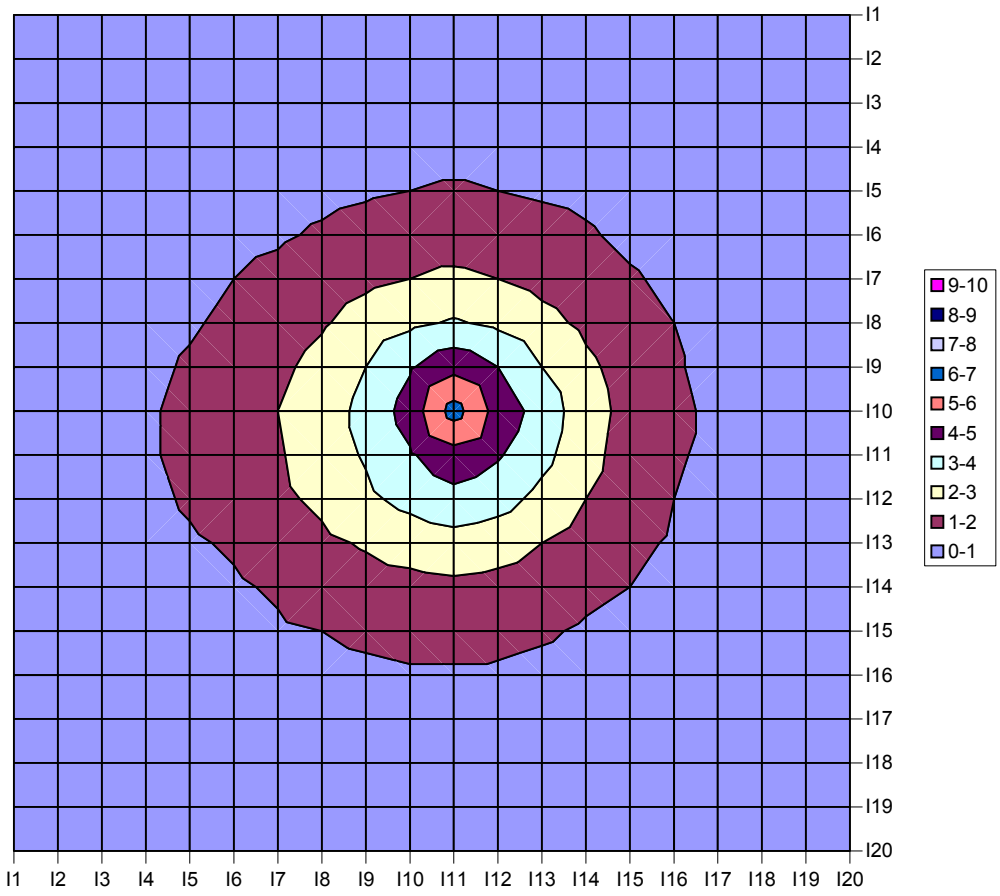


Figure 8.4.1. Solution of the transient heat transport problem after three time steps.

9. Optimal Solution of Fluid Flow Problems

9.1 Background

In this section, we consider solving some tasks of hydrodynamics. The governing equations for these examples are the conservation of linear momentum of a viscous, incompressible fluid in horizontal plane area

x-direction component

$$\frac{\partial V_x}{\partial t} + V_x \frac{\partial V_x}{\partial x} + V_y \frac{\partial V_x}{\partial y} = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \mu \nabla^2 V_x$$

y-direction component

$$\frac{\partial V_y}{\partial t} + V_x \frac{\partial V_y}{\partial x} + V_y \frac{\partial V_y}{\partial y} = -\frac{1}{\rho}\frac{\partial P}{\partial y} + \mu \nabla^2 V_y$$

In addition to these equations we need the equation of conservation of mass (continuity eqaution) for the fluid

$$\frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} = 0$$

where

V_x	*x* component of velocity [m/sec],
V_y	*y* component of velocity [m/sec],
P	Pressure [Pa],
μ	kinematic viscosity [m^2/sec],
ρ	fluid density [kg/m^3]

These equations apply to the case where the viscosity is constant and the fluid weight acts perpendicularly to the *x*-*y* plane, and, consequently, the gravitational force is absent from the equations. The liquid is considered incompressible, which complicates the search for the solution.

9.2 Stationary Flow of an Incompressible Fluid in a Rectangular Area

Consider a stationary flow of an incompressible fluid in a rectangular area with given inflow of fluid on the borders of solution domain.

Consider a rectangular area with zones of inflow and outflow of water on two opposite sides. In the zone of inflow of water, the boundary condition is:

$$V_x = 0.5 \ [m/\text{sec}] \text{ and } V_y = 0 \ [m/\text{sec}]$$

On the other sides of the rectangle, we have the so-called "no-slip" condition:

$$V_x = 0 \ [m/\text{sec}] \text{ and } V_y = 0 \ [m/\text{sec}]$$

In water outflow zone, the boundary condition is

$$\frac{\partial V_x}{\partial x} = 0 \text{ and } V_y = 0$$

The given boundary conditions ensure a parallel output of fluid from the solution domain. This situation is illustrated in Figure 9.2.1.

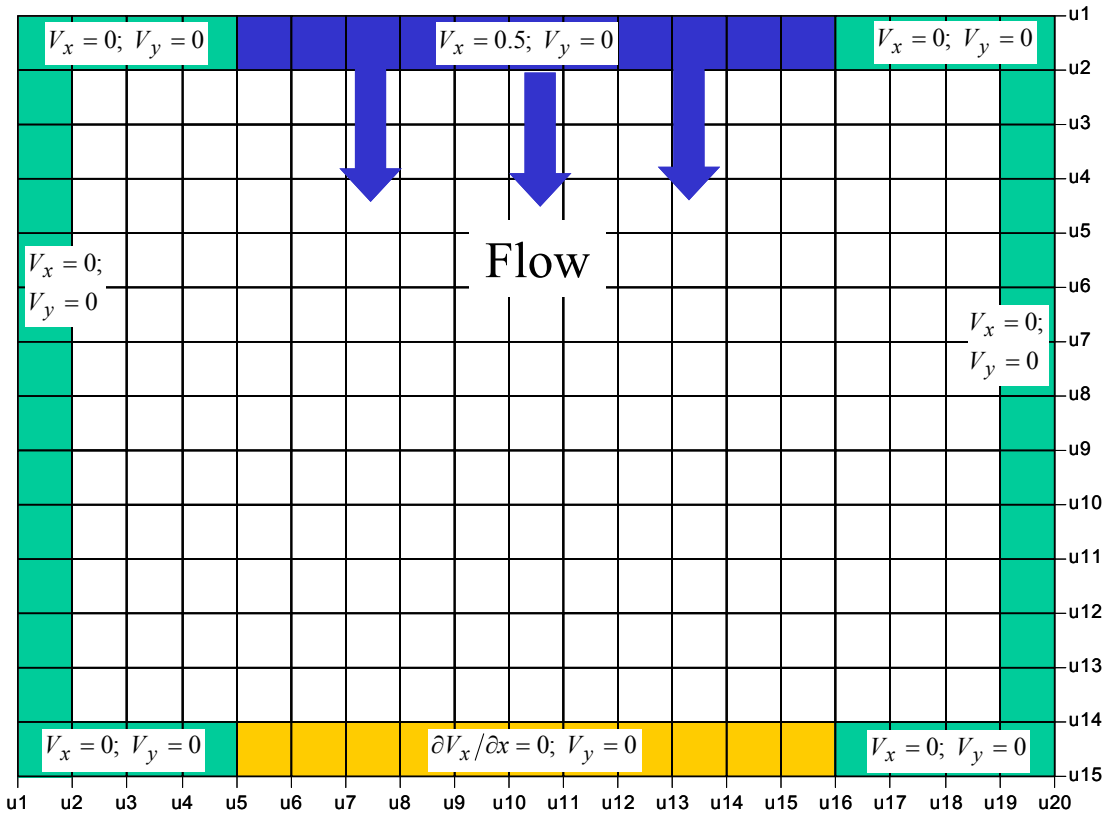


Figure 9.2.1. Flow domain for diffuser example.

Now, consider the finite difference analogs of the differential equations. We must divide the solution domain into small rectangles using a grid of parallel lines and give each intersection point appropriate indices (i,j). At each point we will compute fluid velocity. The third equation presents some difficulty, since it is not a time evolutionary form.

We compute velocity not at the points, but on the lines connecting the points. On lines parallel to the x axis we compute V_x, and on lines parallel to the y axis we calculate V_y.

Using this it is possible to make the following approximation for the continuity equation has the finite difference analog

$$\frac{V_{x,i,j} - V_{x,i-1,j}}{\Delta x} + \frac{V_{y,i,j} - V_{y,i,j-1}}{\Delta y} = 0$$

Pressure (P) is computed at the node points (i,j) of each line, where the pressure gradient is determined from the velocity. That is,

$$\left.\frac{\partial P}{\partial x}\right|_{i,j} \approx \frac{P_{i+1,j} - P_{i,j}}{\Delta x} \quad \text{and} \quad \left.\frac{\partial P}{\partial y}\right|_{i,j} \approx \frac{P_{i,j+1} - P_{i,j}}{\Delta y}$$

The diffusive terms are computed as

$$\mu\nabla^2 V_x \approx \mu\left(\frac{V_{x,i+1,j} - 2V_{x,i,j} + V_{x,i-1,j}}{(\Delta x)^2} + \frac{V_{x,i,j+1} - 2V_{x,i,j} + V_{x,i,j-1}}{(\Delta y)^2}\right)$$

$$\mu\nabla^2 V_y \approx \mu\left(\frac{V_{y,i+1,j} - 2V_{y,i,j} + V_{y,i-1,j}}{(\Delta x)^2} + \frac{V_{y,i,j+1} - 2V_{y,i,j} + V_{y,i,j-1}}{(\Delta y)^2}\right)$$

Details of the finite difference grid are illustrated in Figure 9.2.3.

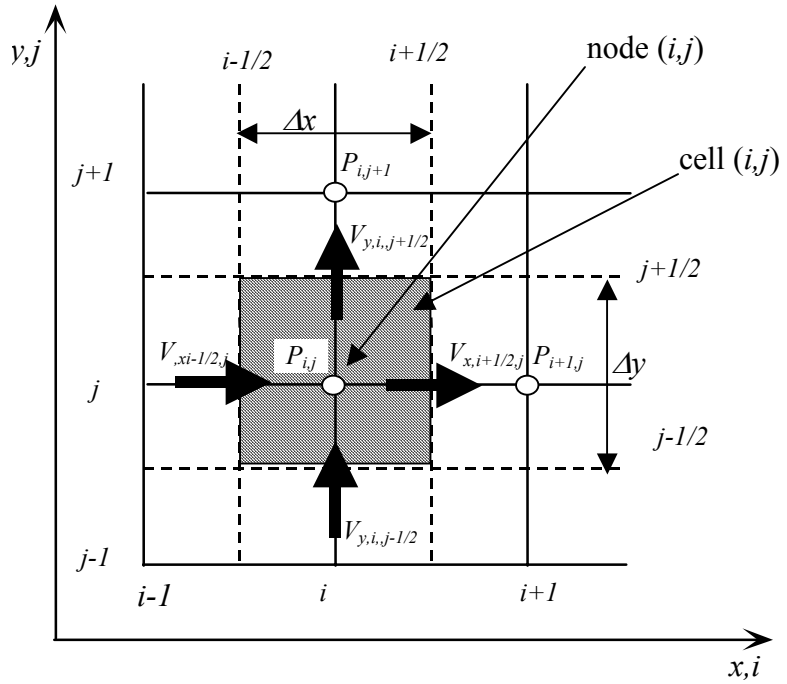


Figure 9.2.3. Finite-difference cell in two dimensional flow.

An attempt to solve the above system of the equations will be unsuccessful. It is not always possible to aproximate differentials by a finte difference, since there are errors introduced by the algebraic aproximations. Considering the initial system of equations, we notice that the first and second

equations contain velocity components which are connected by ∇P. If P is not connected with boundary conditions, all apoximation errors will leave the solution domain through the borders. In the third equation, the velocity vector components are in strong contact with the boundary conditions. That is, in the calculation of the continuity equation we expect errors at each point.

In connection with this difficulty, we write the equation in a slightly different (relaxed) form.

$$\frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} = \delta$$

where δ is the error in the apoximation which must be minimized in the solution.

