

# TOMLAB Models

Marcus M. Edvall<sup>1</sup>, Anders Göran<sup>2</sup>, Kenneth Holmström<sup>3</sup>, and Per Strandberg<sup>4</sup>

November 6, 2006



---

<sup>1</sup>Tomlab Optimization Inc., 855 Beech St #121, San Diego, CA, USA, [medvall@tomopt.com](mailto:medvall@tomopt.com).

<sup>2</sup>Tomlab Optimization AB, Västerås Technology Park, Trefasgatan 4, SE-721 30 Västerås, Sweden, [anders@tomopt.com](mailto:anders@tomopt.com).

<sup>3</sup>Professor in Optimization, Mälardalen University, Department of Mathematics and Physics, P.O. Box 883, SE-721 23 Västerås, Sweden, [kenneth.holmstrom@mdh.se](mailto:kenneth.holmstrom@mdh.se).

<sup>4</sup>Tomlab Optimization AB, Västerås Technology Park, Trefasgatan 4, SE-721 30 Västerås, Sweden, [per.strandberg@tomopt.com](mailto:per.strandberg@tomopt.com).

## Introduction to TOMLAB Models

The TOMLAB Problem bundle is a collection of problems. These examples can be used as templates for customer specific problems as well as solver benchmarking.

Part I, `testprob` contains 750+ problems showing the broad spectrum of problems available to the TOMLAB solvers. The source code is available after installing the TOMLAB distribution (`tomlab/testprob`).

Part II, `modellib`, is a model library where the problems are described in a human-readable format with words, tables and figures. Each problem has been translated into the standard TOMLAB Prob-format. The problems in this part are mainly of the Linear Programming and Mixed Integer Programming classes with applications in the fields of Mining and Processing, Scheduling, Planning, Loading and Cutting, Ground transport, Air transport, Telecommunication, Economics, Timetabling and Public Services. The source code is available in `tomlab/modellib`.

Part III, references to additional downloads from the TOMLAB downloads page: <http://tomopt.com/tomlab/download/manuals.p>

# Contents

Contents	3
<b>I The testprob collection</b>	<b>7</b>
1 Introduction to testprob	7
2 Linear Programming Problems: lp_prob	8
3 Mixed-Integer Linear Programming Problems: mip_prob	10
4 Quadratic Programming Problems: qp_prob	12
5 Mixed-Integer Quadratic Programming Problems: mipq_prob	14
6 Mixed-Integer Quadratic Programming Problems with Quadratic Constraints: miqq_prob	16
7 Nonlinear Programming Problems: con_prob and chs_prob	18
7.1 An example of a nonlinear problem . . . . .	18
7.2 con_prob . . . . .	19
7.3 chs_prob . . . . .	19
8 Mixed-Integer Nonlinear Programming Problems: minlp_prob.	20
9 Linear Least Squares Problems: lls_prob	22
10 (Constrained) Nonlinear Least Squares Problems: cls_prob, mgh_prob and ls_prob	24
10.1 An example of a (constrained) nonlinear least squares . . . . .	24
10.2 cls_prob . . . . .	25
10.3 mgh_prob . . . . .	25
10.4 ls_prob . . . . .	25
11 Global Optimization Problems: glb_prob, lgo1_prob, lgo2_prob and gkls_prob.	26
11.1 Example of a global optimization problem . . . . .	26
11.2 glb_prob . . . . .	27
11.3 lgo1_prob . . . . .	27
11.4 lgo2_prob . . . . .	27
11.5 gkls_prob . . . . .	27

<b>12</b>	<b>Constrained Global Optimization Problems: glc_prob</b>	<b>28</b>
<b>13</b>	<b>Unconstrained Optimization: uc_prob and uhs_prob.</b>	<b>30</b>
13.1	An example of a unconstrained optimization problem (with simple bounds) . . . . .	30
13.2	uc_prob . . . . .	30
13.3	uhs_prob . . . . .	30
<b>14</b>	<b>Linear Semi-Definite Programming Problem with Linear Matrix Inequalities: sdp_prob</b>	<b>31</b>
<b>15</b>	<b>Linear Semi-Definite Programming Problem with Bilinear Matrix Inequalities: bmi_prob</b>	<b>33</b>
<b>16</b>	<b>Constrained Goal Attainment Problems: goals_prob and mco_prob</b>	<b>35</b>
16.1	An example of a constrained goal attainment problems . . . . .	35
16.2	mco_prob . . . . .	36
16.3	goals_prob . . . . .	36
<b>17</b>	<b>Geometric programming problems: gp_prob</b>	<b>37</b>
<b>18</b>	<b>Fitting of positive sums of Exponentials: exp_prob</b>	<b>38</b>
<b>II</b>	<b>The modelib collection</b>	<b>39</b>
<b>19</b>	<b>Introduction to modelib</b>	<b>39</b>
<b>20</b>	<b>Mining and Processing</b>	<b>40</b>
20.1	Production of alloys . . . . .	40
20.2	Animal food Production . . . . .	43
20.3	Refinery . . . . .	47
20.4	Cane sugar Production . . . . .	52
20.5	Opencast mining . . . . .	55
20.6	Production of Electricity . . . . .	58
<b>21</b>	<b>Scheduling</b>	<b>61</b>
21.1	Construction of a Stadium 1 . . . . .	61
21.2	Construction of a Stadium 2 . . . . .	65
21.3	Flow-shop scheduling . . . . .	70
21.4	Job Shop Scheduling . . . . .	74
21.5	Sequencing jobs on a bottleneck machine . . . . .	77
21.6	Paint production . . . . .	80

21.7 Assembly line balancing . . . . .	83
<b>22 Planning</b>	<b>87</b>
22.1 Planning the production of bicycles . . . . .	87
22.2 Production of drinking glasses . . . . .	90
22.3 Material Requirement Planning . . . . .	94
22.4 Planning the production of electronic components . . . . .	98
22.5 Planning the Production of Fiberglass . . . . .	101
22.6 Assignment of production batches to machines . . . . .	104
<b>23 Loading and Cutting</b>	<b>107</b>
23.1 Wagon load balancing . . . . .	107
23.2 Barge loading . . . . .	109
23.3 Tank loading . . . . .	112
23.4 Backing up files . . . . .	114
23.5 Cutting sheet metal . . . . .	116
23.6 Cutting steel bars for desk legs . . . . .	118
<b>24 Ground transport</b>	<b>120</b>
24.1 Car rental . . . . .	120
24.2 Choosing the mode of transport . . . . .	123
24.3 Depot location . . . . .	126
24.4 Heating oil delivery . . . . .	130
24.5 Combining different modes of transport . . . . .	133
24.6 Fleet planning for vans . . . . .	136
<b>25 Air transport</b>	<b>139</b>
25.1 Flight connections at a hub . . . . .	139
25.2 Composing flight crews . . . . .	142
25.3 Scheduling flight landings . . . . .	146
25.4 Airline hub location . . . . .	150
25.5 Planning a flight tour . . . . .	154
<b>26 Telecommunication</b>	<b>157</b>
26.1 Network reliability . . . . .	157
26.2 Dimensioning of a mobile phone network . . . . .	160
26.3 Routing telephone calls . . . . .	164

26.4 Construction of a cabled network . . . . .	169
26.5 Scheduling of telecommunications via satellite . . . . .	171
26.6 Location of GSM transmitters . . . . .	175
<b>27 Economics</b>	<b>178</b>
27.1 Choice of loans . . . . .	178
27.2 Publicity campaign . . . . .	181
27.3 Portfolio selection . . . . .	184
27.4 Financing an early retirement . . . . .	187
27.5 Family budget . . . . .	190
27.6 Choice of expansion projects . . . . .	192
27.7 Mean variance portfolio selection . . . . .	195
<b>28 Timetabling</b>	<b>198</b>
28.1 Assigning personnel to machines . . . . .	198
28.2 Scheduling nurses . . . . .	201
28.3 Establishing a college timetable . . . . .	205
28.4 Exam scheduling . . . . .	208
28.5 Production planning with personnel assignment . . . . .	211
28.6 Plan the personnel at a construction site . . . . .	215
<b>29 Public Services</b>	<b>218</b>
29.1 Water conveyance . . . . .	218
29.2 CCTV surveillance . . . . .	221
29.3 Rigging elections . . . . .	224
29.4 Gritting roads . . . . .	228
29.5 Location of income tax offices . . . . .	231
29.6 Efficiency of hospitals . . . . .	234
<b>III Additional downloads: lp_prob, milp_prob and qp_prob.</b>	<b>237</b>
<b>30 Introduction to additional downloads</b>	<b>237</b>
<b>31 Additional linear programming test problems: lp_prob.</b>	<b>237</b>
<b>32 Additional mixed-integer linear programming test problems: milp_prob.</b>	<b>237</b>
<b>33 Additional quadratic programming test problems: qp_prob.</b>	<b>237</b>

## Part I

# The testprob collection

## 1 Introduction to testprob

Each of the sections in this part will demonstrate a problem class. Most sections also have a small example in Matlab code with a formal mathematical problem description.

The functions `probInit` and `tomrun` respectively initializes and solves the problems in Matlab. For example:

```
Prob = probInit('gp_prob',1);  
tomRun('',Prob)
```

## 2 Linear Programming Problems: lp\_prob

In `lp_prob` there are 54 linear programming test problems with sizes to nearly 700 variables and 500 constraints. In order to define the 22'nd problem and solve it execute the following in Matlab:

```
n      = 22;
Prob   = probInit('lp_prob',n);
Result = tomRun('',Prob);
```

The general formulation in TOMLAB for a linear programming problem is:

$$\begin{aligned} \min_x \quad & f(x) = c^T x \\ \text{s/t} \quad & x_L \leq x \leq x_U \\ & b_L \leq Ax \leq b_U \end{aligned} \tag{1}$$

where  $c, x, x_L, x_U \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m_1 \times n}$ , and  $b_L, b_U \in \mathbb{R}^{m_1}$ . Equality constraints are defined by setting the lower bound to the upper bound.

An example of a problem of this class, (that is also found in the TOMLAB Quickguide) is lpQG:

$$\begin{aligned} \min_{x_1, x_2} \quad & f(x_1, x_2) = -7x_1 - 5x_2 \\ \text{s/t} \quad & x_1 + 2x_2 \leq 6 \\ & 4x_1 + x_2 \leq 12 \\ & x_1, x_2 \geq 0 \end{aligned} \tag{2}$$

**File:** tomlab/quickguide/lpQG.m

```
% lpQG is a small example problem for defining and solving
% linear programming problems using the TOMLAB format.
```

```
Name = 'lpQG';      % Problem name, not required.
c     = [-7 -5]';   % Coefficients in linear objective function
A     = [ 1  2
         4  1];    % Matrix defining linear constraints
b_U   = [ 6 12]';  % Upper bounds on the linear inequalities
x_L   = [ 0  0]';  % Lower bounds on x
```

```
% x_min and x_max are only needed if doing plots
x_min = [ 0  0]';
x_max = [10 10]';
```

```
% b_L, x_U and x_0 have default values and need not be defined.
% It is possible to call lpAssign with empty [] arguments instead
b_L   = [-inf -inf]';
x_U   = [];
x_0   = [];
```

```

% Assign routine for defining an LP problem. This allows the user
% to try any solver, including general nonlinear solvers.
Prob = lpAssign(c, A, b_L, b_U, x_L, x_U, x_0, Name,...
               [], [], [], x_min, x_max, [], []);

% Calling driver routine tomRun to run the solver.
% The 1 sets the print level after optimization.

% Result.x_k contains the optimal decision variables.
% Result.f_k is the optimal value.

Result = tomRun('pdco', Prob, 1);

%Result = tomRun('lpSimplex', Prob, 1);
%Result = tomRun('minos', Prob, 1);
%Result = tomRun('snopt', Prob, 1);
%Result = tomRun('conopt', Prob, 1);
%Result = tomRun('knitro', Prob, 1);
%Result = tomRun('cplex', Prob, 1);
%Result = tomRun('xpress-mp', Prob, 1);

```

### 3 Mixed-Integer Linear Programming Problems: mip\_prob

In `mip_prob` there are 47 mixed-integer linear test problems with sizes to nearly 1100 variables and nearly 1200 constraints. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('mip_prob',n);
Result = tomRun('',Prob);
```

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `mipQG`:

$$\begin{aligned} \min_x \quad & f(x) = c^T x \\ \text{s/t} \quad & x_L \leq x \leq x_U, \\ & b_L \leq Ax \leq b_U, x_j \in \mathbb{N} \quad \forall j \in I \end{aligned} \tag{3}$$

where  $c, x, x_L, x_U \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m_1 \times n}$ , and  $b_L, b_U \in \mathbb{R}^{m_1}$ . The variables  $x \in I$ , the index subset of  $1, \dots, n$  are restricted to be integers.

**File:** `tomlab/quickguide/mipQG.m`

```
% mipQG is a small example problem for defining and solving
% mixed-integer linear programming problems using the TOMLAB format.

Name='Weingartner 1 - 2/28 0-1 knapsack';
% Problem formulated as a minimum problem
A = [ 45    0    85    150    65    95    30    0    170  0 ...
      40    25    20     0     0    25     0     0     25  0 ...
      165   0    85     0     0     0     0    100 ; ...
      30    20   125     5    80    25    35    73    12  15 ...
      15    40     5    10    10    12    10     9     0  20 ...
      60    40    50    36    49    40    19    150];
b_U = [600;600]; % 2 knapsack capacities
c = [1898 440 22507 270 14148 3100 4650 30800 615 4975 ...
     1160 4225 510 11880 479 440 490 330 110 560 ...
     24355 2885 11748 4550 750 3720 1950 10500]'; % 28 weights

% Make problem on standard form for mipSolve
[m,n] = size(A);
c = -c; % Change sign to make a minimum problem
x_L = zeros(n,1);
x_U = ones(n,1);
x_0 = zeros(n,1);

fprintf('Knapsack problem. Variables %d. Knapsacks %d\n',n,m);

% All original variables should be integer
IntVars = n; % Could also be set as: IntVars=1:n; or IntVars=ones(n,1);
x_min = x_L; x_max = x_U; f_Low = -1E7; % f_Low <= f_optimal must hold
```

```

b_L      = -inf*ones(2,1);
f_opt    = -141278;

nProblem = []; % Problem number not used
fIP      = []; % Do not use any prior knowledge
xIP      = []; % Do not use any prior knowledge
setupFile = []; % Just define the Prob structure, not any permanent setup file
x_opt    = []; % The optimal integer solution is not known
VarWeight = []; % No variable priorities, largest fractional part will be used
KNAPSACK = 1; % Run with the knapsack heuristic

% Assign routine for defining a MIP problem.
Prob      = mipAssign(c, A, b_L, b_U, x_L, x_U, x_0, Name, setupFile, ...
                    nProblem, IntVars, VarWeight, KNAPSACK, fIP, xIP, ...
                    f_Low, x_min, x_max, f_opt, x_opt);

Prob.optParam.IterPrint = 0; % Set to 1 to see iterations.
Prob.Solver.Alg = 2; % Depth First, then Breadth search

% Calling driver routine tomRun to run the solver.
% The 1 sets the print level after optimization.

Result = tomRun('mipSolve', Prob, 1);
%Result = tomRun('cplex', Prob, 1);
%Result = tomRun('xpress-mp', Prob, 1);
%Result = tomRun('miqpBB', Prob, 1);
%Result = tomRun('minlpBB', Prob, 1);

```

## 4 Quadratic Programming Problems: qp\_prob

In `qp_prob` there are 41 quadratic programming test problems with sizes to nearly 1200 variables and nearly 500 constraints. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('qp_prob',n);
Result = tomRun('',Prob);
```

The basic structure of a general nonlinear Quadratic Programming problem is:

$$\begin{aligned} \min_x \quad & f(x) = \frac{1}{2}x^T Fx + c^T x \\ \text{s/t} \quad & \begin{array}{l} x_L \leq x \leq x_U \\ b_L \leq Ax \leq b_U \end{array} \end{aligned} \tag{4}$$

where  $c, x, x_L, x_U \in \mathbb{R}^n$ ,  $F \in \mathbb{R}^{n \times n}$ ,  $A \in \mathbb{R}^{m_1 \times n}$ , and  $b_L, b_U \in \mathbb{R}^{m_1}$ . Equality constraints are defined by setting the lower bound equal to the upper bound, i.e. for constraint  $i$ :  $b_L(i) = b_U(i)$ . Fixed variables are handled the same way.

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `qpQG`:

$$\begin{aligned} \min_x \quad & f(x) = 4x_1^2 + x_1x_2 + 4x_2^2 + 3x_1 - 4x_2 \\ \text{s/t} \quad & \begin{array}{l} x_1 + x_2 \leq 5 \\ x_1 - x_2 = 0 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{array} \end{aligned} \tag{5}$$

**File:** `tomlab/quickguide/qpQG.m`

Open the file for viewing, and execute `qpQG` in Matlab.

```
% qpQG is a small example problem for defining and solving
% quadratic programming problems using the TOMLAB format.
```

```
Name = 'QP Example';
F = [ 8 1 % Matrix F in 1/2 * x' * F * x + c' * x
     1 8 ];
c = [ 3 -4 ]'; % Vector c in 1/2 * x' * F * x + c' * x
A = [ 1 1 % Constraint matrix
     1 -1 ];
b_L = [-inf 0 ]'; % Lower bounds on the linear constraints
b_U = [ 5 0 ]'; % Upper bounds on the linear constraints
x_L = [ 0 0 ]'; % Lower bounds on the variables
x_U = [ inf inf ]'; % Upper bounds on the variables
x_0 = [ 0 1 ]'; % Starting point
x_min = [-1 -1 ]'; % Plot region lower bound parameters
x_max = [ 6 6 ]'; % Plot region upper bound parameters
```

```
% Assign routine for defining a QP problem.
Prob = qpAssign(F, c, A, b_L, b_U, x_L, x_U, x_0, Name,...
    [], [], [], x_min, x_max);

% Calling driver routine tomRun to run the solver.
% The 1 sets the print level after optimization.

Result = tomRun('qpSolve', Prob, 1);
%Result = tomRun('snopt', Prob, 1);
%Result = tomRun('sqopt', Prob, 1);
%Result = tomRun('cplex', Prob, 1);
%Result = tomRun('knitro', Prob, 1);
%Result = tomRun('conopt', Prob, 1);
```

## 5 Mixed-Integer Quadratic Programming Problems: mipq\_prob

In `mipq_prob` there are 4 mixed-integer quadratic programming test problems with sizes to about 120 variables and slightly more than 100 constraints. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('mipq_prob',n);
Result = tomRun('',Prob);
```

The basic structure of a general mixed-integer quadratic programming problem is:

$$\begin{aligned} \min_x \quad & f(x) = \frac{1}{2}x^T Fx + c^T x \\ \text{s/t} \quad & x_L \leq x \leq x_U, \\ & b_L \leq Ax \leq b_U, \quad x_j \in \mathbb{N} \quad \forall j \in I \end{aligned} \tag{6}$$

where  $c, x, x_L, x_U \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m_1 \times n}$ , and  $b_L, b_U \in \mathbb{R}^{m_1}$ . The variables  $x \in I$ , the index subset of  $1, \dots, n$  are restricted to be integers.

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `mipqQG`:

**File:** `tomlab/quickguide/mipqQG.m`

```
% mipqQG is a small example problem for defining and solving
% mixed-integer quadratic programming problems using the TOMLAB format.
```

```
c = [-6 0]';
Name = 'XP Ref Manual MIQP';
F = [4 -2; -2 4];
A = [1 1];
b_L = -Inf;
b_U = 1.9;
x_L = [0 0]';
x_U = [Inf Inf]';

% Defining first variable as an integer
IntVars = 1;

% Assign routine for defining a MIQP problem.
Prob = mipqAssign(F, c, A, b_L, b_U, x_L, x_U, [], ...
    IntVars, [], [], [], Name, [], []);

% Calling driver routine tomRun to run the solver.
% The 1 sets the print level after optimization.

Result = tomRun('cplex', Prob, 1);
%Result = tomRun('oqnlp', Prob, 1);
%Result = tomRun('mipqBB', Prob, 1);
%Result = tomRun('xpress-mp', Prob, 1);
```

```
%Result = tomRun('minlpBB', Prob, 1);
```

## 6 Mixed-Integer Quadratic Programming Problems with Quadratic Constraints: miqq\_prob

In `miqq_prob` there are 14 mixed-integer quadratic programming test problems with quadratic constraints with sizes to 10 variables and 8 constraints. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('miqq_prob',n);
Result = tomRun('',Prob);
```

The basic structure of a general mixed-integer quadratic programming problem with quadratic constraints is:

$$\begin{aligned}
 \min_x \quad & f(x) = \frac{1}{2}x^T Fx + c^T x \\
 s/t \quad & x_L \leq x \leq x_U \\
 & b_L \leq Ax \leq b_U \\
 & x^T Q^{(i)} x + a^{(i)T} x \leq r_U^{(i)}, \quad i = 1, \dots, n_{qc} \\
 & x_i \text{ integer} \quad i \in I
 \end{aligned} \tag{7}$$

where  $c, x, x_L, x_U, a^{(i)} \in \mathbb{R}^n$ ,  $F, Q^{(i)} \in \mathbb{R}^{n \times n}$ ,  $A \in \mathbb{R}^{m \times n}$  and  $b_L, b_U \in \mathbb{R}^m$ .  $r_U^{(i)}$  is a scalar. The variables  $x \in I$ , the index subset of  $1, \dots, n$ , are restricted to be integers.

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `miqqQG`:

**File:** `tomlab/quickguide/miqqQG.m`

```
% miqqQG is a small example problem for defining and solving
% mixed-integer quadratic programming problems with quadratic constraints
% using the TOMLAB format.
```

```
Name = 'MIQQ Test Problem 1';
f_Low = -1E5;
x_opt = [];
f_opt = [];
IntVars = [0 0 1];
```

```
F = [2 0 0;0 2 0;0 0 2];
A = [1 2 -1;1 -1 1];
b_L = [4 -2]';
b_U = b_L;
c = zeros(3,1);
```

```
x_0 = [0 0 0]';
x_L = [-10 -10 -10]';
x_U = [10 10 10]';
x_min = [0 0 -1]';
x_max = [2 2 1]';
```

```
% Adding quadratic constraints
```

```
qc(1).Q = speye(3,3);
qc(1).a = zeros(3,1);
qc(1).r_U = 3;

qc(2).Q = speye(3,3);
qc(2).a = zeros(3,1);
qc(2).r_U = 5;

Prob = miqqAssign(F, c, A, b_L, b_U, x_L, x_U, x_0, qc,...
    IntVars, [], [], [],...
    Name, [], [],...
    x_min, x_max, f_opt, x_opt);

Result = tomRun('cplex', Prob, 1);
% Result = tomRun('minlpBB', Prob, 1);
```

## 7 Nonlinear Programming Problems: con\_prob and chs\_prob

The TOMLAB bundle `testprob` provides three sets of test problems for nonlinear problems: `con_prob` and `chs_prob`.

### 7.1 An example of a nonlinear problem

The basic structure of a general nonlinear problem is the following

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s/t} \quad & x_L \leq x \leq x_U \\ & b_L \leq Ax \leq b_U \\ & c_L \leq c(x) \leq c_U \end{aligned} \tag{8}$$

where  $x, x_L, x_U \in \mathbb{R}^n$ ,  $f(x) \in \mathbb{R}$ ,  $A \in \mathbb{R}^{m_1 \times n}$ ,  $b_L, b_U \in \mathbb{R}^{m_1}$  and  $c_L, c(x), c_U \in \mathbb{R}^{m_2}$ .

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `nlpQG`:

$$\begin{aligned} \min_x \quad & f(x) = \alpha(x_2 - x_1^2)^2 + (1 - x_1)^2 \\ \text{s/t} \quad & -10 \leq x_1 \leq 2 \\ & -10 \leq x_2 \leq 2 \\ & \alpha = 100 \end{aligned} \tag{9}$$

TOMLAB requires that general nonlinear problems are defined in Matlab m-files. The function to be optimized must always be supplied. It is recommended that the user supply as many analytical functions as possible. There are six methods available for numerical differentiation and also two for automatic.

The following files define the problem in TOMLAB.

**File:** `tomlab/quickguide/rbbQG.f.m`, `rbbQG.g.m`, `rbbQG.H.m`, `rbbQG.c.m`, `rbbQG_dc.m`, `rbbQG_d2c.m`

```
f: Function value
g: Gradient vector
H: Hessian matrix
c: Nonlinear constraint vector
dc: Nonlinear constraint gradient matrix
d2c: The second part of the Hessian to the Lagrangian
function for the nonlinear constraints.
```

The following file illustrates how to solve this NLP (CON) problem in TOMLAB. Also view the m-files specified above for more information.

**File:** `tomlab/quickguide/nlpQG.m`

Open the file for viewing, and execute `nlpQG` in Matlab.

```
% nlpQG is a small example problem for defining and solving
% nonlinear programming problems using the TOMLAB format.
```

```

Name = 'RBB Problem';
x_0 = [-1.2 1]';      % Starting values for the optimization.
x_L = [-10;-10];     % Lower bounds for x.
x_U = [2;2];         % Upper bounds for x.
fLowBnd = 0;         % Lower bound on function.

c_L = -1000;         % Lower bound on nonlinear constraints.
c_U = 0;             % Upper bound on nonlinear constraints.

Prob = conAssign('rbbQG_f', 'rbbQG_g', 'rbbQG_H', [], x_L, x_U, Name, x_0,...
    [], fLowBnd, [], [], [], 'rbbQG_c', 'rbbQG_dc', 'rbbQG_d2c', [], c_L, c_U);

Prob.Warning = 0;    % Turning off warnings.

Result = tomRun('ucSolve', Prob, 1); % Ignores constraints.

% Result = tomRun('conopt', Prob, 1);
% Result = tomRun('snopt', Prob, 1);

```

## 7.2 con\_prob

con\_prob is a collection of 17 constrained nonlinear test problems with 2 to 100 variables and up to 50 constraints. In order to define the problem n and solve it execute the following in Matlab:

```

Prob = probInit('con_prob',n);
Result = tomRun('',Prob);

```

## 7.3 chs\_prob

chs\_prob is a collection of 180 constrained nonlinear test problems from the Hoch-Schittkowski set with 2 to 50 variables and about 10 constraints. In order to define the problem n and solve it execute the following in Matlab:

```

Prob = probInit('chs_prob',n);
Result = tomRun('',Prob);

```

## 8 Mixed-Integer Nonlinear Programming Problems: minlp\_prob.

In `minlp_prob` there are 14 mixed-integer nonlinear programming test problems with sizes to nearly 50 variables and nearly 50 constraints. In order to define problem number `n` and solve it execute the following in Matlab:

```
Prob = probInit('minlp_prob',n);
Result = tomRun('',Prob);
```

The basic structure of a general mixed-integer nonlinear programming problem is the following

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s/t} \quad & -\infty < x_L \leq x \leq x_U < \infty \\ & b_L \leq Ax \leq b_U \\ & c_L \leq c(x) \leq c_U, \quad x_j \in \mathbb{N} \quad \forall j \in I, \end{aligned} \tag{10}$$

where  $x, x_L, x_U \in \mathbb{R}^n$ ,  $f(x) \in \mathbb{R}$ ,  $A \in \mathbb{R}^{m_1 \times n}$ ,  $b_L, b_U \in \mathbb{R}^{m_1}$  and  $c_L, c(x), c_U \in \mathbb{R}^{m_2}$ . The variables  $x \in I$ , the index subset of  $1, \dots, n$ , are restricted to be integers.

The following files are required to define a problem of this category in TOMLAB.

**File:** `tomlab/quickguide/minlpQG.f.m`, `minlpQG.g.m`, `minlpQG.H.m`, `minlpQG.c.m`, `minlpQG.dc.m`, `minlpQG.d2c.m`

```
f:   Function value
g:   Gradient vector
H:   Hessian matrix
c:   Nonlinear constraint vector
dc:  Nonlinear constraint gradient matrix
d2c: The second part of the Hessian to the Lagrangian
      function for the nonlinear constraints.
```

An example of a problem of this class, (that is also found in the TOMLAB Quickguide) is `minlpQG`

**File:** `tomlab/quickguide/minlpQG.m`

```
% minlpQG is a small example problem for defining and solving
% mixed-integer nonlinear programming problems using the TOMLAB format.
```

```
Name='minlp1Demo - Kocis/Grossman.';
```

```
IntVars = [ 0 0 1 1 1 ]; % Integer variables: x(3)-x(5)
VarWeight = [ ]; % No priorities given
```

```
% There are divisions and square roots involving x(2), so we must
% have a small but positive value for the lower bound on x(2).
BIG = 1E8;
```

```
x_L = [ 0 1/BIG 0 0 0 ]'; % Lower bounds on x
```

```

x_U = [ BIG BIG 1 1 1 ]'; % Upper bounds on x

% Three linear constraints
A = [1 0 1 0 0 ; ...
     0 1.333 0 1 0 ; ...
     0 0 -1 -1 1 ];

b_L = []; % No lower bounds
b_U = [1.6 ; 3 ; 0]; % Upper bounds

c_L = [1.25;3]; % Two nonlinear constraints
c_U = c_L; % c_L==c_U implies equality

x_0 = ones(5,1); % Initial value

x_opt = [1.12,1.31,0,1,1]'; % One optimum known
f_opt = 7.6672; % Value f(x_opt)

x_min = [-1 -1 0 0 0]; % Used for plotting, lower bounds
x_max = [ 1 1 1 1 1]; % Used for plotting, upper bounds

HessPattern = spalloc(5,5,0); % All elements in Hessian are zero.
ConsPattern = [ 1 0 1 0 0; ... % Sparsity pattern of nonlinear
               0 1 0 1 0 ]; % constraint gradient

fIP = []; % An upper bound on the IP value wanted. Makes it possible
xIP = []; % to cut branches. xIP: the x value giving fIP

% Generate the problem structure using the TOMLAB Quick format
Prob = minlpAssign('minlpQG_f', 'minlpQG_g', 'minlpQG_H', HessPattern, ...
                 x_L, x_U, Name, x_0, ...
                 IntVars, VarWeight, fIP, xIP, ...
                 A, b_L, b_U, 'minlpQG_c', 'minlpQG_dc', 'minlpQG_d2c', ...
                 ConsPattern, c_L, c_U, ...
                 x_min, x_max, f_opt, x_opt);
Prob.DUNDEE.optPar(20) = 1;
Prob.P = 1; % Needed in minlpQG_xxx files

% Get default TOMLAB solver for your current license, for "minlp" problems
% Solver = GetSolver('minlp');

% Call driver routine tomRun, 3rd argument > 0 implies call to PrintResult

Result = tomRun('minlpBB',Prob,2);

```

## 9 Linear Least Squares Problems: `lls_prob`

In `lls_prob` there are two linear least squares test problems with about 10 variables and a few constraints. In order to define the first problem and solve it execute the following in Matlab:

```
Prob = probInit('lls_prob',1);
Result = tomRun('',Prob);
```

The basic structure of a general linear least squares problem is:

$$\begin{aligned} \min_x \quad & f(x) = \frac{1}{2} \|Cx - d\| \\ \text{s/t} \quad & \begin{array}{l} x_L \leq x \leq x_U, \\ b_L \leq Ax \leq b_U \end{array} \end{aligned} \tag{11}$$

where  $x, x_L, x_U \in \mathbb{R}^n$ ,  $d \in \mathbb{R}^M$ ,  $C \in \mathbb{R}^{M \times n}$ ,  $A \in \mathbb{R}^{m_1 \times n}$  and  $b_L, b_U \in \mathbb{R}^{m_1}$ .