The final system of equations to solve is:

$$-\frac{1}{\rho}\frac{\partial P}{\partial x} + \mu \nabla^2 V_x = 0$$

$$-\frac{1}{\rho}\frac{\partial P}{\partial y} + \mu \nabla^2 V_y = 0$$

$$\frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} = \delta$$

with finite difference analogs:

$$\frac{1}{\rho}\frac{P_{i+1,j} - P_{i,j}}{\Delta x} = \mu \frac{V_{x,i-1,j} - 2V_{x,i,j} + V_{x,i+1,j}}{(\Delta x)^2}$$

$$\frac{1}{\rho}\frac{P_{i+1,j} - P_{i,j}}{\Delta y} = \mu \frac{V_{y,i-1,j} - 2V_{y,i,j} + V_{y,i+1,j}}{(\Delta y)^2}$$

$$\frac{V_{x,i,j} - V_{x,i-1,j}}{\Delta x} + \frac{V_{y,i,j} - V_{y,i,j-1}}{\Delta y} = \delta_{i,j}$$

The GAMS model is:

```
SET   X /u1*u15 /;
SET   Y /u1*u20 /;
SET   Yout(Y) /u5*u16 /;

* Determination of zone for water movement equation
SET Inside(X,Y);
    Inside(X,Y)                    = yes;
    Inside(X,Y)$(ord(X) = 1)       = no;
```

```
    Inside(X,Y)$(ord(X) = card(X)) = no;
    Inside(X,Y)$(ord(Y) = 1)       = no;
    Inside(X,Y)$(ord(Y) = card(Y)) = no;

* Determination of parameters
Scalar     dx  step space in X directions m        /1/;
Scalar     dy  step space in Y directions m        /1/;
Scalar     r   density of fluid kg per m3       /1000/;
Parameter  m(X,Y)  kinematic viscosity of fluid m2 per sec;
           m(X,Y):= 0.005;

Variables
         Obj           Objective
         D(X,Y)        Error
         P(X,Y)        Pressure
         Vx(X,Y)       x-direction velocity
         Vy(X,Y)       y-direction velocity
         Vdx(Y)        delta Vx at the outlet;

*  Variable Bounds
         Vx.up(X,Y)   =  1.5;
         Vx.lo(X,Y)   = -1.5;
         Vx.l(X,Y)    =  0.5;
         Vy.up(X,Y)   =  1.0;
         Vy.lo(X,Y)   = -1.0;
         Vy.l(X,Y)    =  0.0;
         D.lo(X,Y)    =  0.0;
         D.up(X,Y)    =  0.0000005;
         Vy.l(X,Y)    =  0.0;
         Vy.l(X,Y)$(inside(X,Y)) = 0.0000001;
         P.up(X,Y)    =  2000;
         P.lo(X,Y)    = -2000;
         P.l(X,Y)     =  0.000001;

*  Imposed Boundary conditions
         Vx.lo('u1',Y)      = 0.5;
         Vx.fx('u15',Y)     = 0.5;
         Vx.fx(X,'u1')      = 0;
         Vx.fx(X,'u20')     = 0;
         Vx.fx('u1','u2')   = 0;
         Vx.fx('u1','u3')   = 0;
         Vx.fx('u1','u4')   = 0;
         Vx.fx('u1','u17')  = 0;
         Vx.fx('u1','u18')  = 0;
         Vx.fx('u1','u19')  = 0;
         Vx.fx('u15','u2')  = 0;
         Vx.fx('u15','u3')  = 0;
         Vx.fx('u15','u4')  = 0;
         Vx.fx('u15','u17') = 0;
         Vx.fx('u15','u18') = 0;
         Vx.fx('u15','u19') = 0;
         Vy.fx('u1',Y)      = 0;
         Vy.fx(X,'u1')      = 0;
         Vy.fx(X,'u20')     = 0;
         Vy.fx('u15',Y)     = 0;

Equation
         For_Vx(X,Y)   main equation 1
         For_Vy(X,Y)   main equation 2
         Div_Vxy(X,Y)  main equation 3 clear equation of balance
         Vx_Vx(Y)      outflow zone where determine outflow for X
```

```
         Ben           Objective function;

For_Vx(X,Y)$(Inside(X,Y))..
         (P(X+1,Y)-P(X,Y))/(r*dx) =e=
         m(X,Y)*((Vx(X+1,Y)-2*Vx(X,Y)+Vx(X-1,Y))/(dx*dx)
                +(Vx(X,Y+1)-2*Vx(X,Y)+Vx(X,Y-1))/(dy*dy));

For_Vy(X,Y)$(Inside(X,Y))..
         (P(X,Y+1)-p(X,Y))/(r*dy) =e=
         m(X,Y)*((Vy(X+1,Y)-2*Vy(X,Y)+Vy(X-1,Y))/(dx*dx)
                +(Vy(X,Y+1)-2*Vy(X,Y)+Vy(X,Y-1))/(dy*dy));

Div_Vxy(X,Y)$((ord(X) > 1)$(ord(Y) > 1))..
         (Vx(X-1,Y)-Vx(X,Y))/dx + (Vy(X,Y-1)-Vy(X,Y))/dy
               =e= D(X,Y);

Vx_Vx(Y).. Vdx(Y) =e= Vx('u15',Y)-Vx('u14',Y);

Ben..   Obj =e= SUM(Y$Yout(Y), Vdx(Y)*Vdx(Y))
               +SUM((X,Y),(D(X,Y)*D(X,Y)));

Model flow1 /ALL/;
Solve flow1 using nlp minimizing obj;
```

Figure 9.2.4 illustrates the velocity field in the solution domain. The points are the nodes of the finite difference grid and the tails point in the direction of flow away from the point. Figure 9.2.5 shows three cross-sections of x-direction velocity, V_x.

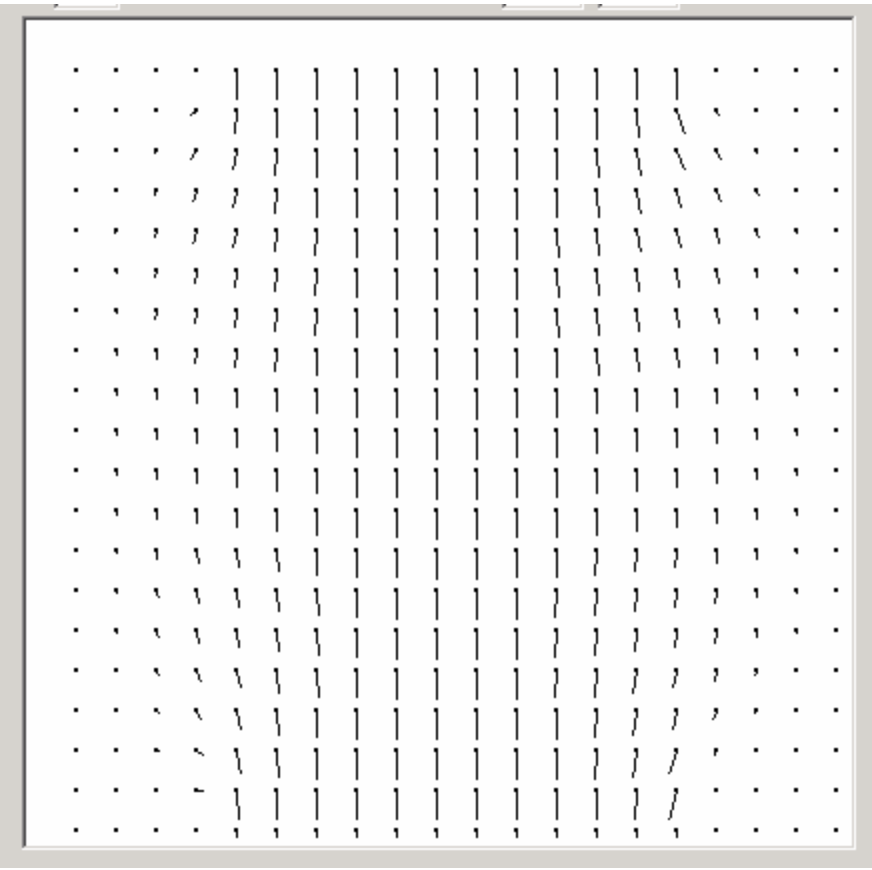


Figure 9.2.4. Flow field for flow from diffuser.

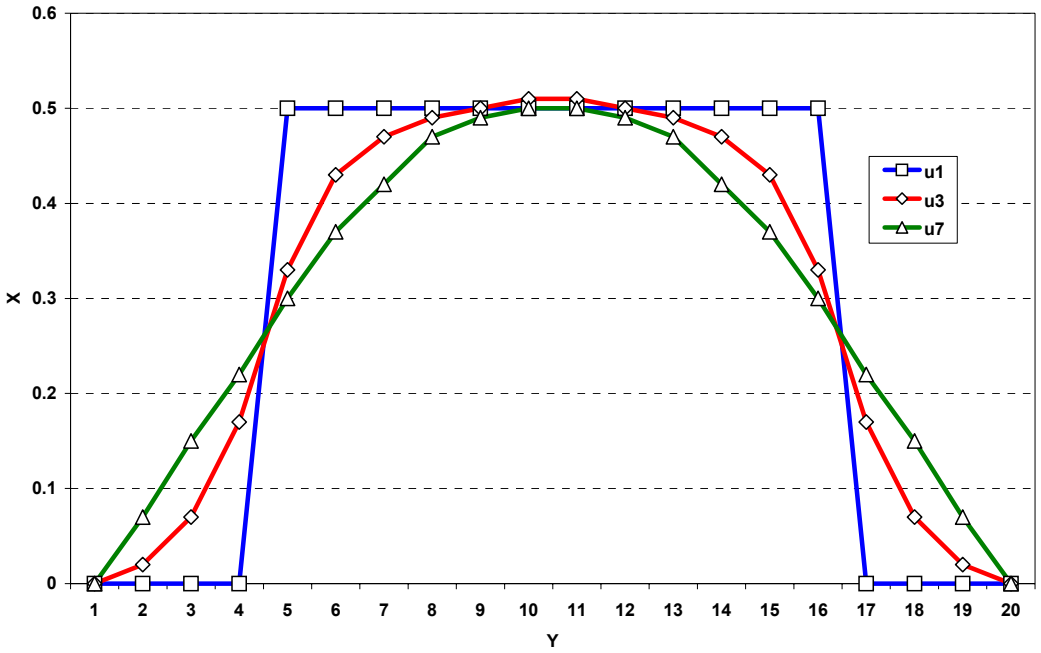


Figure 9.2.5. Three cross-sections of x-direction velocity, V_x.

This task has been solved using a numerical method and it can also be solved by potential thorey. Below we illustrate an example which can not be solved by analytical methods, however, the solution of the task by numerical methods is possible.

9.3 Stationary flow of water in a rectangular area in the presence of an obstacle

Consider a stationary flow of water in a rectangular area in the presence of an obstacle and complete account of hydrodynamic pressure and forces of inertia.

Let's consider a rectangular area. On the top edge of the area an inflow of water with a velocity of 0.5 m/sec occurs. The same flow leaves the area through the bottom edge of the region. The boundary conditions are the same as in the previous example. However, tn the middle of solution domain there is a symmetrically located obstacle, around which the fluid is compelled to flow. The equations governing flow in this case are the same as before and the only addition is the definition of the obstacle in the middle of the solution domain.

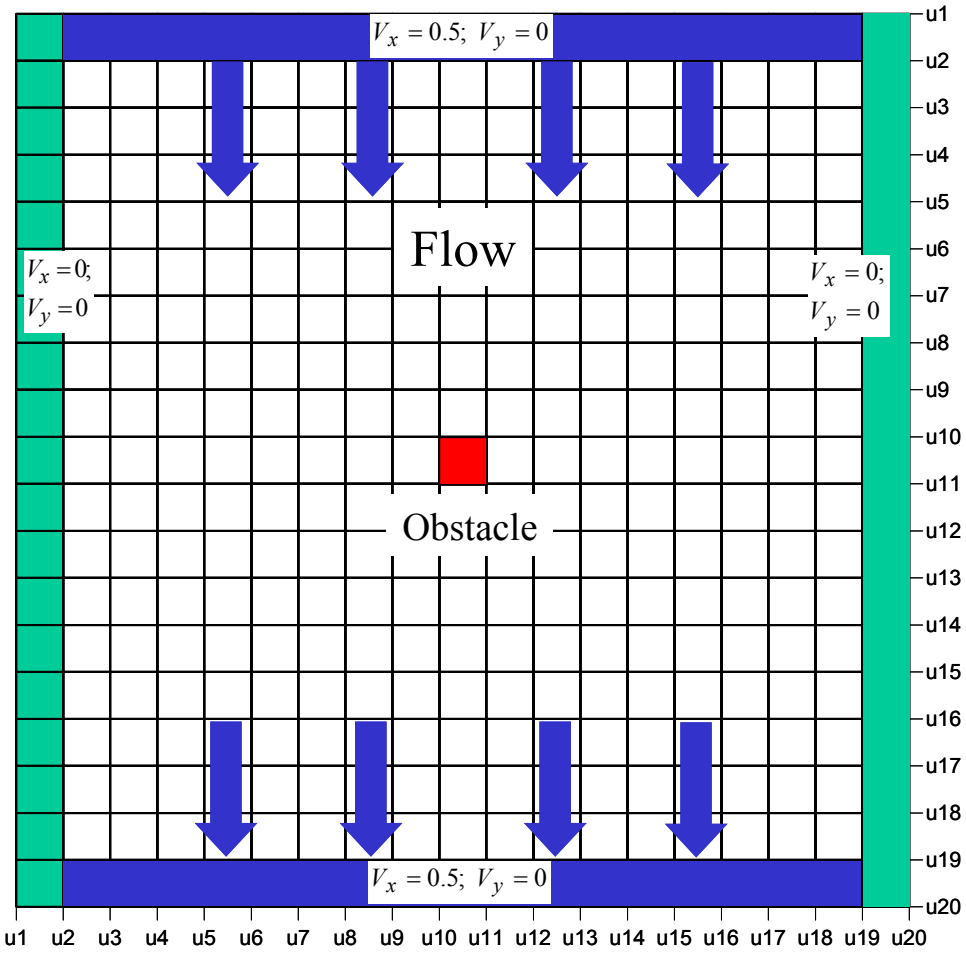


Figure 9.3.1. Flow domain for obstacle example.