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `llsQG`:

**File:** `tomlab/quickguide/llsQG.m`

```
% llsQG is a small example problem for defining and solving
% linear least squares using the TOMLAB format.
Name='LSSOL test example';           % Problem name, not required.
n = 9;
x_L = [-2 -2 -inf, -2*ones(1,6)]';   % Lower bounds on x
x_U = 2*ones(9,1);                  % Upper bounds on x

% Matrix defining linear constraints
A = [ ones(1,8) 4; 1:4,-2,1 1 1 1; 1 -1 1 -1, ones(1,5)];
b_L = [2 -inf -4]';                 % Lower bounds on the linear inequalities
b_U = [inf -2 -2]';                 % Upper bounds on the linear inequalities

% Vector m x 1 with observations in objective ||Cx -y(t)||
y = ones(10,1);

% Matrix m x n in objective ||Cx -y(t)||
C = [ ones(1,n); 1 2 1 1 1 1 2 0 0; 1 1 3 1 1 1 -1 -1 -3; ...
      1 1 1 4 1 1 1 1 1; 1 1 1 3 1 1 1 1 1; 1 1 2 1 1 0 0 0 -1; ...
      1 1 1 1 0 1 1 1 1; 1 1 1 0 1 1 1 1 1; 1 1 0 1 1 1 2 2 3; ...
      1 0 1 1 1 1 0 2 2];

% Starting point.
x_0 = 1./[1:n]';

% x_min and x_max are only needed if doing plots.
x_min = -ones(n,1);
x_max = ones(n,1);
```

```
% x_opt estimate.
x_opt = [2 1.57195927 -1.44540327 -0.03700275 0.54668583 0.17512363 ...
        -1.65670447 -0.39474418 0.31002899];
f_opt = 0.1390587318; % Estimated optimum.
```

```
% See 'help llsAssign' for more information.
Prob = llsAssign(C, y, x_L, x_U, Name, x_0, ...
                [], [], [], ...
                A, b_L, b_U, ...
                x_min, x_max, f_opt, x_opt);
```

```
Result = tomRun('clsSolve', Prob, 1);
%Result = tomRun('nlssol', Prob, 1);
%Result = tomRun('snopt', Prob, 1);
%Result = tomRun('lssol', Prob, 1);
```

## 10 (Constrained) Nonlinear Least Squares Problems: `cls_prob`, `mgh_prob` and `ls_prob`

The TOMLAB bundle `testprob` provides three sets of test problems for (constrained) nonlinear least squares: `cls_prob`, `mgh_prob` and `ls_prob`.

### 10.1 An example of a (constrained) nonlinear least squares

The basic structure of a (constrained) nonlinear least squares problem is the following

$$\begin{aligned} \min_x \quad & f(x) = \frac{1}{2}r(x)^T r(x) \\ \text{s/t} \quad & \begin{aligned} x_L &\leq x \leq x_U, \\ b_L &\leq Ax \leq b_U \\ c_L &\leq c(x) \leq c_U \end{aligned} \end{aligned} \tag{12}$$

where  $x, x_L, x_U \in \mathbb{R}^n$ ,  $r(x) \in \mathbb{R}^M$ ,  $A \in \mathbb{R}^{m_1 \times n}$ ,  $b_L, b_U \in \mathbb{R}^{m_1}$  and  $c_L, c(x), c_U \in \mathbb{R}^{m_2}$ . The following file defines and solves a problem in TOMLAB.

The following files are required to define a problem of this category in TOMLAB.

**File:** `tomlab/quickguide/nllsQG_r.m`, `nllsQG_J.m`

```
r: Residual vector
J: Jacobian matrix
```

The following file illustrates how to solve an NLLS problem in TOMLAB. Also view the m-files specified above for more information.

**File:** `tomlab/quickguide/nllsQG.m`

An example of a problem of this class, (that is also found in the TOMLAB Quickguide) is `nllsQG`:

```
% nllsQG is a small example problem for defining and solving
% nonlinear least squares using the TOMLAB format.
Name='Gisela';

t = [0.25; 0.5; 0.75; 1; 1.5; 2; 3; 4; 6; 8; 12; 24; 32; 48; 54; 72; 80;...
     96; 121; 144; 168; 192; 216; 246; 276; 324; 348; 386];
y = [30.5; 44; 43; 41.5; 38.6; 38.6; 39; 41; 37; 37; 24; 32; 29; 23; 21;...
     19; 17; 14; 9.5; 8.5; 7; 6; 6; 4.5; 3.6; 3; 2.2; 1.6];

x_0 = [6.8729, 0.0108, 0.1248]';

% See help clsAssign for more information.
Prob = clsAssign('nllsQG_r', 'nllsQG_J', [], [], [], Name, x_0, ...
                y, t);

% Parameter which is passed to r and J routines.
```

```

Prob.uP = 5;

Result = tomRun('clsSolve', Prob, 1);
%Result = tomRun('nlssol', Prob, 1);

% Any nonlinear solver can be used. TOMLAB automatically
% uses gateway routines for problem mapping.

%Result = tomRun('filterSQP', Prob, 1);
%Result = tomRun('knitro', Prob, 1);
%Result = tomRun('conopt', Prob, 1);
%Result = tomRun('snopt', Prob, 1);
%Result = tomRun('npsol', Prob, 1);
%Result = tomRun('minos', Prob, 1);
%Result = tomRun('oqnlp', Prob, 1);

```

## 10.2 cls\_prob

cls\_prob consists of 45 constrained nonlinear least squares test problems with up to 14 variables and a mixture of linear and nonlinear constraints. In order to define this problem and solve it execute the following in Matlab:

```

Prob = probInit('cls_prob',1);
Result = tomRun('',Prob);

```

## 10.3 mgh\_prob

In mgh\_prob there are 35 nonlinear least squares (More, Garbow, Hillstrom) test problems with up to 40 variables. In order to define this problem and solve it execute the following in Matlab:

```

Prob = probInit('mgh_prob',1);
Result = tomRun('',Prob);

```

## 10.4 ls\_prob

In ls\_prob there are 15 nonlinear least squares test problems with up to 20 variables. In order to define this problem and solve it execute the following in Matlab:

```

Prob = probInit('ls_prob',1);
Result = tomRun('',Prob);

```

## 11 Global Optimization Problems: glb\_prob, lgo1\_prob, lgo2\_prob and gkls\_prob.

### 11.1 Example of a global optimization problem

The basic structure of a global optimization problem with simple bounds is the following

$$\min_x f(x) \tag{13}$$

$$s/t \quad -\infty < x_L \leq x \leq x_U < \infty$$

where  $x, x_L, x_U \in \mathbb{R}^n$ ,  $f(x) \in \mathbb{R}$ . The variables  $x \in I$ , the index subset of  $1, \dots, n$ , are restricted to be integers.

The following file is required to define a problem of this category in TOMLAB.

**File:** tomlab/quickguide/glbQG.f.m

```
f: Function
```

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is glbQG:

**File:** tomlab/quickguide/glbQG.m

```
% glbQG is a small example problem for defining and solving  
% unconstrained global programming problems using the TOMLAB format.
```

```
Name = 'Shekel 5';  
x_L = [ 0 0 0 0]'; % Lower bounds for x.  
x_U = [10 10 10 10]'; % Upper bounds for x.  
x_0 = [-3.0144 -2.4794 -3.1584 -3.1790]; % May not be used.  
x_opt = [];  
f_opt = -10.1531996790582;  
f_Low=-20; % Lower bound on function.  
x_min = [ 0 0 0 0]; % For plotting  
x_max = [10 10 10 10]; % For plotting
```

```
Prob = glcAssign('glbQG_f', x_L, x_U, Name, [], [], [], ...  
                [], [], [], x_0, ...  
                [], [], [], [], ...  
                f_Low, x_min, x_max, f_opt, x_opt);
```

```
Prob.optParam.MaxFunc = 1500;
```

```
Result1 = tomRun('glbFast', Prob, 1); %Global solver  
Result2 = tomRun('conSolve', Prob, 1); %Local solver
```

```
% Result = tomRun('glbSolve', Prob, 1);  
% Result = tomRun('glcSolve', Prob, 1);  
% Result = tomRun('glcFast', Prob, 1);
```

```
% Result = tomRun('lgo', Prob, 1);  
% Result = tomRun('oqnlp', Prob, 1);
```

## 11.2 glb\_prob

In `glb_prob` there are 51 unconstrained global optimization test problems with sizes to 100 variables. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('glb_prob',n);  
Result = tomRun('',Prob);
```

## 11.3 lgo1\_prob

In `lgo1_prob` there are 30 global optimization test problems in one dimension. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('lgo1_prob',n);  
Result = tomRun('',Prob);
```

## 11.4 lgo2\_prob

In `lgo2_prob` there are 30 global optimization test problems in two up to four dimensions. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('lgo2_prob',n);  
Result = tomRun('',Prob);
```

## 11.5 gkls\_prob

In `gkls_prob` there are 300 global optimization test problems in two dimensions. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('gkls_prob',n);  
Result = tomRun('',Prob);
```

## 12 Constrained Global Optimization Problems: glc\_prob

In `glc_prob` there are 30 global mixed-integer nonlinear programming test problems with sizes to 20 variables and 5 constrains. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('glc_prob',n);
Result = tomRun('',Prob);
```

The basic structure of a constrained global optimization problems is the following

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s/t} \quad & -\infty < x_L \leq x \leq x_U < \infty \\ & b_L \leq Ax \leq b_U \\ & c_L \leq c(x) \leq c_U, \quad x_j \in \mathbb{N} \quad \forall j \in I, \end{aligned} \tag{14}$$

where  $x, x_L, x_U \in \mathbb{R}^n$ ,  $f(x) \in \mathbb{R}$ ,  $A \in \mathbb{R}^{m_1 \times n}$ ,  $b_L, b_U \in \mathbb{R}^{m_1}$  and  $c_L, c(x), c_U \in \mathbb{R}^{m_2}$ . The variables  $x \in I$ , the index subset of  $1, \dots, n$ , are restricted to be integers.

The following files are required to define a problem of this category in TOMLAB.

**File:** `tomlab/quickguide/glcQG.f.m`, `glcQG.c.m`

```
f: Function
c: Constraints
```

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `glcQG`:

**File:** `tomlab/quickguide/glcQG.m`

```
% glcQG is a small example problem for defining and solving
% constrained global programming problems using the TOMLAB format.

Name = 'Hock-Schittkowski 59';
u = [75.196    3.8112    0.0020567  1.0345E-5  6.8306    0.030234  1.28134E-3 ...
     2.266E-7  0.25645    0.0034604  1.3514E-5  28.106    5.2375E-6  6.3E-8    ...
     7E-10    3.405E-4  1.6638E-6  2.8673    3.5256E-5];

x_L = [0 0]'; % Lower bounds for x.
x_U = [75 65]'; % Upper bounds for x.
b_L = []; b_U = []; A = []; % Linear constraints
c_L = [0 0 0]; % Lower bounds for nonlinear constraints.
c_U = []; % Upper bounds for nonlinear constraints.
x_opt = [13.55010424 51.66018129]; % Optimum vector
f_opt = -7.804226324; % Optimum
x_min = x_L; % For plotting
x_max = x_U; % For plotting
x_0 = [90 10]'; % If running local solver
```

```
Prob = glcAssign('glcQG_f', x_L, x_U, Name, A, b_L, b_U, ...  
                'glcQG_c', c_L, c_U, x_0, ...  
                [], [], [], [], ...  
                [], x_min, x_max, f_opt, x_opt);
```

```
Prob.user.u = u;  
Prob.optParam.MaxFunc = 1500;
```

```
Result = tomRun('glcFast', Prob, 1);  
%Result = tomRun('glcSolve', Prob, 1);  
%Result = tomRun('lgo', Prob, 1);  
%Result = tomRun('oqnlp', Prob, 1);
```

## 13 Unconstrained Optimization: uc\_prob and uhs\_prob.

The TOMLAB bundle `testprob` provides two sets of test problems for unconstrained optimization problems (with simple bounds): `uc_prob`, and `uhs_prob`.

### 13.1 An example of a unconstrained optimization problem (with simple bounds)

The basic structure of a unconstrained optimization problems (with simple bounds) is the following

$$\begin{aligned} \min_x \quad & f(x) \\ s/t \quad & -\infty < x_L \leq x \leq x_U < \infty \end{aligned} \tag{15}$$

where  $x, x_L, x_U \in \mathbb{R}^n$  and  $f(x) \in \mathbb{R}$ .

### 13.2 uc\_prob

In `glo1_prob` there are 17 unconstrained optimization problems (with simple bounds) with up to three variables. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('uc_prob',n);  
Result = tomRun('',Prob);
```

### 13.3 uhs\_prob

In `glo2_prob` there are 25 unconstrained optimization problems (with simple bounds) with up to 10 variables. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('uhs_prob',n);  
Result = tomRun('',Prob);
```

## 14 Linear Semi-Definite Programming Problem with Linear Matrix Inequalities: `sdp_prob`

In `sdp_prob` there is 1 linear semi-definite programming test problem with linear matrix inequalities with 3 variables. In order to define this problem and solve it execute the following in Matlab:

```
Prob = probInit('spd_prob',1);
Result = tomRun('',Prob);
```

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `glbQG`:

$$\begin{aligned} \min_x \quad & f(x) = c^T x \\ \text{s/t} \quad & x_L \leq x \leq x_U \\ & b_L \leq Ax \leq b_U \\ & Q_0^i + \sum_{k=1}^n Q_k^i x_k \preceq 0, \quad i = 1, \dots, m. \end{aligned} \tag{16}$$

where  $c, x, x_L, x_U \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m_i \times n}$ ,  $b_L, b_U \in \mathbb{R}^{m_i}$  and  $Q_k^i$  are symmetric matrices of similar dimensions in each constraint  $i$ . If there are several LMI constraints, each may have it's own dimension.

The following file is required to define a problem of this category in TOMLAB.

**File:** `tomlab/quickguide/sdpQG.m`

The following file illustrates how to define and solve a problem of this category in TOMLAB.

This problem appears to be infeasible.

```
% sdpQG is a small example problem for defining and solving
% semi definite programming problems with linear matrix
% inequalities using the TOMLAB format.