The GAMS code for the model is:

```
SET X  /1*20/;
SET Y  /1*20/;

* Determination of zone for water movement
SET vyside(X,Y);
SET vxside(X,Y);
    Vxside(X,Y)                        = yes;
    Vxside(X,Y)$(ord(X) = 1)           = no;
    Vxside(X,Y)$(ord(X) = card(X))     = no;
    Vxside(X,Y)$(ord(Y) = 1)           = no;
    Vxside(X,Y)$(ord(Y) = card(Y))     = no;
    Vyside(X,Y) = Vxside(X,Y);
    Vxside('10','10')                  = no;
    Vxside('10','11')                  = no;

* Parameter determination
Scalar    dx  step space in X directions /1/;
Scalar    dy  step space in Y directions /1/;
Scalar    r   density of fluid kg per m3 /1000/;
Parameter m(X,Y) kinematic viscosity of fluid m2 per sec;
```

```
          m(X,Y):= 0.05;

* MODEL DESCRIPTION *
Variables
         Obj
         D(X,Y)        error
         P(X,Y)        Pressure
         Vx(X,Y)       x-direction velocity
         Vy(X,Y)       y-direction velocity
         ;

* Variable limits
D.lo(X,Y) = 0;
D.up(X,Y) = 7.50;
P.up(X,Y) = 20000;
P.lo(X,Y) =-20000;
P.l(X,Y) =  0.0;
Vx.l(X,Y) = 0.0;
Vy.l(X,Y) = 0.0;
Vy.l(X,Y)$(Vxside(X,Y)) = 0.0;

* Boundary conditions *
Vx.fx( '1',Y) = 0.5;
Vx.fx('20',Y) = 0.5;
Vx.fx(X, '1') = 0;
Vx.fx(X,'20') = 0;
Vy.fx( '1',Y) = 0;
Vy.fx('20',Y) = 0;
Vy.fx(X, '1') = 0;
Vy.fx(X,'20') = 0;

* Fence description *
Vx.fx('10','10') = 0;
Vx.fx('10','11') = 0;

Equation
         For_Vx(X,Y)    main equation 1
         For_Vy(X,Y)    main equation 2
         Div_Vxy(X,Y)   main equation 3 clear equation of balance
         Ben              benefit equation;

Div_Vxy(X,Y)$((ord(X) > 1)$(ord(Y) > 1))..

   (Vx(X,Y)-Vx(X-1,Y))/dx + (Vy(X,Y)-Vy(X,Y-1))/dy =e= D(X,Y);

For_Vx(X,Y)$(Vxside(X,Y))..
* Upwind scheme for inertial terms
*          Vx(X,Y)*(Vx(X+1,Y)-Vx(X-1,Y))/(2*dx)
*   +0.25*(Vy(X+1,Y-1)+Vy(X+1,Y)+Vy(X,Y-1)+Vy(X,Y))
*        *(Vx(X,Y+1)-Vx(X,Y-1))/(2*dy)
*   +(P(X+1,Y)-P(X,Y))/(r*dx)
*   =e=
*   m(X,Y)*((Vx(X+1,Y)-2*Vx(X,Y)+Vx(X-1,Y))/(dx*dx)
*          +(Vx(X,Y+1)-2*Vx(X,Y)+Vx(X,Y-1))/(dy*dy));

For_Vy(X,Y)$(Vyside(X,Y))..
* Upwind scheme for inertial terms
*    0.25*(Vx(X-1,Y+1)+Vx(X-1,Y)+Vx(X,Y+1)+Vx(X,Y))
*        *(Vy(X+1,Y)-Vy(X-1,Y))/(2*dy)
*         +Vy(X,Y)*(Vy(X,Y+1)-Vy(X,Y-1))/(2*dy)
*   +(P(X,Y+1)-P(X,Y))/(r*dy)
```

```
*    =e=
*    m(X,Y)*((Vy(X+1,Y)-2*Vy(X,Y)+Vy(X-1,Y))/(dx*dx)
*            +(Vy(X,Y+1)-2*Vy(X,Y)+Vy(X,Y-1))/(dy*dy));

Ben..
   Obj =e=  sum((X,Y),d(X,Y)*d(X,Y));

model flow2 /ALL/;
* Solve flow2 using nlp minimizing obj;
Option nlp = minos5;
*Option nlp = conopt2;
flow2.WORKSPACE=10;
Solve flow2 using nlp minimizing obj;
```

Figure 9.3.2 shows the flow field for this example. One can notice the flow of fluid around the obstacle and the high degree of recirculating flow behind the object. Figure 7 shows cross-sections of x-direction velocity before the obstacle, at the level of the obstacle and after the obstacle.

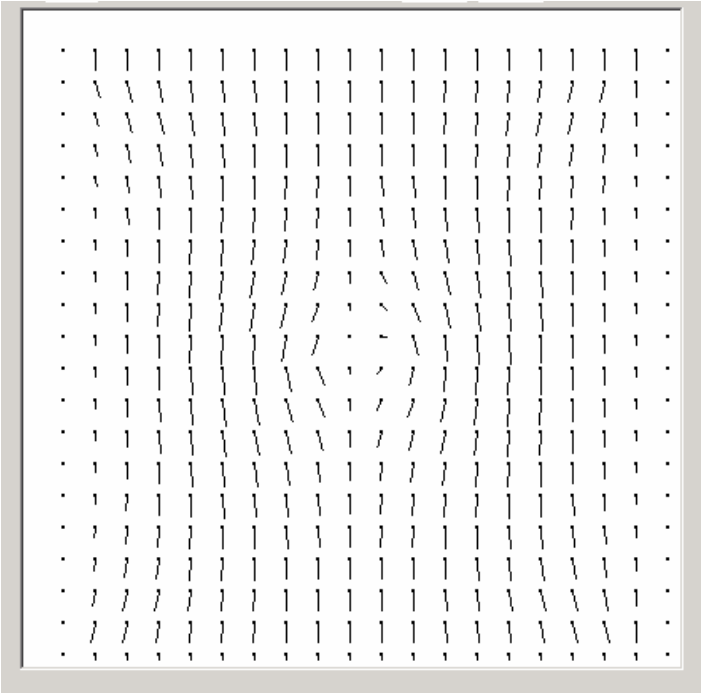


Figure 9.3.2. Flow field for fence example without initerial effects.

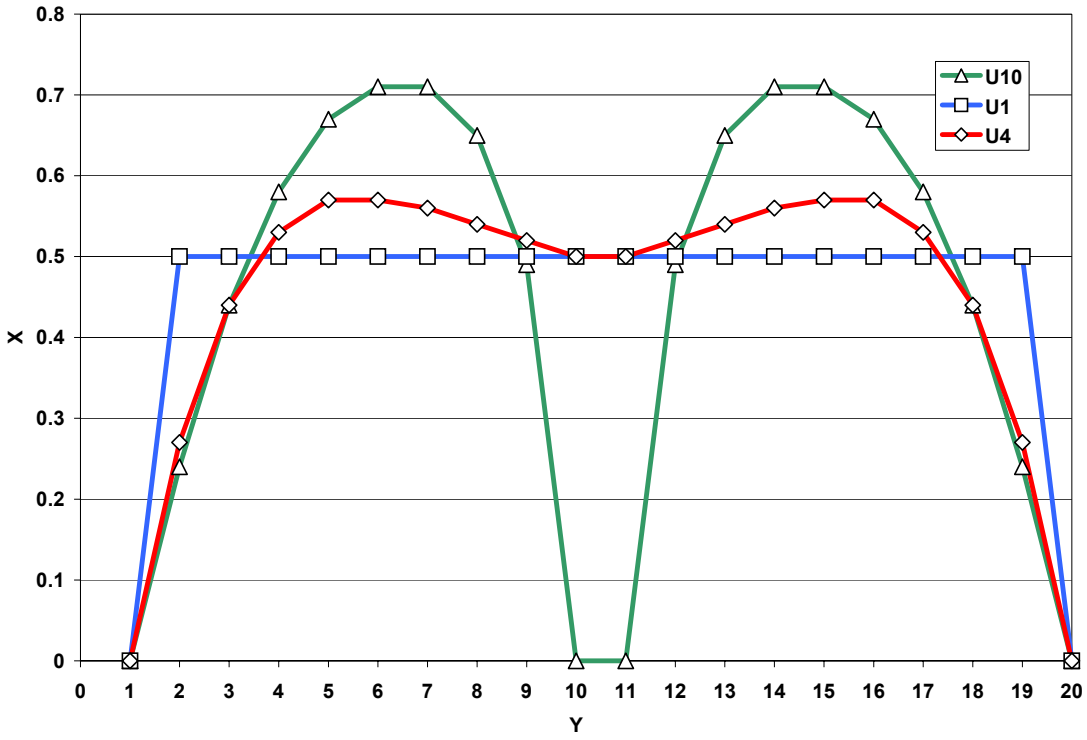


Figure 9.3.3. Three cross-sections of x-direction velocity, V_x.

The user will notice that the inertial terms of the equations are switched in the GAMS code. The user can include these terms in model by removing the asterisks from the code. The inertial terms are computed using the "marker and cell" method (*Peyret and Taylor*, 1985)

$$V_x \frac{\partial V_x}{\partial x} + V_y \frac{\partial V_x}{\partial y}$$

$$\approx V_{x,i,j} \frac{V_{x,i+1,j} - V_{x,i,j}}{2\Delta x} + \frac{V_{y,i+1,j} + V_{y,i,j} + V_{y,i,j-1} + V_{y,i+1,j-1}}{4} \frac{V_{x,i,j+1} - V_{x,i,j-1}}{2\Delta y}$$

$$V_x \frac{\partial V_y}{\partial x} + V_y \frac{\partial V_y}{\partial y}$$

$$\approx \frac{V_{x,i,j} + V_{x,i,j+1} + V_{x,i-1,j+1} + V_{x,i-1,j}}{4} \frac{V_{y,i+1,j} - V_{y,i-1,j}}{2\Delta x} + V_{y,i,j} \frac{V_{y,i,j+1} - V_{y,i,j-1}}{2\Delta y}$$

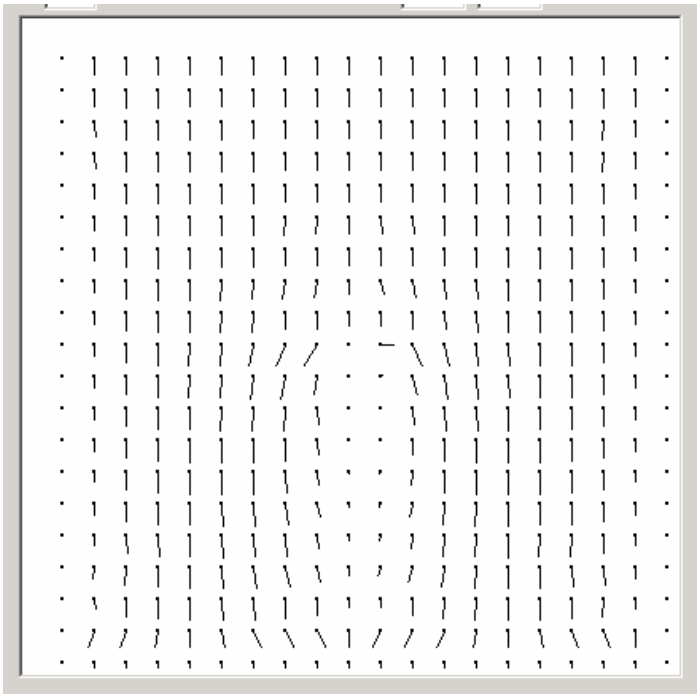


Figure 9.3.4. Flow field for fence example with initerial effects.

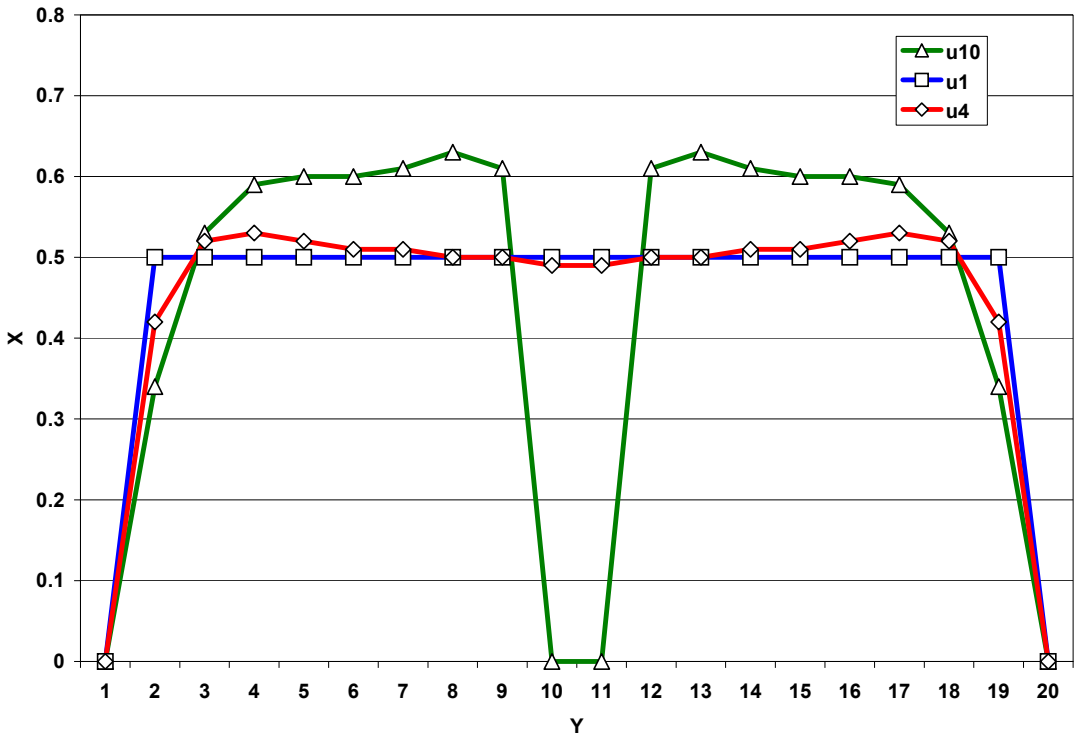


Figure 9.3.5. Three cross-sections of x-direction velocity, V_x, with intierial effects.

Comparing results of the models in which the inertial forces are taken into account indicates their

influence. In one case return flow of the fluid is present (see Figure 8) and in the other it is not present (see Figure 6). There can also be a question about the asymmetry of the aproximations for the inertial terms. However, the velocities are determined not at cell centers, but on cell faces, and for these are symmetric. On the basis of the above elementary examples it is possible to construct a model of transient movement of fluids using the approach described in the secition on heat transfer.

PART 2: THE GAMS LANGUAGE

10. INTRODUCTION

The General Algebraic Modeling System (GAMS) is designed to facilitate the work of modelers, who create mathematical models of real processes around us. Like other languages of this type, GAMS relieves modelers of the necessity to design algorithms to solve models, in addition to developing the models themselves. Thus, the major functional objective of GAMS is to design algorithms for mathematical models. However, it does not mean that this system can be used only for mathematical models with algebraic equations. Those who are acquainted with methods of solving partial differential equations (PDEs) know that numerical methods for solving PDEs always result in systems of algebraic equations which must be solved according to some iterative algorithm. Thus, the applicability of GAMS is wide. In this Tutorial, models and programs using the GAMS language are presented that allow users to develop their own models. This Tutorial should be supplemented by the official GAMS Guide and Tutorials (Brooke et al., 1997).

10.1 INSTALLATION

Detailed instructions for how to install GAMS on your computer are contained in Appendix A. Periodic updates to these instructions can be found on the GAMS website (www.gams.com).

Installation requires a system capable of working in the Windows or MS-DOS mode and its characteristics should not be less than those of a standard IBM 386 PC computer.

The user should be able to state a mathematical problem correctly. That is, formulation of all logical interrelations of the problem and features of its behavior, especially in extreme cases. GAMS cannot think for you. It can only help you to solve mathematical problems if they are stated correctly. If, for some reason, the model does not give correct results, generally, the mistake is not a GAMS mistake. Try to find the mistake in your own model formulation. If you do not manage to find the mistake at first, have a rest and try to find it again. Check your model for mistakes (the theoretical part plus formulation in the GAMS language). However, in some cases, and in the Tutorials there is such an example, GAMS can provide you with a decision which is not optimal. That is, you should be ready for many contingencies. Although in most cases, you will get what you seek.

You should know how to use a text editor in the Windows or MS-DOS mode on your computer. You can use any other text editor in any other system, but the editor you use should be able to create text files. The Windows Notepad editor, Norton Utilities editor and the PE text editor are very apt.

There are two sources on information GAMS and GAMS models:

1. Books and tutorials (the list is at the end of the Tutorials);
2. The internal library of the GAMS models supplied with the main GAMS package. It is in the subdirectory MODLIB. The library is created automatically as the package is installed on your computer. In the library, there are more than 150 models with necessary comments. These models are all operable. All models are text files, which can be printed by any editor.

11. INTRODUCTORY INFORMATION

11.1 Model Structure

Each model in the GAMS language may have its own distinctions. It can be comprised of groups of alternating models, which are linked together only through a small number of common variables or parameters. However, most models have a structure in the form shown in Table 11.1.1.

Table 11.1.1.
Structure of a GAMS Model.

1. SETS	Structures consisting of a complex of indices or names
	Declaration of fact of existence
	Determination of relations between individual structures
2. DATA	PARAMETERS, TABLES, SCALARS
	Declaration of fact of existence
	Determination of values of input parameters
	Preliminary computations
3. VARIABLES	Variables or arrays of variables
	Declaration with assigning a type of variable
	Declaration of limits for possible changes, initial level
4. EQUATIONS	Equations or complexes and arrays of equations
	Declaration with assigning a name
	Recording of equations in the GAMS language
5. MODEL, SOLVE	Model and methods of solution
6. OUTPUT	Output of information into a separate file

During model execution, information is displayed on the screen for any solution circumstances. The compilation results of the model and computations are automatically sent to a special "*LIST*" file. This file has the same name as the file of the reference model, but a different extension – "*LST*". Whatever happens, whether the computation is right or wrong, the file "**.lst*" automatically appears in the same directory as the file "**.gms*". Depending on the situation, the contents of this file may differ from that given below, but the basis remains.

1. Reprinting of the model
2. Map of interrelations of parameters and variables
3. Information on equations
4. Information on the solution status
5. Information on the obtained solution

Before describing basic rules of the GAMS language, we will give some brief information about features of the language. No equation, variable or their interrelationship can be used until they have been declared in the zone of STATEMENTS (items 2 and 3 in Table 11.1.1). These statements can be formed in quite an arbitrary style. We will use this arbitrariness in the examples below. The GAMS language does not distinguish between capital and lowercase letters of the alphabet. In this case, it is useful to observe the GAMS rules. Capitals are used for function words, small letters are used to form the model's own meaning. GAMS accepts comments using only the English alphabet (in the Russian version of this document, comments in Russian appear in italics but should not appear in the model for execution).

11.2 Comments in Models

There are three main methods of introducing comments and elucidation's in the body of a GAMS model.

1. If a line begins with the " * "symbol, the contents of the line are treated as a comment. For example,

```
*plants (plants)
```

2. A program section beginning with the line $ONTEXT and ending with the line $OFFTEXT is a comment. The above functional words begin with the first symbol of the line.

```
$ONTEXT;
-  -  -  -  -  -  -  -  -  -
This guide is for a scientist who knows the Russian language.
+  +  +  +  +  +  +  +  +  +
$OFFTEXT;
```

The user can introduce comments in the bodies of GAMS declarations. In the Russian language version of this document, comments typed in a smaller font remind the user about the necessity to change them in the working program for comments in English or any other language using the Latin alphabet. For instance, the State Language in the Republic of Uzbekistan is based on the Latin alphabet and simplifies the use of comments in GAMS. On the other hand, the language of the Russian Federation uses the Cyrillic alphabet and this alphabet can not be used for comments in GAMS.

11.3 Declarations and Definitions

For most structures in the GAMS language, there are two types of statements:

1. DECLARATION – assignment of a name and possibly a value to a stated parameter, and
2. DEFINITION – calculation using previously defined parameters or structures

Parameters always require DECLARATIONS. Assignment of numerical values to parameters is not always compulsory. However, equations always require both declaration and definition. DEFINITION of equations means the construction of their mathematical structure, and for parameters it means their direct computation. While numerical interrelations are specified in the body of an equation, the order of equation definition does not matter. There is no such notion as "first equation", "second equation" and so on. For parameters and constants simultaneous declaration and definition in one operation is possible. Examples are given below.

For a series of GAMS structures, it is possible to use long names. However, it is better to limit yourself to 8 letters. All names of structures in the GAMS language must begin with a letter and can contain digits or a limited number of other symbols (maximum possible number is shown in the examples).

11.4 Terms, Symbols and Reserved Words

Every programming language, including GAMS, has keywords which have special meaning. Therefore, they cannot be used in the names of variables and parameters. For example, the name ASSIGN can not be used for any variable, because GAMS will read this as a keyword with special meaning, note it as improperly used, and show that an error has occurred. Table 11.4.1 is a list of GAMS keywords.

Table 11.4.1
Keywords in GAMS.

Abort	acronym	Acronyms	Alias	all	and
assign	binary	Card	Display	eps	eq
equation	equations	Ge	Gt	inf	integer
Le	loop	Lt	Maximizing	minimizing	model
models	na	Ne		negative	not
options	or	Ord	Parameter	parameters	positive
Prod	scalar	Scalars	Set	sets	smax
Smin	sos1	Sos2	Sum	system	table
Using	variable	Variables	Or	yes	repeat
Until	while	If	Then	else	semicont
semiint	file	Files	Putpage	puttl	free
No	option	Solve	For		

Table 11.2.2
Special Combinations of Symbols and Their Meaning.

=l=	Less	--	previous in cycle
=g=	More	++	next in cycle
=n=	not equal	**	involution
arithmetic	+,-,/,*		

Table 11.2.3
Symbols and Their Usage.

A to Z	Alphabet	-	minus
a to z	Alphabet	( )	parentheses
&	Ampersand	[ ]	brackets
*	Asterisk	{}	braces
@	at	%	percent
\	Slash	From 0 to 9	digits
:	Colon	#	number
,	Comma	?	question mark
$	Dollar	;	semicolon
.	Period	'	quotation marks, single
+	Plus	/	virgule
"	quotation marks, double_	Underlining	
=	Equals	!	exclamation point
<	is less than	^	diacritic mark (over a vowel)
>	is more than		

12. VARIABLES

12.1 Types and Declaration

Variables are the major object with which GAMS operates. To define a list of variables we use the functional word VARIABLE. There are five types of variables as indicated in Table 124.1.1.

Table 12.1.1
Type of Defining Variables in GAMS.

VARIABLE	*Variables* belong to the entire set of real numbers from "minus" infinity to "plus" infinity
POSITIVE VARIABLE	*Positive variables* belong to the entire set of positive real numbers from zero to "plus" infinity
NEGATIVE VARIABLES	*Negative variables* belong to the entire set of negative real numbers from zero to "minus" infinity
INTEGER VARIABLES	*Integer variables* belong to the entire set of positive integers
BINARY VARIABLES	*Binary variables* are Integer variables that can take the value of 1 or 0 only

VARIABLES can be zero-dimensional or multidimensional. The dimensions are determined by the availability and number of indices. The declaration of variables is based on the following principle. For example, consider the variables from the nonlinear equation model in Section 2.1:

```
VARIABLES x, y1, y2, obj;
```

These are all aero-dimensional (scalar) variables. Variables are included in equations along with parameters and constants. Variables are always defined in the solution process. However, users and modelers are twice faced with variables when they make up a model. The first time, they face variables while declaring them and the second time in the equations.

12.2 Bounds of Variables, Initial and Fixed Values

In most models some variables have bounds on their values. For instance, if some variables are declared as positive, then "zero" is the lower boundary. The same boundary exists for negative variables, but "zero" is their maximum possible value. In addition, other boundaries can be defined with the help of specific suffixes on variables (see Table 12.2.1).

Table 12.2.1
Main Suffixes.

Suffix	Description
.L	Value (level)
.LO	Lower boundary
.UP	Upper boundary
.FX	Fixed value
.m	Dual value

GAMS begins the search of variables for a solution from a boundary of the allowable area of values, most often from zero. However, zero is a dangerous value which should be avoided if possible.

Examples show best how suffixes act.

```
S.lo=2.0;
```

defines the lower bound on instream flow in the water user model of Section 3.1.

```
b.up=100;
```

defines the upper bound of the coefficient b in the reservoir example of Section 3.2.

```
put  "Solution x =  ", put x.l, put /;
```

indicates that the value (level) of the variable x should be printed in the nonlinear equation example of Section 2.1.

In general, the more the user constricts the boundaries of variable values, the sooner a solution will be obained. However, you should not constrict the boundaries of search too much, because in complicated models you may have the situation where there is no area left to search for solutions at

all. Namely, there may be systems of equations in which one variable is within the boundaries allowed by the user, but the second variable is certain to be in an unallowed area. Usually, the consequences of calculating such systems are grave.

13. EQUATIONS

The block of equations in GAMS models consists of two parts:

1. Declaration of the names of equations
2. Definition of the equations

13.1 Declaration of Equations

Declaration of equation names is similar to the declaration of SETs or PARAMETERs. The similarity is in the fact that a list and comments are allowed and recommended. The syntax of declaring equation names is as follows:

```
EQUATION   name  comment ;
```

EQUATION is a keyword which must appear before the name of each equation if there is semicolon at the end of the line. It can be found only one time at the beginning of a list of equation names. Commas separate different names of equations in a row or one row is assigned for each name.

The *name* can include two parts:

- an *identifier* of the name of the equation, and
- *indices* in parentheses.

The identifier (name) of an equation may include no more than 10 symbols and must always begin with a letter. Indices can be either letters or digits representing SETS. Comments may not be more than 80 symbols long and should be on the row where the identifier is located. Consider the example of the equations from the model for solving nonlinear equations presented in Section 2.1

```
EQUATIONS Eq1, Eq2, Objective;
```

This declares three scalar equations, the first equation is named *Eq1*, the second *Eq2*, and the third *objective*.

13.2 Definition of Equations

The definition of equation structures is a mathematical peculiarity in the GAMS language. The syntax for defining equations in GAMS is as follows:

```
name..            expression // special relation character // expression;
```

First, we have *name* (the name of the equation) as shown in the list of declared equation names, then two periods ".." follow separating the name of the equation and the expression of the equation. The expression of the equation is an algebraic statement using allowed GAMS algebraic operations. The equation must contain a relation operator, such as:

```
=E=   right hand side is equal to the left hand side
=G=   right hand side is greater than or equal to the left hand side
=L=   right part is less than or equal to the left hand side
```

The user must carefully balance the number of open and closed brackets. An equation can occupy an unlimited number of rows. Spaces can be used for obvious and convenient comprehension. Equations, once defined, cannot be changed or redefined in a later portion of a model. If such necessity arises, it is essential to introduce a new equation with a new name.