Name = 'sdp.ps example 2';

% Objective function
c = [1 2 3]';

% Two linear constraints
A = [ 0 0 1 ; 5 6 0 ];
b_L = [-Inf; -Inf];
b_U = [ 3 ; -3 ];

x_L = -1000*ones(3,1);
x_U = 1000*ones(3,1);

% Two linear matrix inequality constraints. It is OK to give only
% the upper triangular part.
SDP = [];
```

```
% First constraint
SDP(1).Q{1} = [2 -1 0 ; 0 2 0 ; 0 0 2];
SDP(1).Q{2} = [2 0 -1 ; 0 2 0 ; 0 0 2];
SDP(1).Qidx = [1; 3];

% Second constraint
SDP(2).Q{1} = diag( [0 1] );
SDP(2).Q{2} = diag( [1 -1] );
SDP(2).Q{3} = diag( [3 -3] );
SDP(2).Qidx = [0; 1; 2];

x_0 = [];

Prob = sdpAssign(c, SDP, A, b_L, b_U, x_L, x_U, x_0, Name);

Result = tomRun('pensdp', Prob, 1);
```

## 15 Linear Semi-Definite Programming Problem with Bilinear Matrix Inequalities: `bmi_prob`

In `bmi_prob` there is 1 linear semi-definite programming problem with bilinear matrix inequalities with 3 variables. In order to define this problem and solve it execute the following in Matlab:

```
Prob = probInit('bmi_prob',1);
Result = tomRun('',Prob);
```

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `glbQG`:

$$Q_0^i + \sum_{k=1}^n Q_k^i x_k + \sum_{k=1}^n \sum_{l=1}^n x_k x_l K_{kl}^i \preceq 0 \quad (17)$$

The following file illustrates how to define and solve a problem of this category in TOMLAB.

**File:** `tomlab/quickguide/bmiQG.m`

```
% bmiQG is a small example problem for defining and solving
% semi definite programming problems with bilinear matrix
% inequalities using the TOMLAB format.
```

```
Name='bmi.ps example 3';
A = [];
b_U = [];
b_L = [];

c = [ 0 0 1 ]; % cost vector

% One matrix constraint, set linear part first
SDP = [];

% The constant matrix is stored as
% SDP(i).Q{j} when SDP(i).Qidx(j) == 0
SDP(1).Q{1} = [-10 -0.5 -2 ; -0.5 4.5 0 ; -2 0 0 ];

SDP(1).Q{2} = [ 9 0.5 0 ; 0.5 0 -3 ; 0 -3 -1 ];
SDP(1).Q{3} = [-1.8 -0.1 -0.4 ; -0.1 1.2 -1 ; -0.4 -1 0 ];

% Sparse is fine, too. Eventually, all the matrices are
% converted to sparse format.
SDP(1).Q{4} = -speye(3);

SDP(1).Qidx = [0; 1; 2; 3];

% Now bilinear part
```

```
% K_12 of constraint 1 (of 1) is nonzero, so set in SDP(i).K{1}.
SDP(1).K{1} = [0 0 2 ; 0 -5.5 3 ; 2 3 0 ];
SDP(1).Kidx = [1 2];
n = length(c);

x_L = [-5 ; -3 ; -Inf];
x_U = [ 2 ;  7 ;  Inf];
x_0 = [ 0 ;  0 ;  0 ];

f_Low = [];

Prob = bmiAssign([], c, SDP, A, b_L, b_U, x_L, x_U, x_0, ...
                Name, f_Low);

Result = tomRun('penbmi', Prob, 1);
```

## 16 Constrained Goal Attainment Problems: `goals_prob` and `mco_prob`

The TOMLAB bundle `testprob` provides two sets of problems for constrained goal attainment problems: `goals_prob` and `mco_prob`.

### 16.1 An example of a constrained goal attainment problems

The basic structure of a constrained goal attainment problems is the following:

$$\begin{aligned} \min_x \quad & \max \quad lam : r(x) - w * lam \leq g \\ \text{subject to} \quad & x_L \leq x \leq x_U \\ & b_L \leq Ax \leq b_U \\ & c_L \leq c(x) \leq c_U \end{aligned} \tag{18}$$

where  $x, x_L, x_U \in \mathbb{R}^n$ ,  $r(x) \in \mathbb{R}^N$ ,  $c(x), c_L, c_U \in \mathbb{R}^{m_1}$ ,  $b_L, b_U \in \mathbb{R}^{m_2}$ ,  $A \in \mathbb{R}^{m_2 \times n}$ ,  $g \in \mathbb{R}^m$ , and  $w \in \mathbb{R}^m$ .

An example of a problem of this class, (that is also found in the TOMLAB quickguide) is `goalsQG`:

**File:** `tomlab/quickguide/goalsQG.r.m`, `goalsQG_J.m`, `goalsQG_c`, `goalsQG_dc`

```
r: Residual vector
J: Jacobian matrix
c: Nonlinear constraint vector
dc: Nonlinear constraint gradient matrix
```

The following file illustrates how to define and solve a problem of this category in TOMLAB.

**File:** `tomlab/quickguide/goalsQG.m`

```
% goalsQG is a small example problem for defining and solving
% multi criteria optimization problems using the TOMLAB format.
```

```
Name='EASY-TP355';
% Constrained least squares problem, four quadratic terms and local solutions
% Hock W., Schittkowski K. (1981):
x_0 = zeros(4,1); % Lower bounds for x.
x_L = zeros(4,1); % Upper bounds for x.
x_U = 1e5*ones(4,1); % Starting point.
x_min = []; % For plotting.
x_max = []; % For plotting.
A = [1 0 0 0;0 1 0 0]; % Linear constraints.
b_L = [0.1;0.1]; % Lower bounds.
b_U = [0.1;0.1]; % Upper bounds.
c_L = 0; % Lower bounds.
c_U = 0; % Upper bounds.
y = zeros(2,1); % Residuals
```

```
Prob = clsAssign('goalsQG_r', 'goalsQG_J', [], x_L, x_U, Name, x_0,...
```

```
y, [], [], [], [], [],...  
A, b_L, b_U, 'goalsQG_c', 'goalsQG_dc', [], c_L, c_U,...  
x_min, x_max);
```

```
PriLev = 2;  
Result = tomRun('goalSolve', Prob, PriLev);
```

## 16.2 mco\_prob

In `glb_prob` there are 9 Multi-Criterion unconstrained and constrained nonlinear test problems with up to 10 variables and few constrains. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('mco_prob',n);  
Result = tomRun('',Prob);
```

## 16.3 goals\_prob

In `goals_prob` there are 9 constrained goal attainment test problems with sizes to 9 variables and about 10 constrains. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('goals_prob',n);  
Result = goalSolve(Prob,1);
```

## 17 Geometric programming problems: gp\_prob

In `gp_prob` there are 14 geometric programming test problems with sizes to 12 variables and about 10 constraints. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('gp_prob',n);
Result = tomRun('',Prob);
```

The primal geometric programming problem is defined below (the dual is used internally).

$$(GP) \quad V_{GP} := \begin{array}{ll} \text{minimize} & g_0(t) \\ \text{subject to} & g_k(t) \leq 1, \quad k = 1, 2, \dots, p \\ & t_i > 0, \quad i = 1, 2, \dots, m \end{array} \quad (19)$$

where

$$g_0(t) = \sum_{j=1}^{n_0} c_j t_1^{a_{1j}} \dots t_m^{a_{mj}} \quad (20)$$

$$g_k(t) = \sum_{j=n_{k-1}+1}^{n_k} c_j t_1^{a_{1j}} \dots t_m^{a_{mj}}, \quad k = 1, 2, \dots, p. \quad (21)$$

Given exponents  $a_{ij}$  for the  $i$ th variable in the  $j$ th product term,  $i = 1, \dots, m$  and  $j = 1, \dots, n_p$ , are arbitrary real constants and term coefficients  $c_j$  are positive.

Example problem:

$$(P1) \quad \begin{array}{ll} \min & 5x_1 + 50000x_1^{-1} + 20x_2 + 72000x_2^{-1} + 10x_3 + 144000x_3^{-1} \\ \text{subject to} & 4x_1^{-1} + 32x_2^{-1} + 120x_3^{-1} \leq 1 \\ & x \geq 0 \end{array}$$

The following file illustrates how to define and solve a problem of this category in TOMLAB.

**File:** `tomlab/quickguide/gpQG.m`

```
% gpQG is a small example problem for defining and solving
% geometric programming problems using the TOMLAB format.
```

```
nterm = [6;3];
coef = [.5e1;.5e5;.2e2;.72e5;.1e2;.144e6;.4e1;.32e2;.12e3];
A = sparse([ 1 -1 0 0 0 0 -1 0 0;...
            0 0 1 -1 0 0 0 -1 0;...
            0 0 0 0 1 -1 0 0 -1])';
```

```
Name = 'GP Example'; % File gpQG.m
```

```
% Assign routine for defining a GP problem.
Prob = gpAssign(nterm, coef, A, Name);

% Calling driver routine tomRun to run the solver.
% The 1 sets the print level after optimization.

Result = tomRun('GP', Prob, 1);
```

## 18 Fitting of positive sums of Exponentials: exp\_prob

In `exp_prob` there are 51 Fitting of positive sums of Exponentials test problems with up to 6 variables. In order to define the problem `n` and solve it execute the following in Matlab:

```
Prob = probInit('exp_prob',n);
Result = expSolve(Prob);
```

## Part II

# The `modellib` collection

## 19 Introduction to `modellib`

The bundle `modellib` is a collection of linear and mixed-integer programming problems. For each problem there are two `m`-files. One that formulate and define the problem with words and tables, and a second that interpret the tables into the standard TOMLAB `prob` format and solves it. In order to interpret the results a large amount of text is displayed explaining the `x.k` vector.

The problems in `modellib` are originally from a translation by Hickpe of the text “Programmation linare” by Gueret, Prins and Seveaux with the english title “Applications of optimization...”.

## 20 Mining and Processing

### 20.1 Production of alloys

```
% function Result = productionofalloysEx(PriLev)
%
% Creates a TOMLAB LP problem for production design of alloys
%
% PRODUCTION OF ALLOYS
%
% The company Steel has received an order for 500 tonnes of steel to
% be used in shipbuilding. This steel must have the following
% characteristics (grades).
%
% Characteristics of steel ordered
% +-----+-----+-----+
% |Chemical      |Minimum|Maximum|
% |Element       |Grade  |Grade  |
% +-----+-----+-----+
% |Carbon (C)    | 2.0   | 3.0   |
% |Copper (Cu)   | 0.4   | 0.6   |
% |Manganese (Mn)| 1.2   | 1.65  |
% +-----+-----+-----+
%
% The company has seven different raw materials in stock that may be
% used for the production of this steel. The table below lists the
% grades, available amounts and prices for all raw materials.
%
% Raw material grades, availabilities, and prices
% +-----+-----+-----+-----+-----+
% |Raw material  |C %|Cu %|Mn %|Availability|Cost|
% +-----+-----+-----+-----+-----+
% |Iron alloy 1  |2.5| 0  |1.3 | 400  |200 |
% |Iron alloy 2  | 3 | 0  |0.8 | 300  |250 |
% |Iron alloy 3  | 0 | 0.3|0   | 600  |150 |
% |Copper alloy 1| 0 | 90 |0   | 500  |220 |
% |Copper alloy 2| 0 | 96 |4   | 200  |240 |
% |Aluminum alloy 1| 0 | 0.4|1.2 | 300  |200 |
% |Aluminum alloy 2| 0 | 0.6|0   | 250  |165 |
% +-----+-----+-----+-----+-----+
%
% The objective is to determine the composition of the steel that
% minimizes the production cost.
%
% VARIABLES
%
% compsize                Amount of steel to produce
```

```

% mincomp           Minimal contents of each component.
% maxcomp           Maximal contents of each component.
% rawcompmat        Contents of raw material.
% rawavail          Amount available of raw material.
% rawcost           Cost of raw material.
%
% RESULTS
%
% Result.x_k has the same length as there are raw material. In it we
% find how much of each component we should buy. For an
% interpretation of the results, use PriLev > 1, for example:
% Result = productionofalloysEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%
% OUTPUT PARAMETERS
% Result           Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 7, 2005.   Last modified Oct 7, 2005.

```

```
function Result = productionofalloysEx(PriLev)
```

```

if nargin < 1
    PriLev = 1;
end

```

```

compsize   = 500;
mincomp    = [2;.4;1.2];
maxcomp    = [3;.6;1.65];
rawcompmat = [[2.5; 3; 0; 0; 0; 0; 0],...
              [ 0; 0; .3; 90; 96; .4; .6],...
              [1.3; .8; 0; 0; 4;1.2; 0]];
rawavail   = [ 400;300;600;500;200;300;250];
rawcost    = [ 200;250;150;220;240;200;165];

```

```

Prob = productionofalloys(compsize, mincomp, maxcomp, rawcompmat, rawavail, rawcost);
Result = tomRun('cplex', Prob, PriLev);

```

```

if PriLev > 1,

```

```
names = [ 'Iron 1' ; 'Iron 2' ; 'Iron 3' ; 'Copp 1' ; ...
          'Copp 2' ; 'Alum 1' ; 'Alum 2'];
disp(['the optimal steel contains:'])
for i = 1:length(Result.x_k),
    if Result.x_k(i) > 0,
        disp(['      ' num2str(Result.x_k(i)) ' tonnes of ' names(i,:) ])
    end
end
end
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051007 med Created.
```

```
% 060110 per Added documentation.
```

```
% 060125 per Moved disp to end
```

## 20.2 Animal food Production

```

% function Result = animalfoodproductionEx(PriLev)
%
% Creates a TOMLAB LP problem for animal food production
%
% ANIMAL FOOD PRODUCTION
%
% The company CowFood produces food for farm animals that is sold in
% two forms: powder and granules. The raw materials used for the
% production of the food are: oat, maize and molasses. The raw
% materials (with the exception of molasses) first need to be ground,
% and then all raw materials that will form a product are blended. In
% the last step of the production process the product mix is either
% transformed to granules or sieved to obtain food in the form of
% powder.
%
%           Molasses +-
%                   |
%                   V      +----> Granulating --> Granules
% Oat ----+          |
%           +--> Grinding --> Blending ---+
% Maize ---+         |
%                   +----> Sieving -----> Powder
% Animal food production process
%
% Every food product needs to fulfill certain nutritional
% requirements. The percentages of proteins, lipids and fibers
% contained in the raw materials and the required percentages in the
% final products are listed in the table below.
%
% Contents of nutritional components in percent
%
% +-----+-----+-----+-----+
% |Raw material      |Proteins|Lipids |Fiber|
% +-----+-----+-----+-----+
% |Oat                | 13.6  | 7.1  | 7.0 |
% |Maize              | 4.1   | 2.4  | 3.7 |
% |Molasses           | 5.0   | 0.3  | 25  |
% |Required contents| >=9.5 | >=2  | <=6 |
% +-----+-----+-----+-----+
%
% There are limits on the availability of raw materials. The table
% below displays the amount of raw material that is available every
% day and the respective prices.
%
% Raw material availabilities and prices
%

```

```

% +-----+-----+-----+
% |Raw material|Available amount in kg|Cost in $/kg|
% +-----+-----+-----+
% |Oat          | 11900                | 0.13    |
% |Maize        | 23500                | 0.17    |
% |Molasses     | 750                  | 0.12    |
% +-----+-----+-----+
%
% The cost of the different production steps are given in the
% following table.
%
% Production costs in $/kg
%
% +-----+-----+-----+-----+
% |Grinding|Blending|Granulating|Sieving|
% +-----+-----+-----+-----+
% | 0.25  | 0.05   | 0.42    | 0.17  |
% +-----+-----+-----+-----+
%
% With a daily demand of nine tonnes of granules and twelve tonnes of
% powder, which quantities of raw materials are required and how
% should they be blended to minimize the total cost?
%
% VARIABLES
%
% compsize          Amount of different food to produce
% mincomp           Minimal content of nutrients
% maxcomp           Maximal content of nutrients
% rawcompmat        Content of nutrients in raw sources
% rawavail          Available raw sources
% rawcost           Cost of raw material
% prodcostsraw      Cost to process raw material
% prodcostsprod     Cost to produce products
%
% RESULTS
%
% For an interpretation of the results, use PriLev > 1, for example:
% animalfoodproductionEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%

```

```

% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 7, 2005.   Last modified Oct 7, 2005.

function Result = animalfoodproductionEx(PriLev)

if nargin < 1
    PriLev = 1;
end

compsize    = [9000;12000];
mincomp     = [9.5;2;0];
maxcomp     = [100;100;6];
rawcompmat  = [[13.6;4.1;5],...
               [7.1;2.4;.3],...
               [7;3.7;25]];
rawavail    = [11900;23500;750];
rawcost     = [.13;.17;.12];
prodcostsraw = [[.25;.05;0;0],...
                [.25;.05;0;0],...
                [0;.05;0;0]];
prodcostsprod = [[0;0;.42;0],...
                 [0;0;0;.17]];

% .25;.05;.42;.17
% grind, blend, gran, siev

Prob = animalfoodproduction(compsize, mincomp, maxcomp, rawcompmat,...
    rawavail, rawcost, prodcostsraw, prodcostsprod);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    raws = length(rawavail);
    prods = length(compsize);

    for prod = 1:prods,
        disp(['produce ' num2str(Result.x_k(end-prods+prod)) ...
            ' of product ' num2str(prod) ','])
        disp(' and use the following ingredients:')

        for raw = 1:raws,
            disp([' ' num2str(Result.x_k(raws*(prod-1) + raw)) ...
                ' units of ingredient ' num2str(raw) ])
        end
    end
end

```

```
end  
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051007 med Created.
```

```
% 060110 per Added documentation.
```

```
% 060125 per Moved disp to end
```

## 20.3 Refinery

```

% function Result = refineryEx(PriLev)
%
% Creates a TOMLAB LP problem for refinery production
%
% REFINERY
%
% A refinery produces butane, petrol, diesel oil, and heating oil
% from two crudes. Four types of operations are necessary to obtain
% these products: separation, conversion, upgrading, and blending.
%
% The separation phase consists of distilling the raw product into,
% among others, butane, naphtha, gasoil, and a residue. The residue
% subsequently undergoes a conversion phase (catalytic cracking) to
% obtain lighter products. The different products that come out of
% the distillation are purified (desulfurization or sweetening) or
% upgraded by a reforming operation that augments their octane value.
% Finally, to obtain the products that will be sold, the refinery
% blends several of the intermediate products in order to fulfill the
% prescribed characteristics of the commercial products. The
% following drawing gives a simplified overview on the production
% processes in this refinery.
%
% After the distillation, crude 1 gives 3% butane, 15% naphtha, 40%
% gasoil, and 15% residue. Crude 2 results in 5% butane, 20% naphtha,
% 35% gasoil, and 10% residue. The reforming of the naphtha gives 15%
% butane and 85% of reformat (reformed naphtha). The catalytic
% cracking of the residue results in 40% of cracked naphtha and 35%
% of cracked gasoil (note that these percentages do not add up to
% 100% because the process also produces 15% of gas, 5% coke and
% another type of residue that are not taken into consideration in
% our example). The petrol is produced with three ingredients:
% reformed naphtha (reformat), butane, and cracked naphtha. The
% diesel oil is obtained by blending sweetened gasoil, cracked
% gasoil, and cracked naphtha. The heating oil may contain gasoil and
% cracked naphtha without any restrictions on their proportions.
%
%
%           Crudes
%           |
%           V
%           Distillation -----+
%           |
% +-----+-----+-----+-----+
% |           |           |           |
% Gasoil      Residue     Naphta      |
% |           |           |           |

```

```

% V          V          V          V
% Desulf.    Cat. crack  Reform --> Butane
% |          |          |          |
% Sw.Gasoil  C.Naphta   C.Gasoil  Reformate   |
% |          |          |          |          V
% |          |          |          | To Petrol Blending and Butane
% |          |          |          V
% |          |          |          To Petrol Blending
% |          |          V
% |          |          To Heating Oil and Diesel Blending
% |          V
% |          To Petrol, Heating Oil and Diesel Blending
% V
% To Heating Oil and Diesel Blending
%

```

% Simplified representation of a refinery

```

%
% Certain conditions on the quality of the petrol and diesel oil are
% imposed by law. There are three important characteristics for
% petrol: the octane value, vapor pressure and volatility. The octane
% value is a measure of the anti-knock power in the motor. The vapor
% pressure is a measure of the risk of explosion during storage,
% especially with hot weather. The volatility is a measure for how
% easy the motor is started during cold weather. Finally, the
% maximum sulfur content of the diesel oil is imposed by
% antipollution specifications. The following table summarizes the
% required characteristics of the final products and the composition
% of the intermediate ones. Fields are left empty if no particular
% limit applies. We work with the assumption that all these
% characteristics blend linearly by weight (in reality, this is only
% true for the sulfur contents).
%

```

% Characteristics of intermediate and final products

```

%+-----+-----+-----+-----+-----+-----+-----+-----+
%|Characteristic|Butane|Reformate|Cracked|Cracked|Desulf |Petrol|Diesel|
%|              |      |          |naphtha|gasoil |gasoil |      |oil  |
%+-----+-----+-----+-----+-----+-----+-----+-----+
%|Octane value  | 120 | 100 | 74  | -  | -  | >=94 | -  |
%|Vapor pressure| 60  | 2.6 | 4.1 | -  | -  | <=12.7| -  |
%|Volatility    | 105 | 3   | 12  | -  | -  | >=17  | -  |
%|Sulfur (in %) | -   | -   | 0.12| 0.76| 0.03| -    | <=0.05|
%+-----+-----+-----+-----+-----+-----+-----+-----+
%

```

```

% In the next month the refinery needs to produce 20,000 tonnes of
% butane, 40,000 tonnes of petrol, 30,000 tonnes of diesel oil, and
% 42,000 tonnes of heating oil. 250,000 tonnes of crude 1 and 500,000

```

```

% tonnes of crude 2 are available. The monthly capacity of the
% reformer are 30,000 tonnes, for the desulfurization 40,000 tonnes
% and for the cracking 50,000 tonnes. The cost of processing is based
% on the use of fuel and catalysts for the different operations: the
% costs of distillation, reforming, desulfurization, and cracking are
% $ 2.10, $ 4.18, $ 2.04 and $ 0.60 per tonne respectively.
%
% VARIABLES
%
% demand                Demand of each product
% supply                Supply of crudes
% capacity              Capacity of reformer, desulfer and cracker.
% costs                 Cost for each step.
% supplycomp            Contents of crudes.
% numvbls               Number of variables
% octane                Octane in intermediates
% vappres               Vapor Pressure in intermediates
% volatility            Volatility of intermediates
% sulfur                Sulfur in intermediates
% reformer              Output of components from reformer
% cracker               Output of components from cracker
% petrolspec_L          Lower bounds (contents in petrol)
% petrolspec_U          Upper bounds (contents in petrol)
% dieselspec_L          Lower bounds (contents in diesel)
% dieselspec_U          Upper bounds (contents in diesel)
%
% RESULTS
%
% x_k is a vector containing:
% 4 Final products      butane, petrol, diesel, heating oil
% 3 Intermediates to Petrol  petbutane, reformerate, petcrknaptha
% 3 Intermediates to Diesel  dslgasoil, dslcrknaptha, dslcrkgasoil
% 3 Intermediates to Heating Oil hogasoil, hocrknaptha, hocrkgasoil
% 4 Intermediates from Distilling distbutane, naphtha, residue, gasoil
% 2 Intermediates from Reforming refbutane, reformerate
% 2 Intermediates from Cracking crknaptha, crkgasoil
% 2 Crudes               crude1, crude2
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev                Print Level
%
% OUTPUT PARAMETERS

```

```

% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 10, 2005.  Last modified Jan 4, 2006.

function Result = refineryEx(PriLev)

if nargin < 1
    PriLev = 1;
end

demand      = [20;40;30;42]*1000;
supply      = [250;500]*1000;
capacity    = [30;40;50]*1000;
costs       = [2.1;2.1;4.18;.6;2.04];
supplycomp  = [.03 .15 .15 .40; .05 .2 .1 .35];
numvbls     = 4+3+3+3+4+2+2+2;

octane      = [120; 100; 74];
vappres     = [60; 2.6; 4.1];
volatility   = [105; 3; 12];
sulfur      = [.03; .12; .76];

reformer    = [.15 .85]';
cracker     = [.40 .35]';

petrolspec_L = [94; 17];
petrolspec_U = [12.7];
dieselspec_L = [-inf];
dieselspec_U = [.05];

% PRODUCTS
% butane, petrol, diesel, heating      (FINAL PRODUCTS, 4) 4
% petbutane, reformerate, petcrknaphtha (IPETROL, 3) 7
% dslgasoil, dslcrknaphtha, dslcrkgasoil (IDIESEL, 3) 10
% hogasoil, hocrknaphtha, hocrkgasoil   (IHO, 3) 13
% distbutane, naphtha, residue, gasoil  (IDIST, 4) 17
% refbutane, reformerate                (IREF, 2) 19
% crknaphtha, crkgasoil                 (ICRACK, 2) 21
% crude1, crude2                        (CRUDES, 2) 23

Prob = refinery(demand, supply, capacity, costs, supplycomp, numvbls, octane,...
    vappres, volatility, sulfur, reformer, cracker, petrolspec_L, petrolspec_U, ...
    dieselspec_L, dieselspec_U);

Result = tomRun('cplex', Prob, PriLev);

```

```

if PriLev > 1,
    disp('4 Final products')
    disp([' butane      - ' num2str(Result.x_k(1))])
    disp([' petrol      - ' num2str(Result.x_k(2))])
    disp([' diesel      - ' num2str(Result.x_k(3))])
    disp([' heating oil - ' num2str(Result.x_k(4))])
    disp('3 Intermediates to Petrol')
    disp([' petbutane   - ' num2str(Result.x_k(5))])
    disp([' reformerate - ' num2str(Result.x_k(6))])
    disp([' petcrknaptha - ' num2str(Result.x_k(7))])
    disp('3 Intermediates to Diesel')
    disp([' dslrgasoil   - ' num2str(Result.x_k(8))])
    disp([' dslcrknaptha - ' num2str(Result.x_k(9))])
    disp([' dslcrkgasoil - ' num2str(Result.x_k(10))])
    disp('3 Intermediates to Heating Oil')
    disp([' hogasoil     - ' num2str(Result.x_k(11))])
    disp([' hocrknaptha  - ' num2str(Result.x_k(12))])
    disp([' hocrkgasoil  - ' num2str(Result.x_k(13))])
    disp('4 Intermediates from Distilling')
    disp([' distbutane   - ' num2str(Result.x_k(14))])
    disp([' naphtha      - ' num2str(Result.x_k(15))])
    disp([' residue      - ' num2str(Result.x_k(16))])
    disp([' gasoil       - ' num2str(Result.x_k(17))])
    disp('2 Intermediates from Reforming')
    disp([' refbutane    - ' num2str(Result.x_k(18))])
    disp([' reformerate  - ' num2str(Result.x_k(19))])
    disp('2 Intermediates from Cracking')
    disp([' crknaptha    - ' num2str(Result.x_k(20))])
    disp([' crkgasoil    - ' num2str(Result.x_k(21))])
    disp('2 Crudes')
    disp([' crude1      - ' num2str(Result.x_k(22))])
    disp([' crude2      - ' num2str(Result.x_k(23))])
end

```

```
% MODIFICATION LOG
```

```
%
```

```
% 051007 med Created
```

```
% 060104 med Updated and corrected
```

```
% 060110 per Added documentation.
```

```
% 060125 per Moved disp to end
```

## 20.4 Cane sugar Production

```
% function Result = canesugarproductionEx(PriLev)
%
% Creates a TOMLAB LP problem for cane sugar production
%
% CANE SUGAR PRODUCTION
%
% PROBLEM
%
% The harvest of cane sugar in Australia is highly mechanized. The sugar
% cane is immediately transported to a sugar house in wagons that run on
% a network of small rail tracks. The sugar content of a wagon load
% depends on the field it has been harvested from and on the maturity of
% the sugar cane. Once harvested, the sugar content decreases rapidly
% through fermentation and the wagon load will entirely lose its value
% after a certain time. At this moment, eleven wagons all loaded with the
% same quantity have arrived at the sugar house. They have been examined
% to find out the hourly loss and the remaining life span (in hours) of
% every wagon, these data are summarized in the following table.
%
% Table: Properties of the lots of cane sugar
%
% Lot          1  2  3  4  5  6  7  8  9 10 11
% Loss (kg/h)  43 26 37 28 13 54 62 49 19 28 30
% Life span (h)  8  8  2  8  4  8  8  8  8  8  8
%
% Every lot may be processed by any of the three, fully equivalent
% production lines of the sugar house. The processing of a lot takes two
% hours. It must be finished at the latest at the end of the life span of
% the wagon load. The manager of the sugar house wishes to determine a
% production schedule for the currently available lots that minimizes the
% total loss of sugar.
%
% VARIABLES
%
% proclines          the number of processing lines
% proctime           time in hours to process a wagon
% loss               loss of sugar in kg per hour
% lifespan            how long the sugar in a wagon will last
%
% RESULTS
%
% For an interpretation of the results, set PriLev > 1, for example:
% Result = canesugarproductionEx(2);
%
% REFERENCES
```

```

%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 7, 2005.  Last modified Dec 8, 2005.

function Result = canesugarproductionEx(PriLev)

if nargin < 1
    PriLev = 1;
end

proclines = [3];
proctime  = [2];
loss      = [43;26;37;28;13;54;62;49;19;28;30];
lifespan  = [8;8;2;8;4;8;8;8;8;8];

Prob = canesugarproduction(proclines, proctime, loss, lifespan);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    lots      = length(lifespan);
    timeslots = ceil(lots/proclines(1)); % even a fraction means work
    temp      = reshape(Result.x_k,lots,timeslots);

    disp(['To minimize loss ( ' num2str(Result.f_k) ' ) use this schema'])

    for time = 1:timeslots,
        disp(['at ' num2str((time-1)*proctime(1)+8) '.00:'])
        idx = find(temp(:,time));
        disp([' process the lots ' num2str(idx) ])
    end
    disp(['at ' num2str((time)*proctime(1)+8) '.00:'])
    disp(' harvesting completed')
end

% MODIFICATION LOG
%
% 051007 med Created

```

% 051208 med Lifespan factor wrong  
% 060109 per lifespan(9) changed to 8  
% 060109 per Added documentation.  
% 060125 per Moved disp to end