However, it is possible to change the number of estimated elements of the equation by changing the data which the equation uses. In addition, you can exclude part of an equation by using logical conditions ($ operator) in the name of an equation or in the computation part of an equation. Again, consider the scalar equations from the model in Section 2.1

```
EQUATIONS Eq1, Eq2, Objective;

Eq1.. y1 =E= 10*x*x+10*x+10;
Eq2.. y2 =E= -10*x+100;
Objective.. obj =E= (y1-y2)*(y1-y2);
```

Scalar equations are often encountered while making up optimization models. The examples written above are scalar equations. They contain only scalar variables. However, scalar equations can contain index variables and use index operators. For example, consider the equations form the model of water users presented in Section 3.1

```
EQUATIONS objective, cap;

objective..obj =E= SUM(i,a(i)*x(i)+b(i)*x(i)**2);

cap..sum(i,x(i))+s-r =E= 0.0;
```

In this case the names of the equations are scalar, and the equations use the index SET '*i*'. After summing, the index disappears and the equations are scalar. Correspondence between the dimensions of the name and left and right parts of equations is essential in constructing models.

Index (vector) equations differ from scalar equations only in the closer correspondence between the dimensions of the equation's left and right parts. SETs which take a position in an equation to the left of the ".." and immediately after the name of an equation, can be considered as control indices, because the equation will be solved for all possible values of the indices. Consider the model of

least squares estimation presented in Section 2.2

```
EQUATION mod(t), residual(t), objective;

mod(t).. a*x1(t)*x1(t)-b/x3(t)-c/x2(t)+EXP(-y(t)*y(t)) =E= y(t);

residual(t).. e(t) =E= y(t)-y_hat(t);

objective.. obj=E=sum(t,power(e(t),2));
```

The equation "*objective*" is a scalar equation, but the equations "*mod(t)*" and "*residual(t)*" are index equations indexed on the SET "*t*" and they will be solved for every value of the set.

Each GAMS model must contain at least one scalar variable (the value of an objective function being optimized) and one scalar equation (the objective function). GAMS can optimize only a scalar variable.

13.3 Using Symbols as Indices in Equations

Very often, it is necessary to use symbols (e.g., values of SETs or indices) in equations. This can be done using quotation marks. As an example, consider the establishment of upper bounds (capacity) on reservoir 1 storage in the model of river system management in Section 3.3

```
S.up('Res_1' ,t) = 1000;
```

The bound will be set only for reservoir '*Res_1*'. In this instance, SET *t* in the variable *s('Res_1',t)* is defined for all twelve months.

13.4 Assembling a Model

After all the SETS, PARAMETERS, VARIABLES, and EQUATIONS of a model have been declared a system (called a *MODEL*) can be formed from a collection of equations and given a name. When all the declared equations are included in the model, the syntax of declaring the model is as follows:

```
MODEL    name  comment /ALL/;
```

For example:

```
MODEL Lakes /all/;
```

The model is named "*Lakes*", and the keyword *ALL* means that the model includes all of the equations. Sometimes, the user may want to create numerous variants of equations and unite them in groups, forming different models with distinct names and they may have various optimization

objectives.

13.5 Solving a Model

Once a model has been defined, the next step is to solve it. This is ordered by including a *SOLVE* statement in the model. Consider the example of the river system management model in Section 3.3

```
SOLVE Lakes USING NLP MINIMIZING obj;
```

The *SOLVE* statement contains 6 parts:

1. The word *SOLVE* (required);
2. The previously defined name of the model (*Lakes*) (required but the name is user defined);
3. The word *USING* (required);
4. The type of "*solver*" to use in the solution (*NLP* in this case, required but the name is user defined). See Table 13.5.1 for a list of available solvers;
5. The word *MINIMIZING* or *MAXIMIZING* (required); and
6. The name of a scalar variable (*obj* in this case) to be optimized (required but the name is user defined)

Table 13.5.1
Types of Solvers in GAMS (abbreviated list).

Type of Solver	*Description*
LP	Linear programming. The model cannot contain nonlinear or discrete (binary and integer) variables.
NLP	Nonlinear programming. In the model, nonlinear forms must be continuous functions and the model may not contain discrete variables.
MIP	Mixed integer programming. Similar to RMIP, but the requirements of discreteness of variables and equations are stringent. Discrete variables must take discrete values within boundaries.
MINLP	Mixed integer nonlinear programming. The same characteristics as for RMINLP, but the requirements of discreteness are very stringent.

14. OUTPUT TO TEXT FILES

14.1 Declaration of Output Files

GAMS allows writing information about the model and its solution to text files. In addition, a text output file "*model_name.LST*" will always be produced. This file is created automatically, but its format is not entirely useful for analysis purposes. For display of specific information in this file, the *DISPLAY* operator can be used. This operator is best appreciate by studying it in examples. This

operator can be useful when testing programs, since it is possible to display information in the quickest way. However, there is an opportunity to create additional output files. The following is the full syntax for defining output text files

```
FILE pointer_name comments / external_filename /
```

where `FILE` is a keyword used to define files;
`pointer_name` is a pointer used by GAMS model to direct output to the external text file *external_filename*;
`external_filename` is an external text file which receives the output from GAMS.

GAMS allows working with a set of output files, but you should read about it in the main GAMS User Guide.

Consider the model for solving a set of nonlinear equations in Section 2.1

```
FILE res /Eq1.txt/
```

where `res` is a pointer to the external file named "*Eq1.txt*"; and
`Eq1.txt` is the text file that will receive the model output.

The pointer to the output file is named *res* in the model and it directs information into the file *Eq1.txt*.

14.2 Output of Information

The *PUT* operator is used for both assigning the name of the "*current_file*" to a declared text file and writing information into this file. The general syntax of the *PUT* operator is

```
put  Filename_1  elements;
```

Consider the model from Section 2.1 again as an example

```
PUT  res;
put  "Solution x =  ", put x.l, put /;
```

This example shows how the pointer *res* is associated with the external text file *Eq1.txt*. In the first line, the output is directed into the file associated with the pointer *res*. In the next line, some text and the value (level) of the variable *x* is inserted into the file.

Sometimes, it is necessary to output the value of parameters or tables. This usually requires the use fo the LOOP function. Consider the example from the least swuares model of Section 2.2

```
PUT " t   x(1,t)   x(2,t)     x(3,t)     y(t)     y_hat(t)"/;
LOOP((t),PUT t.TL, x1(t), x2(t), x3(t), y.L(t), y_hat(t)/;);
```

Where the first line prints a header at the top of a table and the second line loops through all the vales of the set *t* and prints the value of *t, x1(t), x2(t), x3(t), y.L(t),* and *y_hat(t).* Note that the variable *y* needs the suffix "*.L*" after it, but the parameter *y_hat* does not. This is because *y_hat* is a defined data whose "level" is not determined by GAMS.

15. SETS

15.1 Set Naming and Declaration

SETS are a major item in the GAMS language. SETS can be used to construct connections between variables and equations in models. SETS are the equivalent of indices in a typical programming language. Usually, ordinals function as indices, for instance "first", "second", etc. In GAMS, indices have names, written through a combination of letters and digits, without spaces. On the one hand, these are ordinals, but on the other hand, they have names. A set name must begin with either a letter or a digit, but the next symbol can be a letter, digit or the marks "+" and "-". For example:

```
Jlobest     1999  1972-73
ROKETer     D6H83       Navruz-99
```

The next example illustrates using SETS in the GAMS language

```
SETS  i  plants          / Tashkent, Almaty/
      j  consumers       / Tokyo, London, Moscow/;
```

GAMS remembers the sequence of elements in a SET. For instance here, *Almaty* is second among *i*, and *Moscow* is third among *j*. The group of SETS begins with the word SETS and ends with the semicolon symbol ";". It is possible to declare the same information in the model through two individual groups:

```
SET       i  plants      / Tashkent, Almaty /;
SET       j  consumers   / Tokyo, London , Moscow /;
```

Symbols "/" separate lists of SET elements. SET Element in a list are separated by commas. It is possible to write a comment between the SET name and the slashes, but all comments must fit on one row.

GAMS models can be very large and contain an enormous number of structures. One GAMS option that eases the definition of elements of a SET is shown in the example below

```
SET t  year    /1965, 1966, 1967, ...    ...   ... 1998, 1999/;
```

A different, and shorter, way to accomplish this in GAMS is

```
SET t  year    /1965*1999/;
```

Sometimes in models, we may want to define two identical SETS

```
SET   j  station  / Toktogul, Uchkurgan, Charvak, Chardara /;
SET   k  station  / Toktogul, Uchkurgan, Charvak, Chardara /;
```

Simultaneous definition of the same SET under different names is useful when the modeler wants to use the SET element interrelations in a model. To avoid repetition, two sets can be made identical by using the "ALIAS" command. For example

```
SET   j  station  / Toktogul, Uchkurgan, Charvak, Chardara /;
ALIAS(j,k);
```

That is, the set *j* was created and assigned members, and the set *k* was created and made identical to the set *j*.

In GAMS, it is possible to define subsets of sets. Consider the model of water management from Section 3.3 with the river network as shown in Figure 15.1.1 below.

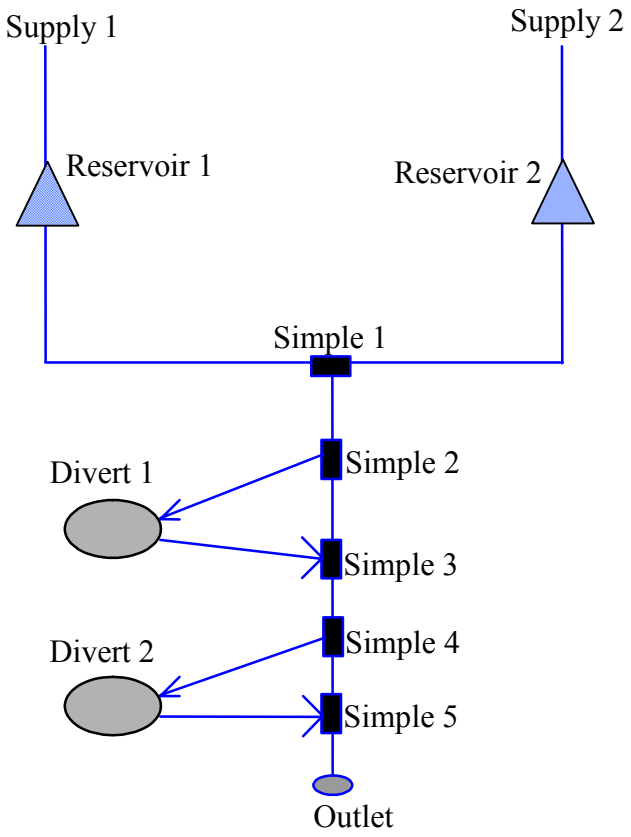


Figure 15.1.1. Structure of a River Network for the Water Management Task

The set of nodes in the river system is defined as

```
SET  n nodes
/
Supply_1, Supply_2,
Simple_1*Simple_5,
Divert_1, Divert_2,
Res_1, Res_2,
Outlet /;
```

and five subsets of this set are created which have various characteristics such as junctions, water sources, water users, irrigation zones, and reservoirs

```
SET
nn(n)     Simple nodes     /Simple_1*Simple_5/
ns(n)     Supply nodes     /Supply_1, Supply_2/
nr(n)     Water user nodes /Divert_1, Divert_2, Outlet/
nrr(n)    Irrigation nodes /Divert_1, Divert_2/
nl(n)     Reservoir nodes  /Res_1, Res_2/;
```

In this example, *n* is a collection of nodes in the modeling network, i.e., a list of objects to be used in the model. The subsets *nn*, *ns*, *nr*, *nrr*, and *nl* are subsets of the set *n* which have various geographical significance as specified in the name (which is a comment and GAMS ignores this). Among the objects are simple (junction) nodes, *nn*, water sources (supplies) nodes, *ns*, water users, *nr*, irrigation nodes, *nrr*, and reservoir nodes, *nl*. *ns(n)* is a subset of flow sources, elements of which, *Supply_1,* and *Supply_2*. The same is true for the subsets of water users *nr(n)* and reservoirs *nl(n)*. Note, that the subset *nr*( ) does not include all elements of set *n*, but only a part of them.

15.2 SET Operations

Operations can be performed on sets using the symbols

+, -, *, not

Symbol " + " performs the set union operation

```
s3(n) = s1(n) + s2(n);
```

The set *s3(n)* is the set equal to all elements in *s1(n)* and all elements in *s2(n*

Operator "*" performs the set intersection operation; only the elements included in both set A and set B belong to the intersection of the sets A and B

```
s3(n)  = s1(n) * s2(n);
```

s3(n) is a set consisting of the elements which are both in *s1(n)* and in *s2(n)*.

The symbol "NOT" performs the set complement operation

```
s3(n) = NOT s1(n);
```

The set *s3(n)* consists of elements included in *n*, but not in *s1(n)*. That is, all elements in *s1(n)* are deleted in *s3(n)*.

Operator "-" performs the operation of difference of sets. This set consists of elements, which belong to set A but not set B

```
s3(n) = s1(n) - s2(n);
```

s3(n) is a set consisting of elements which are members of *s1(n)*, but not members of *s2(n)*.