## 20.5 Opencast mining

```

% function Result = opencastminingEx(PriLev)
%
% Creates a TOMLAB MIP problem for open cast mining
%
% OPENCAST MINING
%
% PROBLEM
%
% An opencast uranium mine is being prospected. Based on the results of
% some test drillings the mine has been subdivided into exploitation units
% called blocks. The pit needs to be terraced to allow the trucks to drive
% down to its bottom. The uranium deposit extends from east to west. The
% pit is limited in the west by a village and in the east by a group of
% mountains. Taking into account these constraints, 18 blocks of 10,000
% tonnes on three levels have been identified (Figure 6.3). To extract a
% block, three blocks of the level above it need to be extracted: the
% block immediately on top of it, and also, due to the constraints on the
% slope, the blocks to the right and to the left.
%
%
%      Village                                Mountains
%      +---+---+---+---+---+---+---+---+
% Level 1: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
%      +---+---+---+---+---+---+---+---+
% Level 2: |   | 9 | 10| 11| 12| 13| 14|   |
%      +---+---+---+---+---+---+---+---+
% Level 3: |   |   | 15| 16| 17| 18|   |   |
%      +---+---+---+---+---+---+---+---+
%
% For example: If we were to extract block 17 we also need to extract the
%                blocks 11, 12 and 13. And in order to extract these blocks
%                we also need to extract blocks 3, 4, 5, 6 and 7.
%
% It costs $ 100 per tonne to extract a block of level 1, $ 200 per
% tonne for a block of level 2, and $ 300 per tonne for a block of
% level 3, with the exception of the hatched blocks that are formed of a
% very hard rock rich in quartz and cost $ 1000 per ton. The only blocks
% that contain uranium are those displayed in a gray shade (1, 7, 10, 12,
% 17, 18). Their market value is 200, 300, 500, 200, 1000, and
% $ 1200/tonne respectively. Block 18, although rich in ore, is made of
% the same hard rock as the other hatched blocks. Which blocks should be
% extracted to maximize the total benefit?
%
% VARIABLES
%

```

```

% values          The profit for extracting the blocks
% depends         Block dependencies.
%
% RESULT
%
% The result will be a vector with binary variables, as many as we have
% blocks. A 1 means do extract, and a 0 means do not extract.
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html

%
% INPUT PARAMETERS
% PriLev         Print Level
%
% OUTPUT PARAMETERS
% Result         Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 7, 2005.   Last modified Oct 7, 2005.

function Result = opencastminingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

values      = [200-100;-100;-100;-100;-100;-100;300-100;-100;...
              -1000;500-200;-200;200-200;-200;-1000;-1000;-1000;...
              1000-300;1200-1000]*10000;
depends      = [-1 -1 -1 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0;...
              0 -1 -1 -1 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0;...
              0 0 -1 -1 -1 0 0 0 0 0 3 0 0 0 0 0 0 0 0;...
              0 0 0 -1 -1 -1 0 0 0 0 0 3 0 0 0 0 0 0;...
              0 0 0 0 -1 -1 -1 0 0 0 0 0 3 0 0 0 0;...
              0 0 0 0 0 -1 -1 -1 0 0 0 0 3 0 0 0 0;...
              0 0 0 0 0 0 0 0 -1 -1 -1 0 0 0 3 0 0;...
              0 0 0 0 0 0 0 0 0 -1 -1 -1 0 0 0 3 0;...
              0 0 0 0 0 0 0 0 0 0 -1 -1 -1 0 0 0 3];

Prob = opencastmining(values, depends);
Result = tomRun('cplex', Prob, PriLev);

```

```

if PriLev > 1,
    level1 = [];
    for i = 1 : 8,
        if Result.x_k(i) == 1,
            level1 = [level1 ' * '];
        else,
            level1 = [level1 ' - '];
        end
    end
    level2 = [ ' - '];
    for i = 9 : 14,
        if Result.x_k(i) == 1,
            level2 = [level2 ' * '];
        else,
            level2 = [level2 ' - '];
        end
    end
    level2 = [level2 ' - '];
    level3 = [ ' - ' ' - '];
    for i = 15 : 18,
        if Result.x_k(i) == 1,
            level3 = [level3 ' * '];
        else,
            level3 = [level3 ' - '];
        end
    end
    level3 = [level3 ' - ' ' - '];
disp(' ')
disp('Extraction profile')
disp(' ')
disp(['Level 1: ' level1])
disp(['Level 2: ' level2])
disp(['Level 3: ' level3])
disp(' ')
disp('Legend: * means do extract')
disp('          - means do not extract')
end

% MODIFICATION LOG
%
% 051007 med    Created.
% 060109 per    Added documentation.
% 060125 per    Moved disp to end

```

## 20.6 Production of Electricity

```

% function Result = productionofelectricityEx(PriLev)
%
% Creates a TOMLAB MIP problem for production of electricity
%
% PRODUCTION OF ELECTRICITY
%
% Power generators of four different types are available to satisfy the
% daily electricity demands (in megawatts) summarized in the following
% table. We consider a sliding time horizon: the period 10pm-12am of day d
% is followed by the period 0am-6am of day d + 1.
%
% Daily electricity demands (in MW)
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Period |0am-6am|6am-9am|9am-12pm|12pm-2pm|2pm-6pm|6pm-10pm|10pm-12am|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Demand | 12000| 32000| 25000| 36000| 25000| 30000| 18000|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% The power generators of the same type have a maximum capacity and may be
% connected to the network starting from a certain minimal power output.
% They have a start-up cost, a fixed hourly cost for working at minimal
% power, and an hourly cost per additional megawatt for anything beyond
% the minimal output. These data are given in the following table.
%
% Description of power generators
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Available|Min. output|Max.  |capacity|Fix cost|Add. MW cost|Start-up|
% |number  |in MW      |in MW |$/h     |        |$/h         |cost    |
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Type 1  | 10        | 750  | 1750   | 2250   | 2.7        | 5000   |
% |Type 2  | 4         | 1000 | 1500   | 1800   | 2.2        | 1600   |
% |Type 3  | 8         | 1200 | 2000   | 3750   | 1.8        | 2400   |
% |Type 4  | 3         | 1800 | 3500   | 4800   | 3.8        | 1200   |
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% A power generator can only be started or stopped at the beginning of a
% time period. As opposed to the start, stopping a power plant does not
% cost anything. At any moment, the working power generators must be able
% to cope with an increase by 20% of the demand forecast. Which power
% generators should be used in every period in order to minimize the total
% daily cost?
%
% VARIABLES
%

```

```

% demand          MW needed each period
% available       Generators of each type available
% mincap          Minimal output of generator
% maxcap          Maximal output of generator
% fixcost         Fix cost per hour
% runningcost     Cost per hour and MW above mincap
% startcost       Cost for starting a generator
% periodlengths  Hours in each period
%
% RESULTS
%
% For an interpretation of the results use PriLev > 1, for example:
% Result = productionofelectricityEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 10, 2005. Last modified Oct 10, 2005.

function Result = productionofelectricityEx(PriLev)

if nargin < 1
    PriLev = 1;
end

demand      = [12;32;25;36;25;30;18]*1000;
available   = [10;4;8;3];
mincap      = [750;1000;1200;1800];
maxcap      = [1750;1500;2000;3500];
fixcost     = [2250;1800;3750;4800];
runningcost = [2.7;2.2;1.8;3.8];
startcost   = [5000;1600;2400;1200];
periodlengths = [6;3;3;2;4;4;2];
Prob = productionofelectricity(demand, available, mincap, maxcap,...
    fixcost, runningcost, startcost, periodlengths);
Result = tomRun('cplex', Prob, PriLev);

```

```

if PriLev > 1,
    temp      = reshape(Result.x_k,length(available(:)),length(demand(:)),3);
    intervals = ['00-06';'06-09';'09-12';'12-14';'14-18';'18-22';'22-24'];
    starts    = temp(:,:,1);
    running   = temp(:,:,2);
    extra     = temp(:,:,3);
    [reacs, times] = size(starts);
    for time = 1:times,
        disp([' At ' intervals(time,:) '...'])
        for reac = 1:reacs,
            if running(reac,time) > 0,
                disp(['   there are ' num2str(running(reac,time)) ' ...
                    ' reactors of type ' num2str(reac) ' running' ])
            if starts(reac,time) > 0,
                disp(['       (we started ' num2str( starts(reac,time))...
                    ' additional reactors)'])
            end
            if extra(reac,time) > 0,
                disp(['       (we also produce ' num2str( extra(reac,time))...
                    ' MW more than the minimal level)'])
            end
        end
    end
end
end
end
end

% MODIFICATION LOG
%
% 051010 med   Created.
% 060109 per   Added documentation.
% 060125 per   Moved disp to end

```

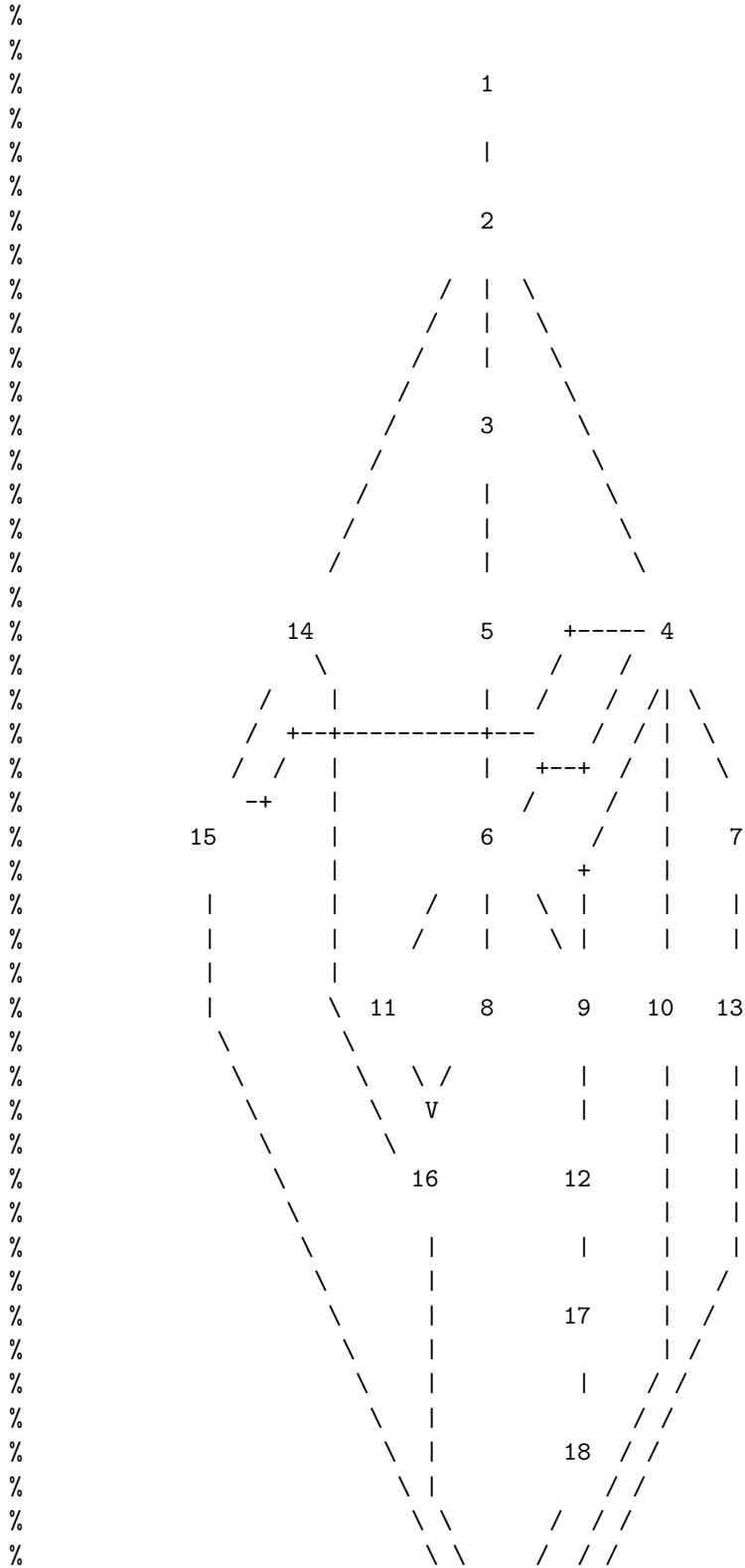
## 21 Scheduling

### 21.1 Construction of a Stadium 1

```

% function Result = constructionofastadium1Ex(PriLev)
%
% Creates a TOMLAB MIP problem for production of a stadium
%
% CONSTRUCTION OF A STADIUM 1
%
% A town council wishes to construct a small stadium in order to
% improve the services provided to the people living in the
% district. After the invitation to tender, a local construction
% company is awarded the contract and wishes to complete the task
% within the shortest possible time. All the major tasks are listed
% in the following table. The durations are expressed in weeks. Some
% tasks can only start after the completion of certain other tasks.
% The last two columns of the table refer to question 2 which we
% shall see later.
%
% Data for stadium construction
%
% +-----+-----+-----+-----+-----+
% | | | | | Max. | Add. cost per |
% |Task|Description |Duration|Predecessors|reduct.|week (in 1000$)|
% +-----+-----+-----+-----+-----+
% | 1 |Installing the construction site| 2 | none | 0 | |
% | 2 |Terracing | 16 | 1 | 3 | 30 |
% | 3 |Constructing the foundations | 9 | 2 | 1 | 26 |
% | 4 |Access roads and other networks | 8 | 2 | 2 | 12 |
% | 5 |Erecting the basement | 10 | 3 | 2 | 17 |
% | 6 |Main floor | 6 | 4,5 | 1 | 15 |
% | 7 |Dividing up the changing rooms | 2 | 4 | 1 | 8 |
% | 8 |Electrifying the terraces | 2 | 6 | 0 | |
% | 9 |Constructing the roof | 9 | 4,6 | 2 | 42 |
% | 10 |Lighting of the stadium | 5 | 4 | 1 | 21 |
% | 11 |Installing the terraces | 3 | 6 | 1 | 18 |
% | 12 |Sealing the roof | 2 | 9 | 0 | |
% | 13 |Finishing the changing rooms | 1 | 7 | 0 | |
% | 14 |Constructing the ticket office | 7 | 2 | 2 | 22 |
% | 15 |Secondary access roads | 4 | 4,14 | 2 | 12 |
% | 16 |Means of signalling | 3 | 8,11,14 | 1 | 6 |
% | 17 |Lawn and sport accessories | 9 | 12 | 3 | 16 |
% | 18 |Handing over the building | 1 | 17 | 0 | |
% +-----+-----+-----+-----+-----+
%
%
%
```

% Precedence graph of construction tasks



```

%
%          \ \ / / /
%          \  / /
%          F
%
% Question 1: (answered here)
% Which is the earliest possible date of completing the construction?
%
% Question 2: (see constructionofastadium2Ex.m )
% The town council would like the project to terminate earlier than
% the time announced by the builder (answer to question 1). To obtain
% this, the council is prepared to pay a bonus of $30K for every week
% the work finishes early. The builder needs to employ additional
% workers and rent more equipment to cut down on the total time. In
% the preceding table he has summarized the maximum number of weeks
% he can save per task (column "Max. reduct.") and the associated
% additional cost per week. When will the project be % completed if
% the builder wishes to maximize his profit?
%
% VARIABLES
%
% taskduration           The time it takes to complete each task.
% taskprecedence        Describes the order of the tasks.
%
% RESULTS
%
% x_k has the week each task is finished
% (including task_19 = finished), try this:
% Result = constructionofastadium1Ex(2);
% Result.x_k'
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%
% OUTPUT PARAMETERS
% Result           Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 10, 2005.   Last modified Jan 2, 2006.

function Result = constructionofastadium1Ex(PriLev)

```

```

if nargin < 1
    PriLev = 1;
end

taskduration = [2;16;9;8;10;6;2;2;9;5;3;2;1;7;4;3;9;1];
taskprecedence = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0;...
    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0;...
    0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0;...
    0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0;...
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0];

Prob = constructionofastadium1(taskduration, taskprecedence);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    tasks = length(taskduration) + 1; % number of tasks plus one
    [finished, task] = sort(Result.x_k);

    disp('for a best solution')
    for i = 1:tasks,
        disp([' finish task ' num2str(task(i)) ' week ' num2str(Result.x_k(task(i))) ])
    end
end

% MODIFICATION LOG
%
% 051010 med Created.
% 060111 per Added documentation.
% 060126 per Moved disp to end

```

## 21.2 Construction of a Stadium 2

```

% function Result = constructionofastadium2Ex(PriLev)
%
% Creates a TOMLAB MILP problem for production of a stadium
%
% CONSTRUCTION OF A STADIUM 2
%
% A town council wishes to construct a small stadium in order to
% improve the services provided to the people living in the
% district. After the invitation to tender, a local construction
% company is awarded the contract and wishes to complete the task
% within the shortest possible time. All the major tasks are listed
% in the following table. The durations are expressed in weeks. Some
% tasks can only start after the completion of certain other tasks.
% The last two columns of the table refer to question 2 which we
% shall see later.
%
% Data for stadium construction
%
% +-----+-----+-----+-----+-----+
% | | | | | Max. | Add. cost per |
% |Task|Description |Duration|Predecessors|reduct.|week (in 1000$)|
% +-----+-----+-----+-----+-----+
% | 1 |Installing the construction site| 2 | none | 0 | |
% | 2 |Terracing | 16 | 1 | 3 | 30 |
% | 3 |Constructing the foundations | 9 | 2 | 1 | 26 |
% | 4 |Access roads and other networks | 8 | 2 | 2 | 12 |
% | 5 |Erecting the basement | 10 | 3 | 2 | 17 |
% | 6 |Main floor | 6 | 4,5 | 1 | 15 |
% | 7 |Dividing up the changing rooms | 2 | 4 | 1 | 8 |
% | 8 |Electrifying the terraces | 2 | 6 | 0 | |
% | 9 |Constructing the roof | 9 | 4,6 | 2 | 42 |
% | 10 |Lighting of the stadium | 5 | 4 | 1 | 21 |
% | 11 |Installing the terraces | 3 | 6 | 1 | 18 |
% | 12 |Sealing the roof | 2 | 9 | 0 | |
% | 13 |Finishing the changing rooms | 1 | 7 | 0 | |
% | 14 |Constructing the ticket office | 7 | 2 | 2 | 22 |
% | 15 |Secondary access roads | 4 | 4,14 | 2 | 12 |
% | 16 |Means of signalling | 3 | 8,11,14 | 1 | 6 |
% | 17 |Lawn and sport accessories | 9 | 12 | 3 | 16 |
% | 18 |Handing over the building | 1 | 17 | 0 | |
% +-----+-----+-----+-----+-----+
%
%
%
% Precedence graph of construction tasks
%

```



```

%
%
% Question 1: (see constructionofastadium1Ex.m)
% Which is the earliest possible date of completing the construction?
%
% Question 2: (answered here)
% The town council would like the project to terminate earlier than
% the time announced by the builder (answer to question 1). To obtain
% this, the council is prepared to pay a bonus of $30K for every week
% the work finishes early. The builder needs to employ additional
% workers and rent more equipment to cut down on the total time. In
% the preceding table he has summarized the maximum number of weeks
% he can save per task (column "Max. reduct.") and the associated
% additional cost per week. When will the project be % completed if
% the builder wishes to maximize his profit?
%
% VARIABLES
%
% taskduration          The time it takes to complete each task.
% taskprecedence       Describes the order of the tasks.
%
% RESULTS
%
% For an interpretation of the results, use PriLev > 1, for example:
% Result = constructionofastadium2Ex(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 10, 2005.  Last modified Jan 2, 2006.

function Result = constructionofastadium2Ex(PriLev)

if nargin < 1
    PriLev = 1;
end

```

```

taskduration = [2;16;9;8;10;6;2;2;9;5;3;2;1;7;4;3;9;1];
taskprecedence = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0;...
    0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0;...
    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0;...
    0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0;...
    0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0;...
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0];

Prob = constructionofastadium1(taskduration, taskprecedence);
Prob.PriLevOpt = PriLev-1;
Result = tomRun('cplex', Prob, PriLev-1);

maxreduc = [0;3;1;2;2;1;1;0;2;1;1;0;2;2;1;3;0];
costperweek = [0;30;26;12;17;15;8;0;42;21;18;0;0;22;12;6;16;0]*1000;
bonus = 30000;
idx = sum(taskprecedence,2);

Prob = constructionofastadium2(maxreduc, costperweek, bonus, idx, taskduration, Result);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    tasks = length(maxreduc) + 1; % number of tasks plus one
    temp = reshape(Result.x_k,tasks,2)';
    [finished, task] = sort(temp(2,:));

    disp('for a best solution')
    for i = 1:tasks,
        if temp(1,task(i)) > 0,
            disp([' finish task ' num2str(task(i)) ' week ' ...
                num2str(temp(2,task(i))) ' (with a reduction of ' ...
                num2str(temp(1,task(i))) ')'])
        else
            disp([' finish task ' num2str(task(i)) ' week ' ...
                num2str(temp(2,task(i))) ])
        end
    end
end

```

```
    end
  end
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051010 med    Created.
```

```
% 060111 per    Added documentation.
```

```
% 060126 per    Moved disp to end
```

### 21.3 Flow-shop scheduling

```
% function Result = flowshopschedulingEx(PriLev)
%
% Creates a TOMLAB MIP problem for flow shop scheduling
%
% FLOW-SHOP SCHEDULING
%
% A workshop that produces metal pipes on demand for the automobile
% industry has three machines for bending the pipes, soldering the
% fastenings, and assembling the links. The workshop has to produce
% six pieces, for which the durations of the processing steps (in
% minutes) are given in the following table. Every workpiece first
% goes to bending, then to soldering, and finally to assembly of the
% links. Once started, any operations must be carried out without
% interruption, but the workpieces may wait between the machines.
%
% Processing durations in minutes
%
% +-----+-----+-----+
% |Workpiece|1|2|3|4|5|6|
% +-----+-----+-----+
% |Bending  |3|6|3|5|5|7|
% |Soldering|5|4|2|4|4|5|
% |Assembly |5|2|4|6|3|6|
% +-----+-----+-----+
%
% Every machine only processes one piece at a time. A workpiece may
% not overtake any other by passing onto the following machine. This
% means that if at the beginning a sequence of the workpieces is
% established, they will be processed on every machine in exactly
% this order. Which is the sequence of workpieces that minimizes the
% total time for completing all pieces?
%
% VARIABLES
%
% nummachines          The number of machines
% proctimes            Time to process a workpiece in a machine
%
% RESULTS
%
% The vector x_k can be interpreted by running the following:
% Result = flowshopschedulingEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
```

```

% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 13, 2005.  Last modified Oct 13, 2005.

function Result = flowshopschedulingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

nummachines = 3;
proctimes    = [3 6 3 5 5 7;...
                5 4 2 4 4 5;...
                5 2 4 6 3 6];

Prob = flowshopscheduling(nummachines, proctimes);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    p      = size(proctimes,2);          % number of pieces
    m      = nummachines;               % number of machines
    order  = reshape(Result.x_k(1:p*p),p,p); % the order of the pieces
    order(find(order<0.5)) = 0;         % delete bad zeros
    wait1  = Result.x_k(p*p+1:p*p+p);   % wait before machine 1
    wait2  = Result.x_k(end-2*p+1:end-p); % wait before machine 2
    wait3  = Result.x_k(end-p+1:end);   % wait before machine 3
    proctimes    = [3 6 3 5 5 7;...      % the processing times
                    5 4 2 4 4 5;...
                    5 2 4 6 3 6];
    sequence = [] ;                      % the sequence of pieces
    for i = 1:p,
        sequence = [sequence find(order(:,i))]; % insert piece by piece
    end
    disp(['A best sequence is: ' num2str(sequence)])

    mt1 = []; % empty machine-times for machine 1
    mt2 = []; % empty machine-times for machine 2
    mt3 = []; % empty machine-times for machine 3

```

```

t11 = 0;    % timepoint for entering machine 1
t12 = 0;    % timepoint for exiting machine 1
t21 = 0;    % timepoint for entering machine 2
t22 = 0;    % timepoint for exiting machine 2
t31 = 0;    % timepoint for entering machine 3
t32 = 0;    % timepoint for exiting machine 3
first2 = 1; % the first piece in machine 2
first3 = 1; % the first piece in machine 3

for i = 1:length(sequence),
    id = sequence(i);

    % times in and out of machine 1
    t11 = t12 + wait1(id);
    t12 = t11 + proctimes(1,id);

    % times in and out of machine 2
    if first2 == 1,
        t21 = t12 + wait2(id);
        first2 = 0 ;
    else,
        t21 = t22 + wait2(id);
    end
    t22 = t21 + proctimes(2,id);

    % times in and out of machine 3
    if first3 == 1,
        t31 = t22 + wait3(id);
        first3 = 0 ;
    else,
        t31 = t32 + wait3(id);
    end
    t32 = t31 + proctimes(3,id);

    % times in and out of machines
    mt1 = [mt1; t11 t12];
    mt2 = [mt2; t21 t22];
    mt3 = [mt3; t31 t32];
end

mt = [mt1 mt2 mt3];

disp('FLOW FOR THE PIECES')
for j = 1:p,
    disp(['piece ' num2str(sequence(j)) ' has this flow' ])
    for k = 1:m,
        disp([' machine ' num2str(k) ': ' num2str(mt(j,(k-1)*2+1)) '-' num2str(mt(j,(k-1)*2+2)) ])
    end
end

```

```

    end
end
disp('FLOW FOR THE MACHINES')
for k = 1:m,
    disp(['machine ' num2str(k) ' has this flow' ])
    for l = 1:p,
        disp([' piece ' num2str(sequence(l)) ': ' ...
            num2str(mt(l,((k-1)*2+1))) '-' num2str(mt(l,((k-1)*2+2))])])
    end
end
end

end

% MODIFICATION LOG
%
% 051010 med    Created.
% 060111 per    Added documentation.
% 060124 per    Interpretation of results upgraded.
% 060126 per    Moved disp to end

```

## 21.4 Job Shop Scheduling

```

% function Result = jobshopschedulingEx(PriLev)
%
% Creates a TOMLAB MIP problem for job shop scheduling
%
% JOB SHOP SCHEDULING
%
% A company has received an order for three types of wallpapers: one
% (paper 1) has a blue background with yellow patterns, another
% (paper 2) a green background and blue and yellow patterns, and the
% last (paper 3) has a yellow background with blue and green
% patterns. Every paper type is produced as a continuous roll of
% paper that passes through several machines, each printing a
% different color. The order in which the papers are run through the
% machines depends on the design of the paper: for paper 1 first the
% blue background and then the yellow pattern is printed. After the
% green background for paper 2, first the blue and then the yellow
% patterns are printed. The printing of paper 3 starts with the
% yellow background, followed by the blue and then the green
% patterns.
%
%
%           p2           p1           p3
%           |           |           |
%           v           v           v
% +-----+ +-----+ --p1-> +-----+
% |GREEN| --p2-> |BLUE| --p2-> |YELLOW|
% +-----+ <-p3-- +-----+ <-p3-- +-----+
%
%           |           |           |
%           v           v           v
%           p3           p1           p2
%
% Production flows through printing machines
%
% The processing times differ depending on the surface that needs to
% be printed. The times (in minutes) for applying every color of the
% three paper types are given in the following table.
%
% Times required for applying every color
%
% +-----+-----+-----+-----+-----+
% |Machine|Color |Paper 1|Paper 2|Paper 3|
% +-----+-----+-----+-----+-----+
% | 1     |Blue  | 45    | 20    | 12    |
% | 2     |Green | -     | 10    | 17    |
% | 3     |Yellow| 10    | 34    | 28    |

```

```

% +-----+-----+-----+-----+-----+
%
% Knowing that every machine can only process one wallpaper at a time
% and that a paper cannot be processed by several machines
% simultaneously, how should the paper printing be scheduled on the
% machines in order to finish the order as early as possible?
%
% VARIABLES
%
% the colors are ordered as follows:
%   color1 = blue   (paper 1's first color)
%   color2 = green  (paper 2's first color)
%   color3 = yellow (paper 3's first color)
%
%
% proctimes           Times for the papers in the machines
% flow                The order of colors on the paperS
% final               The last machine for each paper
% bigM                The total processingtimes.
%
% RESULTS
%
% run this for explanation of x_k
% Result = jobshopschedulingEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev             Print Level
%
% OUTPUT PARAMETERS
% Result             Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Jan 2, 2006.   Last modified Jan 2, 2006.

function Result = jobshopschedulingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

proctimes = [45 20 12;...

```

```

    0 10 17;...
    10 34 28];

flow      = [1 3 0;...
            2 1 3;...
            3 1 2];

final     = [3;3;2];

bigM      = sum(sum(proctimes));

Prob = jobshopscheduling(proctimes, flow, final, bigM);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    c      = 3; % number of colors
    p      = 3; % number of papers
    temp   = reshape(Result.x_k,c,c+c+2);
    disp(['all papers are finished after ' num2str(max(temp(:,c+1))) ' minutes'])
    disp(' ')
    for j = 1:p,
        disp(['paper ' num2str(j) ':'])
        colortimes = temp(j,1:c);
        [time,col] = sort(colortimes);
        for i = 1:c,
            this_col = col(i);
            this_time = time(i);
            if this_time == 0
                disp([' starts using color ' num2str(this_col) ' after ' ...
                    num2str(this_time) ' minutes (or not at all)'])
            else
                disp([' starts using color ' num2str(this_col) ' after ' ...
                    num2str(this_time) ' minutes'])
            end
        end
    end
end
end

% MODIFICATION LOG
%
% 060102 med Created.
% 060111 per Added documentation.
% 060126 per Moved disp to end

```

## 21.5 Sequencing jobs on a bottleneck machine

```

% function Result = sequencingjobsonabottleneckmEx(PriLev)
%
% Creates a TOMLAB MIP problem for sequencing jobs on a bottleneck machine
%
% SEQUENCING JOBS ON A BOTTLENECK MACHINE
%
% In workshops it frequently happens that a single machine determines
% the throughput of the entire production (for example, a machine of
% which only one is available, the slowest machine in a production
% line, etc.). This machine is called the critical machine or the
% bottleneck. In such a case it is important to schedule the tasks
% that need to be performed by this machine in the best possible way.
% The aim of the problem is to provide a simple model for scheduling
% operations on a single machine and that may be used with different
% objective functions. We shall see here how to minimize the total
% processing time, the average processing time, and the total
% tardiness. A set of tasks (or jobs) is to be processed on a single
% machine. The execution of tasks is non-preemptive (that is, an
% operation may not be interrupted before its completion). For every
% task i its release date and duration are given. For the last
% optimization criterion (total tardiness), a due date (latest
% completion time) is also required to measure the tardiness, that
% is, the amount of time by which the completion of jobs exceeds
% their respective due dates. The following table lists the data for
% our problem.
%
% What is the optimal value for each of the objectives:
%   1 - minimizing the total duration of the schedule (= makespan)?
%   2 - minimizing the mean processing time?
%   3 - minimizing the total tardiness?
%
% Task time windows and durations
%
% +-----+-----+-----+-----+-----+-----+
% |Job      | 1| 2| 3| 4| 5| 6| 7|
% +-----+-----+-----+-----+-----+-----+
% |Release date| 2| 5| 4| 0| 0| 8| 9|
% |Duration   | 5| 6| 8| 4| 2| 4| 2|
% |Due date   |10|21|15|10| 5|15|22|
% +-----+-----+-----+-----+-----+-----+
%
% VARIABLES
%
% releasedate      Release dates of jobs (row 1 in table)
% duration         Time to perform a job (row 2 in table)