15.3 Defining Multivariable Indices

Multidimensional SETS are often useful in modeling, especially when connections between set elements are important components of the system being considered. In a multidimensional set, the indices of the set elements are separated by commas. Again, consider the model of water management from Section 3.3 with the river network as shown in Figure 12.1.1 above. A portion of the GAMS code which defines a two-dimensional set in this model is

```
ALIAS(n,n1);

SET
n_from_n(n,n1)  node n gets water from node n1 (any node)
/
Res_1.Supply_1,
Res_2.Supply_2,
Simple_1.Res_1,
Simple_1.Res_2,
Simple_2.Simple_1,
Divert_1.Simple_2,
Simple_3.Simple_2,
Simple_3.Divert_1,
Simple_4.Simple_3,
Divert_2.Simple_4,
Simple_5.Simple_4,
Simple_5.Divert_2,
Outlet.Simple_5
/;
```

Here, the set *n1*, which is identical to set *n*, is created using the "*ALIAS*" command. Then the two-dimensional set *n_from_n(n,n1)* is created which contains all of the downstream to upstream connections in the network. That is, it shows where each node in the system gets water from. The set elements are comprised of one element from the first set, *n*, and one element from the second set, *n1*, separate by a period. For example, the first element is *Res_1.Supply_1* which indicates that reservoir 1 gets water from water supply 1, and so on. This set is used in the model to determine the sources of water for any node in the system.

As another example, consider the set of main reservoirs on the Syrdarya River in Central Asia

```
SET  r  reservoirs  / Toktogul, Kairakum, Andijan, Charvak, Chardara /;
```

Each reservoir is located in one of the four countries (Kyrgyzstan, Uzbekistan, Tajikistan, or Kazakhstan) of the Syrdarya Basin. We can represent these countries as another set

```
SET  c  country     / Kyrgyzstan, Uzbekistan, Tajikistan, Kazakhstan /;
```

Then a two-dimentsional set can be created to associate each reservoir with its country, viz.,

```
SET  r_to_c(r,c)   reservoir_to_country
    / Toktogul.Kyrgyzstan,
      Kairakum.Tajikistan,
      Andijan.Uzbekistan,
      Charvak.Uzbekistan,
      Chardara.Kazakhstan /;
```

The previous example was an example of ONE-TO-ONE mapping. We may also have MANY-TO-MANY mappings in GAMS. Examples of this construction in the GAMS language can be found in the Gams Language Guide.

15.4 ORD( ) and CARD( ) Operators

The GAMS compiler memorizes index names of a set as well as their sequence of definition. If the sequence of a set is arranged once, it will remain that way for other cases.

15.4.1 ORD( ) Operator

The *ORD* operator returns an ordinal number equal to the index position in a set. Here are some examples showing how the ORD operator is used.

```
SET t      /100*120/
PARAMETER  val(t);
val(t) = ORD(t);
```

The result of the operator will be the following:

```
val('100') = 1;
val('101') = 2,
…
val('120') = 21
```

15.4.2 CARD( ) Operator

The *CARD* operator returns an ordinal number equal to the number of elements in a set. For example:

```
SET  t     /45*68/
PARAMETER  s;
s = CARD(t);
```

The result will be that the *CARD( )* operator is assigned the value 24 (the number of values from 45 to 68, inclusive). The *CARD( )* operator is frequently used for initiating and canceling computation at the final stage. Consider the example

```
c.fx(t)$( ORD(t) = CARD(t) ) = demand(t);
```

which shows that when the value of the index *t* is equation to the number of items in the set, i.e., the last value, the value of the variable *c* is fixed equal to the parameter *demand(t)*.

15.5 LAG and LEAD Operators

Operators *LAG* and *LEAD* are used for correlating an element of a set with the next or preceding element of the set. GAMS has two variants of *LAG* and *LEAD* operators:

9. Linear LAG and LEAD operators (+, -)
10. Circular LAG and LEAD operators (++, --).

The difference between these two types of operators is in the method of processing at the initial and final points of the sequence. In circular operators, after the final element of the sequence, there comes the first, while in linear operators sets are broken.

15.5.1 Linear LAG and LEAD Operators

Linear *LAG* and *LEAD* operators (+, -) are very convenient for modeling time periods that are not cycled. A particular year is an example if the user does not want the initial values to be identical with the final ones. For example, it is unlikely that the initial storage of a long-term reservoir will be identical to its ending storage. GAMS can differentiate linear operators (+, -) from arithmetic operators according to the context.

Consider the following example from the reservoir operation model in Section 3.2.2 that illustrates application of linear *LAG* operator

```
balance(t).. (1+a(t))*S(t) =E= (1-a(t))*beg_S $(ord(t) EQ 1)
                             + (1-a(t))*S(t-1)$(ord(t) GT 1)
                             + Q(t) - R(t)- b(t);
```

In this case, when the index $t = 1$, the initial storage (*beg_s*) of the reservoir is used in the balance equation, but when the index $t > 1$ the variable storage is used in the calsulation.

15.5.2 Circular LAG and LEAD Operators

When circular *LAG* and *LEAD* operators are used the initial and final values of sets are adjacent, i.e. the "first-1" is "final", the "last +2" is the "first +1", etc. The circular LAG and LEAD operators (++, - -) are convenient for modeling cycled periods of time. A particular year is an example, if the user wants the initial values to be identical with the final ones. For example, often the initial storage of a seasonal reservoir is to be identical with its end storage. Naturally, January is the month following December despite changing years. This is illusteadte using the previous example, where it is not desired to specify the intial storage volume of the reservoir

```
balance(t).. (1+a(t))*S(t) =E= (1-a(t))*S(t--1) + Q(t) - R(t)- b(t);
```

That is, the storage value for time period t =card(t)+1 it is set equal to the value 1. This formulation of the equation will lead to a long-term, stationary solution for the model whch is independent of the initial storage condition of the reservoir (Loucks et al., 1981, Sec 2.8.1).

The operators *LEAD* and *LAG* can be used in assignments. Use of operators *LAG* and *LEAD* on the left side of the assignment is called an "index" while the use of it on the right side is called "assignment" and includes definition of the area of assignment.

The *LAG* and *LEAD* operators can be used in equations. Principles defined above for these operators are true for equations. The location of operators *LAG* and *LEAD* in the body of the equation is very important. *LAG* and *LEAD* operators located on the left side of the symbol ".." are modifications of the area for defining equations. They become indicators if located on the right side of the symbol "..". If their conditions are not met, they disappear.

16. SCALARS, PARAMETERS, AND TABLES

16.1 General Rules

The major working elements of the GAMS language are identifiers: names given to sets, parameters, variables, models, etc. Identifiers are required to start with a letter followed by up to 9 letters or digits (total length is 10 characters). Examples of inadmissible identifiers include

```
15    $rhs       0ie        ewoifkoioiiio          ueru&wc
```

Digital data are contained in arrays (zero, scalar, or multidimensional matrices called *parameters* in GAMS). The SETs, described above, can play the role of indices for these arrays. To declare an array to contain data values, we use the functional words:

```
SCALAR                (zero-dimensional),
PARAMETER         (one-dimensional), and
TABLE             (multidimensional).
```

Parameters are declared in a free format with the following general structure

```
Parameter a(i,j)  input-output matrix
```

where	`Parameter`	data-type-keyword (required)
	`a`	identifier (required)
	`(i,j)`	domain list (optional)
	`input-output matrix`	text (optional)

Scalars are defined similarly but without the domain list. *Tables* are simply multi-dimensional versions of parameters.

The assignment of numerical values to parameters can be carried out in two ways:

1) Assignment, and
2) Computation.

16.2 Data Entry Through Assignment

As an example of how to enter numerical values in a GAMS model, consider the model for solving a set of nonlinear equations in Section 2.1. The coefficients of the set of equations are defined using SCALARS

```
SCALAR A /10/, B /10/, C /10/, M /-10/, N /100/;
```

Parameters *A, B, C, D, M,* and *N* are used in the model as zero dimensional parameters (scalars). The values are assigned simultaneously upon declaration (statement of fact of existence) of the scalars.

The sequence of declaration and definition of numerical values for one-dimensional arrays (*PARAMETERS*) is a bit different. Conside the model of least squares estimation from Section 2.2

```
SETS t  / 1, 2, 3, 4, 5, 6, 7, 8 /;

PARAMETER x1(t)  /1 2, 2 3, 3 3, 4 3, 5 5, 6 5, 7 6, 8 7/;
```

In this example, declaration and definition occur simultaneously. Each pair of values of the index *t* and the parameter *x1(t)* are separated by commas and the entire list is enclosed in slashes.

The following example from the model of river management in Section 3.3 illustrates data entry to a TABLE at the moment of definition:

```
SET n nodes
/ Supply_1, Supply_2,
  Simple_1*Simple_5,
  Divert_1, Divert_2,
  Res_1, Res_2,
```

```
  Outlet /;

SET t months / Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec /;

TABLE Supply(n,t) water supplies (m3 per sec)
          Jan   Feb   Mar   Apr  May  Jun  Jul  Aug   Sep   Oct   Nov   Dec
Supply_1  128   125   234   360  541  645  807  512   267   210   181   128
Supply_2   39    39    52   121  168  144  105   78    49    44    45    39
```

In this example, the water supply to each of two sources is defined as a table with two rows, one for each supply, and twelve columns, one for each month. If some cells are missing from a table, it means that the value of the array for the corresponding complex of indices is not defined. However, if this element of the array is used in further computations, it will be considered as equal to zero. Note that only two of the elements of the SET *n* are used in the table (the source nodes *Supply_1* and *Supply_2*), the rows of the table for the other elements are not defined, but we can still use the SET *n* later in computations as if each node might have a source of water associated with it.

16.3 Data Entry Through Computation

After declaration, parameter values may be computed later in the model. Determination of data through direct computation or assignment following the initial definition is possible and sometimes useful. Consider the model of two-dimensional groundwater flow in Section 4.5 where the finite-difference coefficients are computed as parameters

```
PARAMETER
        A(I,J)   finite-difference coefficient
        B(I,J)   finite-difference coefficient
        C(I,J)   finite-difference coefficient
        D(I,J)   finite-difference coefficient;

        A(I,J) =  T/(DX*DX) ;
        B(I,J) = -T/(DX*DX) ;
        C(I,J) =  T/(DY*DY) ;
        D(I,J) = -T/(DY*DY) ;
```

Note the lack of any instructions as to which index values of *I* or *J* should be used in the calculation. GAMS carries out these computations for *all* indices *I* and *J* which are defined through the SET operator. If you need to assign or compute a specific value of one or a small group of elements of an array, you can do this as follows, for example:

```
c( 'Tashkent', 'Fergana' )  =  f * 0.001 ;
c( 'Tashkent', 'Chinaz' )   =  20 * 0.001 ;
```

To organize computations within some limits of the variables *i* and *j*, you can use subsets as discussed previously

A collection of standard logical and mathematical functions may be used in GAMS models for assignment or computation purposes. The reader is referred to the GAMS Users Manual for descriptions of these functions. Besides those standard functions, two special functions are available

for the user:

```
SUM(..., ...)
PROD(..., ...)
```

An example of the SUM function is given in the objective function of the least squares estimation model in Section 2.2

```
objective.. obj=E=sum(t,power(e(t),2));
```

This means that the value of "*obj*" will be assigned the algebraic sum of all elements of the unidimensional array "*e*(*t*)" summed over the entire range of indices "*t*".

An example of the PROD function is

```
t(i) = PROD(j,b(i,j));
```

This means that in the unidimensional array "*t( )*" each element, indexed by "*i*", will be defined as a product of elements of the two-dimensional array "*b( )*" on "*j*" with the fixed related index "*i*".

17. CONDITIONAL OPERATOR ($)

17.1 General Statements

This section considers the conditional operator (*$*), which is one of the strongest features of the GAMS language. This operator provides the ability to guide calculations or definitions with logical conditions. The general format of the operator is

```
item$(condition)
```

and this can be read "fulfill the item if the condition is true", where the condition is of logical character capable of acquiring Boolean values (true or false, existence or nonexistence). Conditions may not contain variables. However, the use of suffixes on variables (e.g., *.L* or *.M*) is allowed in $ operations, but this transfers the variables into constant values.

The essence of the $ operator can be explained by considering the following example

```
if (b > 1.5), then a = 2;
```

which can be written in the GAMS language as follows

```
a$( b > 1.5 ) = 2;
```

If the condition in the parathenses '*b > 1.5*' is not fulfilled then the assignment '*= 2*' is not executed.

17.2 Embedded $ Conditions

Conditions can be embedded or nested in structures. The term *$(condition1$(condition2))* can be read as *$(condition1 and condition2)*. That is, the first and second conditions must be fulfilled simultaneously before the assignment will take place. When using embedded $ conditions all expressions following the $ sign must be inserted into a pair of parentheses.

Consider the following example:

```
d(i)$( w $q ) = v(i);
```

where *i*, *d*, and *w* are sets, while *q* and *v(i)* are parameters. The assignment will be executed for elements *d(i)* for which simultaneously the values of *w* and *q* have true values. The operator given above can be rewritten as follows:

```
d(i)$( w AND q ) = v(i);
```

17.3 Conditional Assignment

The example given above was connected with conditional assignment. In this example, the $ condition was on the left side of the assignment. The effect of the $ condition depends on its position with regard to the equal sign. The assignment is not fulfilled if the logical conditions are not satisfied. This means that values of parameters on the left will remain unchanged for those indices where conditions are not fulfilled.

If the parameters of the left side of the assignment are not numerically determined, their values will be equal to 0, since the logical conditions are not satisfied. The $ operator is very useful for cases when it is necessary to avoid dividing by zero. For example:

```
r(i)$( z(i) NE 0 ) = 1./z(i);
```

The *z(i)* parameter was defined in the model and the operator is used to calculate *r(i)* without making mistakes. If any value associated with *z(i)* is equal to 0, the assignment will not be executed and the preceding value for *r(i)* will persist. If *(i)* was not initialized before, then all indices with *z(i)* equaling 0, will be assigned 0 value. If *z(i)* was not defined for cases when it equaled 0, then the assignment above can be written as follows:

```
r(i)$z(i)  =  1./z(i);
```

17.4 Conditional Computation

Assignment is always executed when the assignment operator appears with a $ condition on the right side of the assignment operator (equal sign). If the logical condition is not fulfilled, then the parameter or the variable will be transformed into 0, even if it was equal to some number previously. For example

```
x = 20$(y > 7.5);
```

which means

```
if (y > 7.5)  then (x=20),  else (x=0)
```

Another example is from reservoir management model of Section 3.2

```
balance(t).. (1+a(t))*S(t) =E= (1-a(t))*beg_S $(ord(t) EQ 1)
                             + (1-a(t))*S(t-1)$(ord(t) GT 1)
                             + Q(t) - R(t)- b(t);
```

In this case, when the index $t = 1$, the initial storage (*beg_s*) of the reservoir is used in the balance equation, otherwise it is omitted. Similarly when the index $t > 1$ the variable *S(t-1)* is used in the calculation, otherwise it is omitted.

17.5 Conditional Indexing

Another useful application of the $ condition is the control of index operations. For example

```
a  =  SUM( i$(n(i)), b(i));
```

The operator calculates the sum of *b(i)* for all *i* for which the parameter *n(i)* has a true value.

Another example of using $ operator is the correlation of SET-to-SET operations. The river management model of Section 3.3 uses the set *n_from_n(n,n1)* which defines the upstream to downstream connections in the flow network. When computing the flow balance for nodes in the network, it is necessary to know which upstream nodes contribute to the inflow of a node *n* under consideration. This is accomplished by summing over all nodes *n1* which are upstream from node *n* in the *n_from_n(n,n1)* set. That is

```
* Simple node:  Inflow = Sum of Releases from Upstream - Sum of Diversions
R_nn(n,t).. Q(n,t) =e= sum(n1$(n_from_n(n,n1)),R(n1,t))
                     - sum(n1$(n_to_nr(n,n1)),Divert(n1,t));
```

Similarly, in this equation, the sum of all water flowing from the node *n* to downstream nodes *n1* is accomplished by conditioning the sum on the set *n_to_nr(n,n1)*.

17.6 Conditional Equations

The $ operator is also used for managing the definition of equations. $ control of equation definition restricts the number of equations included in the model. Consider the following example from the river system management model in Section 3.3

```
* Reservoir node:  Release = Mass Balance
R_nl(n,t)$(nl(n)).. S(n,t) =e= beg_S(n)$(ord(t) EQ 1)
                             +  S(n,t-1)$(ord(t) GT 1)
                             + Q(n,t)-R(n,t);
```

The equation *R_nl(n,t)* is conditioned by the expression *$(nl(t))*. The equation is included for the indeces *n* and *t* if the calculation node, *n*, refers to a member of *nl(t)*, the set of reservoirs.

17.7 Special Constructions for Conditional Variables

One of the main difficulties of using $ operators is that they are prohibited from use with variables computed during the optimization process. Suppose we have to solve a problem with the equations:

$$y = \begin{cases} x^2 & \text{if } x < 0 \\ x^3 & otherwise \end{cases}$$

where x, and y are variables. We have the following structure of computation

```
VARIABLES y, x;
EQUATION  first;
first.. y  =E=  (x*x)$(x < 0) + (x*x*x)$(x > 0);
MODEL andre /ALL/;
SOLVE andre USING NLP MINIMIZING y;
```

The GAMS compiler will inform you of an error made by using the $ operator with variables.

```
**** THE FOLLOWING NLP ERRORS WERE DETECTED IN MODEL ANDR:
**** 53 IN EQUATION FIRST      .. ENDOG $ OPERATION
**** 53 IN EQUATION FIRST      .. ENDOG $ OPERATION
```

However, using the feature of the computer to define a square root as positive value, the necessary result can be achieved by using the following construction

$$y = x^2 * \left(\frac{\sqrt{x^2} - x}{2\sqrt{x^2}} \right) + x^3 * \left(\frac{\sqrt{x^2} + x}{2\sqrt{x^2}} \right)$$

or in the GAMS language

```
first.. y =e= (x*x)*(0.5*(sqrt(x*x)-x)/(sqrt(x*x)+0.000001))
                + (x*x*x)*(0.5*(sqrt(x*x)+x)/(sqrt(x*x)+0.000001));
```

18. ADDITIONAL OPERATORS

GAMS has a number of other operators frequently used in other programming languages. This section discusses these operators and their use. There operators are:

1) LOOP - cycle operator;
2) IF-ELSE - conditional operator;
3) FOR - cycle operator; and
4) WHILE - conditional cycle operator.

18.1 Loop Operator

The LOOP operator is often used with the PUT operator for printing out and displaying information on the screen or in files. The operator has the following syntax:

LOOP (control_ area [$(condition)] operator {; operator});

If *control_area* contains more than one set, then parentheses are needed. The LOOP operator provides performance of the operator within the cycle limits for each index of the set in turn. The sequence of operations is determined by the sequence of indices. LOOP is a general type of index operation. The cycle set can be controlled by the $ operator, but it must be static or embedded. The cycles can be controlled by more than one set. However, it is impossible to declare or solve equations within the LOOP operator. It is not allowed to transform control sets within the body of the cycle.

Consider the following hypothetical case of the integration model:

```
SET  x  / d15*d25 /
PARAMETER  nachalo(x)  / d15 423 /
           differ(x)   / d15  58.4, d16  47.7, d17  68.8
                         d18  46.5, d19  57.9,  d20  46.9
                         d21  64.5, d22  65.4,  d23  34.7
                         d24  76.4, d25  34.4/;
```

The LOOP operator can be used to compute increasing sums:

```
LOOP(x, nachalo(x+1)  =  nachalo(x)  +  differ(x));
```

Only one operator and one control set - *x* is presented here within the cycle operation area

```
nachalo(x+1) = nachalo(x) + differ(x)
```

Consider the following example used in the algorithm of the Newton method for extracting a square root. In practice, the imbedded function SQRT() is used. However, in this case, we show the functioning of LOOP operator using a program from the GMAS Language Guide.

The method is:

if x is an approximate square root of v, then $(x+v/x) - 2*x = 0$

The GAMS code is:

```
SET              i number of iterations is 100      /i-1*i-100/;
PARAMETER        x(i);
SCALARS
     v           number whose square root is sought /23.456/
     sqrtval     answer
     err         error /1.0/
     tol         error tolerance /1.0e-06/;

x("i-1")  =  v/2 ;

loop(i$(err > tol),
            x(i+1) = 0.5*(x(i) + v/x(i));
            sqrtval = x(i+1);
            err  = ABS (x(i+1) - x(i))/x(i+1)
       );

ABORT$(err>tol) "square root not found"

DISPLAY "square root found",sqrtval, x;
```

The result:

```
19 square root found
19 PARAMETER sqrtval              =        4.843  answer
19 PARAMETER x

       i-1 11.728,    i-2  6.864,    i-3  5.141,    i-4  4.852
       i-5  4.843,    i-6  4.843,    i-7  4.843
```

In this example the user can see the use of the ABORT operator, which is used for fail-safe canceling of computation.

18.2 IF-ELSE Operator

The IF-ELSE operator is useful for transferring from one operator to another. In some cases, it can be written down as a set of $ conditions. The IF operator can be used for making GAMS code more understandable.

The optional "ELSE" part allows formulating the traditional construction "IF-THEN-ELSE". The following syntax is for the "IF-THEN-ELSE" operator:

```
if (condition,
        operators;
{condition ELSEIF, operators}
        [operators ELSE;]
);
```

Note, the braces and brackets are not required, but can be used. "Condition" means logical conditions described in the paragraph on conditional operations of GAMS.

Declaration or definition of equations can not be performed within the IF operator. Consider the following set of operators taken from the GAMS Language Guide:

```
p(i)$(f <= 0) = -1;
p(i)$(f >  0)  and (f < 1)) = p(i)**2;
p(i)$(f >  1) = p(i)**3;
q(j)$(f <= 0) = -1;
q(j)$(f >  0)  and (f < 1)) = q(j)**2;
q(j)$(f >  1) = q(j)**3;
```

the same can be expressed through the IF-ELSEIF-ELSE operators:

```
IF (f <= 0,
    p(i) = -1;
    q(j) = -1;
ELSEIF ((f > 0) AND (f < 1)),
    p(i) = p(i)**2;
    q(j) = q(j)**2;
ELSE
    p(i) = p(i)**3;
    q(j) = q(j)**3;
);
```

The body of the IF operator can contain the SOLVE operator. Consider the following example of the model from the GAMS Language Guide:

```
IF  (ml.MODELSTAT EQ 4),
*     the model ml is undefined;
*     change the conditions on the boundaries of x and solve it again;
      x.up(j) = 2*x.up(j) ;
      SOLVE ml USING LP MINIMIZING  lop;
ELSE IF (ml.MODELSTAT NE 1),
      ABORT "error occurred in solving the model ml";););
```

The user can see a new suffix ".MODELSTAT" here. Its meaning is quite transparent and its numerical value is relevant to situations raised in the process of solving the problem. A complete list of situations is given in the GAMS Language Guide.

18.3 WHILE Operator

WHILE operator is used for cycling the process of calculation before the execution of some logical condition. The syntax of the WHILE operator is the following:

```
while ((condition)),
                operators;
      );
```

As with the FOR operator, declaration and definition of equations can not be done in the WHILE operator. However, the WHILE operator can be used for control of the SOLVE operator. Consider the following part of a GAMS program that is randomly looking for the global optimum of a non-convex function. The example is taken from the GAMS Language Guide:

```
SCALAR  count;
count = 1;
SCALAR  globmin;
globmin = INF;
OPTION  BRATIO = 1;

WHILE ((count  LE  1000),
        x.l(j) = UNIFORM(0,1);
        SOLVE ml  USING  lp  MINIMIZING  obj;
        IF  (obj.l  LE  globmin,
                globmin = obj.l;
                globinit(j)  = x.l(j);
             );
        count = count +1;
);
```

In this example, the non-convex model is solved for 1000 randomly taken numbers, while the global solution is found through constant comparison.

Model [PRIME] from the GAMS model library illustrates the use of the WHILE operator in an example with simple generation of numbers less than 200.

18.4 FOR Operator

The FOR operator is used to repeat a bloc of operators. It has the following syntax:

```
FOR  (i = initial TO final [BY step],
       operators;
);
```

Note that *i* is not a set, but a numerical parameter. The step defines the growth of *i* after each repetition of the FOR loop. The *initial*, *final* and *step* values do not have to be integer-valued. The *initial* and *final* values can be positive or negative real numbers. The *step* must be a positive real number.

The FOR operator can be used to control the SOLVE operator. Consider the following example from the GAMS program randomly studying a non-convex function for finding a global optimum. The example is taken from the GAMS Language Guide:

```
SCALAR i;
SCALAR globmin;
globmin = inf;
OPTION  BRATIO = 1;

FOR (i = 1 TO 1000,
         x.l(j)  =  UNIFORM(0,1);
         SOLVE ml  USING  NLP  MINIMIZING  obj ;
         IF  (obj.l  LE globmin,
              globmin = obj.L ;
              globinit(j)  =  x.L(j);
         );
);
```

In this example, the non-convex function is examined for 1000 randomly taken points, while the global solution is found through comparison. The use of real numbers as initial and final values of the cycle and steps, is shown in the following example:

```
FOR (s  =  -18.4  TO  10.3  BY 6.1, DISPLAY s; );
```

Resulting printout of the file will contain the following lines:

```
      1 PARAMETER  S          =  -18.400
      2 PARAMETER  S          =  -12.300
      3 PARAMETER  S          =  -6.200
      4 PARAMETER  S          =  -0.100
      5 PARAMETER  S          =  +6.000
      6 PARAMETER  S          =  -12.100
```

Note, that S was increased by the GAMS compiler 6.1 during each cycle until it exceeded 10.3.

References

Brooke, A., D. Kendrick, A. Meeraus, and R. Raman, GAMS Language Guide, RELEASE 2.25, Version 92, GAMS Development Corporation, Washington D.C., 1997

Cuzan, A.G., *Appropriators Versus Expropriators: The Political Economy of Water in the West*, in Water Rights: Scarce Resource Allocation, Bureaucracy, and the Environment, T.L. Anderson (ed.), Pacific Institute for Public Policy Research, San Francisco, pp. 13-43, 1983.

Dinar, A., and J. Letey, Modeling Economic Management and Policy Issues of Water in Irrigated Agriculture, Praeger, Westport, 1996.

Field B.C., Environmental Economics: An Introduction, McGraw Hill Publishers, New York, 1994

Gardner, B.D., *Water Pricing and Rent Seeking in California Agriculture*, in Water Rights: Scarce Resource Allocation, Bureaucracy, and the Environment, T.L. Anderson (ed.), Pacific Institute for Public Policy Research, San Francisco, pp. 83-116, 1983.

Gibbons, D. C., The Economic Value of Water, Resources for the Future, Washington D.C., 1986.

Gisser, M., and R.N. Johnson, *Institutional Restrictions on the Transfer of Water Rights and the Survival of an Agency*, in Water Rights: Scarce Resource Allocation, Bureaucracy, and the Environment, T.L. Anderson (ed.), Pacific Institute for Public Policy Research, San Francisco, pp. 137-165, 1983.

Hirshleifer, J., J. C. De Haven, and J. W. Milliman, Water Supply: Economics, Technology, and Policy, Univ of Chicago Press, Chicago, 1960.

Howe, C.W., *Protecting Public Values under Tradable Water Permit Systems: Efficiency and Equity Considerations*, Environment and Behavior Program, Institute of Behavioral Science, University of Colorado, Boulder, Dec. 1996.

Howe, C.W., D.R. Schurmeier, and W.D. Shaw, Jr. Innovative Approaches to Water Allocation: The Potential for Water Markets, *Water Resources Research*, 22(4): 439-445, 1986.

Loucks, D.P., J.R. Stedinger, and D.A. Haith, Water Resource Systems Planning and Analysis, Prentice Hall, Englewood Cliffs, 1981.

McKinney, D.C. and X. Cai, *Multiobjective Water Resource Allocation Model for Toktogul Reservoir*, Technical Report, US Agency for International Development, Environmental Policy and Technology Project, Central Asia Regional EPT Office, Almaty, Kazakstan, April, 1997.

McKinney, D.C. and A.K. Kenshimov (eds.), *Optimization of the Use of Water and Energy*

Resources in the Syrdarya Basin Under Current Conditions, Technical Report, US Agency for International Development, Environmental Policies and Institutions for Central Asia (EPIC) Program, Almaty, Kazakstan, 2000.

Michelsen, A.M. and R.A. Young, Optioning agricultural water rights for urban water supplies during drought, *American Journal of Agricultural Economics* 75(11): 1010-1020, 1993.

Pearce, D. W., and R. K. Turner, Economics of natural Resources and the Environment, Johns Hopkins University Press, Baltimore, 1990.

Rosegrant, M.W., *Water resources in the twenty-first century: Challenges and implications for action,* Food, Agriculture, and the Environment Discussion Paper No. 20, Washington, D.C., IFPRI, 1997.

Rosegrant, M.W., and H.P. Binswanger, Markets in Tradable Water Rights: Potential for Efficiency Gains in Developing Country Water Resource Allocation, *World Development*, 22(11):1613-1625, 1994.

Rosegrant, M.W., R. Gazmuri Schleyer, and S.N. Yadav, Water Policy for Efficient Agricultural Diversification: Market Based Approaches, *Food Policy*, 20(3):203-223, 1995.

Rosegrant, M.W., C. Ringler, D.C. McKinney, X. Cai, A. Keller, and G. Donoso, Integrated economic-hydrologic water modeling at the basin scale: the Maipo river basin. *Agricultural Economics*, Vol.24, No.1, pp.33-46, 2000.

Savitsky A.G. *Optimal control of water, land and hydropower resources in Central Asia region by modern modelling technologies help*. Material ICID, Portigal, Lisbon, September 1998. (13 p.)

TWDB, Texas Water Development Board, *Water for Texas: Today and Tomorrow*, Austin, Texas, 1997.

Tregarthen, T.D., *Water in Colorado: Fear and Loathing in the Marketplace*, in Water Rights: Scarce Resource Allocation, Bureaucracy, and the Environment, T.L. Anderson (ed.), Pacific Institute for Public Policy Research, San Francisco, pp. 119-136, 1983.

Appendix A - GAMS Installation Instructions

SYSTEM REQUIREMENTS

To install the system you need approximately 20 MB of free disk space. *GAMS* dynamically allocates the memory (RAM) needed to generate and execute the models. At least 32 MB of RAM is recommended to run models of reasonable size. Huge models may need much more memory. A math Coprocessor is required and we recommend using a Pentium or higher Intel compatible chip. To install the Windows version, you must be running Windows 95 or higher or Windows NT 3.51 or higher. Windows 3.1 users must use the DOS version of *GAMS*.

INSTALLATION

1. Run `setup.exe`: If you are installing from diskettes, you can run `setup.exe` from the first system diskette; you will be prompted to insert the remaining diskettes in turn. You can also copy all the diskette files to a scratch directory on the hard disk and run `setup.exe` from there. If you are instlling from CD, you will run the file `\systems\win\setup.exe` directly from the *GAMS* distribution CD. The setup program will first prompt you for the name of the directory in which to install *GAMS*. We call this directory the `` `GAMS directory' ``. You may accept the default choice or pick another directory. Please remember that if you want to install two different versions of *GAMS*, they should be in separate directories (for example `c:\gams` and `c:\gams_old`). The *GAMS* software will be installed in the directory you choose.
2. Copy the license file: If no license file is found in the *GAMS* directory, you will be prompted for one during the installation. If you are not sure if you have a license file, or do not have it yet, choose `No' when asked if you wish to copy a license file. You can always do this later. If no valid license file is found, *GAMS* will still function in demonstration mode and will only solve small problems. For example, all demonstration and student systems are shipped without a license. If you have a license file that you wish to copy to the *GAMS* directory at this time, answer `` `Yes' `` to the license file prompt. You will now be given the opportunity to browse the file system and find the directory containing the license file you wish to copy. For example, if your system came with a license file on a diskette, browse to the `A:` drive. When you have found the correct directory, choose `` `Copy license file' `` to perform the copy.
3. Choose default solvers: Run the *GAMS* IDE by double-clicking `gamside.exe` from the *GAMS* directory. To view or edit the default solvers, choose File → Options → Solvers from the IDE. You can accept the existing defaults if you wish, but most users will want to pick default solvers for each model type. It is a good idea to review the solver defaults when installing a new *GAMS* system or when updating a license file.
4.

Run a few models to test the *GAMS* system: The online help for the IDE (`Help → Help Topics → Guided Tour`) describes how to copy a model from the *GAMS* model library, run it, and view the solution. To test your installation, run the following models from the *GAMS* model library:

```
LP:    trnsport (objective value:  153.675)

NLP:   chenery  (objective value:  1058.9)

MIP:   bid      (optimal solution: 15210109.512)

MINLP: procsel  (optimal solution: 1.9231)

MCP:   scarfmcp (no objective function)

MPSGE: scarfmge (no objective function)
```

If there are any problems during these test runs, read the section on Troubleshooting.

COMMAND LINE INSTALLATION

Users wishing to use *GAMS* from the command line (i.e. in console mode) may want to perform the following steps after they have installed the system as described above. These steps are not necessary to run *GAMS* via the IDE.

1.

Run the program `gamsinst`: `gamsinst` is a command-line program used to install and configure *GAMS*. It prompts the user for default solvers to be used for each model type. If possible choose solvers you have licensed since unlicensed solvers will only run in demonstration mode. These solver defaults can be changed by:

(a) rerunning `gamsinst` and resetting the default values
(b) setting a command line default, e.g. `gams trsnport lp=bdmlp`
(c) by an option statement in the *GAMS* model, e.g: `option lp=bdmlp`

If you have to support different operating systems from the same installation, please use `` `gamsinst -sys all' `` A complete log of the installation is stored in `gamsinst.log`.

N.B. The system-wide solver defaults are shared by the command-line and GUI versions of *GAMS*, so you can also choose these defaults using *GAMS* IDE.

2.

Add the *GAMS* directory to your path. To avoid having to type in an absolute path name each time you run *GAMS*, we recommend adding the *GAMS* directory to your `PATH` when using the console-mode (not the *GAMS* IDE) version of *GAMS*. In case more than one *GAMS* system is installed on the machine, then separate paths have to be set before invoking each version. Under Windows 95/98, this change requires editing the `autoexec.bat` file and adding the *GAMS* directory to the path, as the following example illustrates:

```
path=your\current\path\setting

path=%PATH%;c:\gams
```

Under Windows NT the following procedure must be applied to add the *GAMS* directory to your path,

- Open the System Properties under the Control Panel.
- On the Environment tab click on the existing variable `PATH`
- In the Value box, add the *GAMS* directory to the path, as the following example illustrates: `c:\your\current\path\setting;C:\gams`
- Click Set.

Note for CPLEX Users:
The *GAMS* installation program will not alter any existing CPLEX license. To enable or change a CPLEX license, you need to run a CPLEX specific license program after completing the standard *GAMS* installation. Please refer to the annex on licensing in the CPLEX section of the Solver Manual for details.

NETWORK SUPPORT

GAMS 2.50 supports the Uniform Naming Convention (UNC) both for the system and for the solvers. *GAMS* can be installed and accessed over a network of PC's. For example: *GAMS* has been installed on a machine called `bach` in the directory `gams`. Now you are sitting in front of another machine and want to run a model (`trnsport.gms`), which is located on drive `d:\`. To run this model either enter (on a command line):

```
>\\bach\gams\gams d:\trnsport
```

or just (if the path to the remote machine is in your PATH-statement)

```
>gams trnsport
```

This also works with models on shared drives. For example, with the above mentioned installation on `\\ bach\gams` you may want to execute the model `x.gms` which is located on `\\bach\public` but some include files are sitting on another machine: `\\berg\ includes`. The *GAMS* call would be:

```
>\\bach\gams\gams \\bach\public\x idir \\berg\includes
```

TROUBLE-SHOOTING

GAMS rarely causes problems, and most of the problems that occur can be corrected quite easily by following the notes in this section.

Insufficient memory

When *GAMS* produces an output like this one:

```
--- TEST.GMS(4) 671 Kb

*** Out of Memory *** . . .
```

GAMS itself is running out of memory. If the memory installed on your machine is not sufficient for your problems, try to run the model on a machine with more memory. Note that using virtual memory will allow you to solve larger problems than possible using real memory alone. However, using virtual memory will slow down the *GAMS* system.

Environment space

When running the console-mode version of *GAMS*, it may under some circumstances be necessary to adjust the settings for the initial environment to 4096. In order to change this under Windows 95/98, click the right mouse button on the title bar of MS-DOS Prompt Window and select Properties. Open the *Memory* tab and change the value of the `Initial environment'- box to 4096.

Other technical problems:

For other technical questions like insufficient disk space, CPLEX licensing problems or solver failures please check the *GAMS* web sites or contact our support staff directly.

USING GAMS - IDE

A tutorial on how to use the GAMS – IDE has been developed by Dr. Bruce McCarl and it is available at www.gams.com/mccarl/useide.pdf.

About the Authors

Daene McKinney is an Associate Professor in the Environmental and Water Resources Engineering program of the Department of Civil Engineering at The University of Texas at Austin. During 1998-2000, he was Team Leader and Senior Environmental Policy Advisor for the USAID Environmental Policy and Institutions for Central Asia (EPIC) Program in Almaty, Kazakhastan. During that time the work resulting in the current document was initiated. Dr. McKinney earned his Ph.D. in Civil Engineering with a major in Water Resources Engineering at Cornell University in 1990. After working for the US EPA for one year as an environmental engineer and hydrogeologist, he joined the faculty at the University of Texas. Dr. McKinney has served as an Associate Editor of Operations Research and the Journal of Water Resource Planning and Management. He has served as the Chair of the ASCE Water Resource Systems Committee, and as a member of the Board of Directors and the International Committee of the Universities Council on Water Resources (UCOWR). Dr. McKinney teaches undergraduate courses in Fluid Mechanics, Numerical Methods, Hydrology, and graduate courses in Water Resources Planning and Management, Transboundary Water Resources and Numerical Methods for Environmental Engineers. Dr. McKinney's research interests include developing and applying numerical methods for simulation, optimization, and uncertainty analysis of water resources management problems, and the development of laboratory and field experimental techniques for the characterization and remediation of aquifer and groundwater contamination.

Andrey Savitsky's has been a Lead Specialist in the Water Resources Department of the Scientific Information Center of the Interstate Coordination Water Commission (SIC ICWC). During 1998 – 2000 he was a Water Resources Advisor for the USAID Environmental Policy and Institutions for Central Asia (EPIC) Program in Tashkent, Uzbekistan. He was responsible for mathematical modeling of optimal water and energy resources management within the EPIC Program. In 1986, he earned his degree of Candidate of Technical Sciences with a major in Hydraulics and Hydrological Engineering on the subject of Water and Salt Regimes of Reservoirs of the Central Asia. At the beginning of his scientific career he worked in the Hydroaerodynamics Laboratory at the Institute of Mechanics and Antiseismic Constructions for the Academy of Sciences of the Uzbek Soviet Socialist Republic. He was involved in studying dynamics of multiphase environments in industry (heat exchange systems in reactors; manufacturing of experimental samples for cumulative, non-stationary, high temperature flows of gas; mathematical modeling of multiphase interpenetrating motions of continuum). Then he joined the Scientific and Research Institute of Irrigation where he worked for a long period of time. The majority of his works and achievements are connected with mathematical simulation of dynamic and cinematic processes in rivers and reservoirs, dams in the process of demolition, in Soil – Plant – Atmosphere systems in agricultural cycles. In SIC ICWC, Savitsky's work is connected with optimization of water resources and development software programs for users. Savitsky's research interests include developing and applying mathematical methods for simulation, optimization of management of natural and artificial phenomena.