```

```

% duedate                Deadline of each job (row 3 in table)
%
% RESULTS
%
% For an interpretation of the results try the following:
% [Result1, Result2, Result3] = sequencingjobsonabottleneckmEx(2)
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Jan 2, 2006.  Last modified Jan 2, 2006.

function [Result1,Result2,Result3] = sequencingjobsonabottleneckmEx(PriLev)

if nargin < 1
    PriLev = 1;
end

releasedate = [2 5 4 0 0 8 9]';
duration    = [5 6 8 4 2 4 2]';
duedate     = [10 21 15 10 5 15 22]';

Prob1 = sequencingjobsonabottleneckm(releasedate, duration, duedate, 1);
Result1 = tomRun('cplex', Prob1, PriLev);

Prob2 = sequencingjobsonabottleneckm(releasedate, duration, duedate, 2);
Result2 = tomRun('cplex', Prob2, PriLev);

Prob3 = sequencingjobsonabottleneckm(releasedate, duration, duedate, 3);
Result3 = tomRun('cplex', Prob3, PriLev);

if PriLev > 1,
    j          = length(releasedate); % number of jobs
    sequence2 = reshape(Result2.x_k,j,j+2); % results from 1 and 2
    sequence3 = reshape(Result3.x_k,j,j+3); % results from 3
    sequence2(find(sequence2(1:7*7)<0.1)) = 0; % remove bad zeros
    sequence3(find(sequence3(1:7*7)<0.1)) = 0; % remove bad zeros

```

```

s2 = []; % blank sequence
s3 = []; % blank sequence
for t = 1:j,
    s2 = [s2 find(sequence2(t,1:j))];
    s3 = [s3 find(sequence3(t,1:j))];
end
disp(['An order to minimize duration (= ' ...
    num2str(sequence2(j,j+2)) ') and to minimize mean '...
    'processing time (= ' num2str(sequence2(j,j+2)/j) ')'])
disp(num2str(s2))
disp(' ')
tard = sum(sequence3(:,size(sequence3,2)));
disp(['An order to minimize tardiness (= ' num2str(tard) ') ' ])
disp(num2str(s3))
disp(' ')
end

% MODIFICATION LOG
%
% 060102 med Created.
% 060111 per Added documentation.
% 060126 per Moved disp to end

```

## 21.6 Paint production

```
% function Result = paintproductionEx(PriLev)
%
% Creates a TOMLAB MIP problem for paint production
%
% PAINT PRODUCTION
%
% As a part of its weekly production a paint company produces five
% batches of paints, always the same, for some big clients who have a
% stable demand. Every paint batch is produced in a single production
% process, all in the same blender that needs to be cleaned between
% two batches. The durations of blending paint batches 1 to 5 are
% respectively 40, 35, 45, 32, and 50 minutes. The cleaning times
% depend on the colors and the paint types. For example, a long
% cleaning period is required if an oil-based paint is produced after
% a water-based paint, or to produce white paint after a dark color.
% The times are given in minutes in the following table CLEAN where
% CLEANij denotes the cleaning time between batch i and batch j.
%
% Matrix of cleaning times
%
% +-----+-----+
% | 1| 2| 3| 4| 5|
% +-----+-----+
% |1| 0|11| 7|13|11|
% |2| 5| 0|13|15|15|
% |3|13|15| 0|23|11|
% |4| 9|13| 5| 0| 3|
% |5| 3| 7| 7| 7| 0|
% +-----+-----+
%
% Since the company also has other activities, it wishes to deal
% with this weekly production in the shortest possible time (blending
% and cleaning). Which is the corresponding order of paint batches?
% The order will be applied every week, so the cleaning time between
% the last batch of one week and the first of the following week
% needs to be counted for the total duration of cleaning.
%
% VARIABLES
%
% cleantimes          Times to clean from batch i to j
% prodtimes          Production times per batch
%
% RESULTS
%
```

```

% For an interpretation of the results, use PriLev > 1, for example:
% Result = paintproductionEx(2);
%
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 17, 2005.  Last modified Jan 2, 2006.

function Result = paintproductionEx(PriLev)

if nargin < 1
    PriLev = 1;
end

cleantimes = [ 0 11  7 13 11;...
              5  0 13 15 15;...
              13 15  0 23 11;...
              9 13  5  0  3;...
              3  7  7  7  0];

prodtimes = [40;35;45;32;50];

Prob = paintproduction(prodtimes, cleantimes);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    batches = size(cleantimes,1);           % number of batches
    temp1   = reshape(Result.x_k,batches,batches+1); % reshape x_K
    link    = [];                           % connections

    for i = 1:batches,
        for j = 1:batches,
            if temp1(i,j) == 1
                link = [[i j ]; link ];    % finding connections
            end
        end
    end
end

```

```

        end
    end
end

first = link(1:1);           % start batch
next = link(1,2);          % next batch
order = [first];           % ordered batches

for k = 1:batches,
    order = [order next];   % adding next
    next = link(find(link(:,1)==next),2); % finding new next
end
disp(['one best order: ' num2str(order)]) % display solution
end

% MODIFICATION LOG
%
% 051010 med   Created.
% 060111 per   Added documentation.
% 060126 per   Moved disp to end

```

## 21.7 Assembly line balancing

```

% function Result = assemblylinebalancingEx(PriLev)
%
% Creates a TOMLAB MIP problem for assembly line balancing
%
% Assembly line balancing
%
% An electronics factory produces an amplifier on an assembly line
% with four workstations. An amplifier is assembled in twelve
% operations between which there are certain precedence constraints.
% The following table indicates the duration of every task
% (in minutes) and the list of its immediate predecessors (the
% abbreviation PCB used in this table stands for printed circuit
% board).
%
% The production manager would like to distribute the tasks among the
% four workstations, subject to the precedence constraints, in order
% to balance the line to obtain the shortest possible cycle time,
% that is, the total time required for assembling an amplifier. Every
% task needs to be assigned to a single workstation that has to
% process it without interruption. Every workstation deals with a
% single operation at a time. We talk about cycle time because the
% operations on every workstation will be repeated for every
% amplifier. When an amplifier is finished, the amplifiers at
% stations 1 to 3 advance by one station, and the assembly of a new
% amplifier is started at the first workstation.
%
% List of tasks and predecessors
% +-----+-----+-----+-----+
% |Task|Description                |Duration|Predecessors|
% +-----+-----+-----+-----+
% | 1 |Preparing the box          | 3      |             |
% | 2 |PCB with power module      | 6      | 1           |
% | 3 |PCB with pre-amplifier     | 7      | 1           |
% | 4 |Filter of the amplifier    | 6      | 2           |
% | 5 |Push-pull circuit          | 4      | 2           |
% | 6 |Connecting the PCBs        | 8      | 2,3        |
% | 7 |Integrated circuit of the pre-amplifier| 9      | 3           |
% | 8 |Adjusting the connections  | 11     | 6           |
% | 9 |Heat sink of the push-pull | 2      | 4,5,8      |
% |10 |Protective grid            | 13     | 8,11       |
% |11 |Electrostatic protection   | 4      | 7           |
% |12 |Putting on the cover       | 3      | 9,10       |
% +-----+-----+-----+-----+
%
% A precedence graph of the same table with times within ( ).

```

```

%
%           1 (3)
%
%          / \
%         /   \
%
%        3 (7)   2 (6)
%
%       / \   / | \
%      /   \ /  |  \
%
%     7 (9)   6   5   4 (6)
%             (8) (4)
%            |   |   |
%            |   |   |
%            |   |   |
%           11 (4)  8   |   |
%                (11)|   /
%               \ /   \ | /
%               \ /   \ | /
%
%            10 (13)  9 (2)
%
%           \   /
%            \ /
%
%           12 (3)
%
% VARIABLES
%
% duration           Time or each task.
% stations           Stations available.
% dependsvec1 and 2  For a task t: If dependsvec1[i] = t and
%                    dependsvec2[i] = d then k depends on d.
%
% RESULTS
%
% To get the results interpreted:
% Result = assemblylinebalancingEx(2);
%
% REFERENCES
%
```

```

% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 17, 2005.  Last modified Jan 2, 2005.

function Result = assemblylinebalancingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

duration    = [3;6;7;6;4;8;9;11;2;13;4;3];
stations    = 4;
dependsvec1  = [2;3;4;5;6;6;7;8;9;9;9;10;10;11;12;12];
dependsvec2  = [1;1;2;2;2;3;3;6;4;5;8; 8;11; 7; 9;10];

Prob = assemblylinebalancing(stations, duration, dependsvec1, dependsvec2);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    t      = length(duration);           % number of tasks
    s      = stations;                  % number of workstations
    temp   = reshape(Result.x_k(1:t*s),t,s); % reshaping distribution
    time   = Result.x_k(t*s+1);         % total time

    for i = 1:s,
        disp(['tasks managed by station ' ... % filter + disp
              num2str(i) ': ' ...
              num2str(find(temp(:,i)))'])
    end

    disp(['total time ' num2str(time)])    % disp the time
end

% MODIFICATION LOG
%
% 051010 med   Created
% 060111 per   Added documentation.

```

% 060126 per Moved disp to end

## 22 Planning

### 22.1 Planning the production of bicycles

```
% function Result = planningtheprodofbicyclesEx(PriLev)
%
% Creates a TOMLAB MIP problem for planning the production of bicycles
%
% PLANNING THE PRODUCTION OF BICYCLES
%
% A company produces bicycles for children. The sales forecast in
% thousand of units for the coming year are given in the following
% table. The company has a capacity of 30,000 bicycles per month.
% It is possible to augment the production by up to 50% through
% overtime working, but this increases the production cost for a
% bicycle from the usual $ 32 to $ 40.
%
% Sales forecasts for the coming year in thousand units
%
% +---+---+---+---+---+---+---+---+---+---+
% |Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec|
% +---+---+---+---+---+---+---+---+---+---+
% | 30| 15| 15| 25| 33| 40| 45| 45| 26| 14| 25| 30|
% +---+---+---+---+---+---+---+---+---+---+
%
% Currently there are 2,000 bicycles in stock. The storage costs have
% been calculated as $ 5 per unit held in stock at the end of a month.
% We assume that the storage capacity at the company is virtually
% unlimited (in practice this means that the real capacity, that is
% quite obviously limited, does not impose any limits in our case).
% We are at the first of January. Which quantities need to be
% produced and stored in the course of the next twelve months in order
% to satisfy the forecast demand and minimize the total cost?
%
% VARIABLES
%
% normcapacity          the normal production capacity
% extracapacity         extra capacity
% normcost              normal cost
% extracost             cost per bike if overtime
% demand               bikes wanted per month
% startstock           bikes in store
% storagecost          cost to have a bike in store one month
%
% RESULTS
%
% For an interpretation of the results, try this:
```

```

% Result = planningtheprodofbicyclesEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 17, 2005.  Last modified Oct 17, 2005.

function Result = planningtheprodofbicyclesEx(PriLev)

if nargin < 1
    PriLev = 1;
end

normcapacity    = 30000;
extracapacity   = 15000;
normcost        = 32;
extracost       = 40;
demand          = [30;15;15;25;33;40;45;45;26;14;25;30]*1000;
startstock      = 2000;
storagecost     = 5;

Prob = planningtheprodofbicycles(normcapacity, extracapacity, normcost,...
    extracost, demand, startstock, storagecost);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    months = length(demand);
    temp = reshape(Result.x_k,months,3);

    for i = 1:months,
        disp(['Month ' num2str(i) ':' ])
        disp([' produce ' num2str(temp(i,1)) ' regular bikes' ])
        if temp(i,2) > 0,
            disp([' and ' num2str(temp(i,2)) ' extras' ])
        end
    end
end

```

```
    end
    if temp(i,3) > 0,
        disp([' let      ' num2str(temp(i,3)) ' be stored' ])
    end
end
```

```
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051017 med Created.
```

```
% 060110 per Added documentation.
```

```
% 060125 per Moved disp to end
```

## 22.2 Production of drinking glasses

```

% function Result = productionofdrinkinglassesEx(PriLev)
%
% Creates a TOMLAB MIP problem for production of drinking glasses
%
% PRODUCTION OF DRINKING GLASSES
%
% The main activity of a company in northern France is the production
% of drinking glasses. It currently sells six different types
% (V1 to V6), that are produced in batches of 1000 glasses, and wishes
% to plan its production for the next 12 weeks. The batches may be
% incomplete (fewer than 1000 glasses). The demand in thousands for
% the 12 coming weeks and for every glass type is given in the
% following table.
%
% Demands for the planning period (batches of 1000 glasses)
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Week| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|12|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |V1  |20|22|18|35|17|19|23|20|29|30|28|32|
% |V2  |17|19|23|20|11|10|12|34|21|23|30|12|
% |V3  |18|35|17|10| 9|21|23|15|10| 0|13|17|
% |V4  |31|45|24|38|41|20|19|37|28|12|30|37|
% |V5  |23|20|23|15|10|22|18|30|28| 7|15|10|
% |V6  |22|18|20|19|18|35| 0|28|12|30|21|23|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% For every glass type the initial stock is known, as well as the
% required final stock level (in thousands). Per batch of every glass
% type, the production and storage costs in $ are given, together
% with the required working time for workers and machines (in hours),
% and the required storage space (measured in numbers of trays).
%
% The number of working hours of the personnel is limited to 450
% hours per week, and the machines have a weekly capacity of 850
% hours. Storage space for up to 1000 trays is available. Which
% quantities of the different glass types need to be produced in
% every period to minimize the total cost of production and storage?
%
% Data for the six glass types
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% | |Production|Storage|Initial|Final|           |           |Storage|
% | | cost     | cost  | stock |stock|Timeworker|Timemachine| space |
% +-----+-----+-----+-----+-----+-----+-----+-----+

```

```

% |V1| 100 | 25 | 50 | 10 | 3 | 2 | 4 |
% |V2| 80 | 28 | 20 | 10 | 3 | 1 | 5 |
% |V3| 110 | 25 | 0 | 10 | 3 | 4 | 5 |
% |V4| 90 | 27 | 15 | 10 | 2 | 8 | 6 |
% |V5| 200 | 10 | 0 | 10 | 4 | 11 | 4 |
% |V6| 140 | 20 | 10 | 10 | 4 | 9 | 9 |
% +-----+-----+-----+-----+-----+
%
% VARIABLES
%
% demand          Weekly demand of V1-V6
% workermax       The staffs weekly capacity in hours
% machinemax      The machines weekly capacity in hours
% maxstorage      Maximal storage space
% productioncost  Cost to produce a batch of a glasstype.
% storagecost     Cost to store a batch of a glasstype
% initialstock    Initial stock of glasstypes
% finalstock      Required final stock
% timeworker      Personnel-time required for a batch
% timemachine     Machine-time required for a batch
% storagespace    Space required by batch in storage
%
% RESULTS
%
% for an interpretation of the results, try
% Result = productionofdrinkingglassesEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 17, 2005. Last modified Oct 17, 2005.

function Result = productionofdrinkingglassesEx(PriLev)

if nargin < 1
    PriLev = 1;
end

```

```

demand          = [20 22 18 35 17 19 23 20 29 30 28 32;...
    17 19 23 20 11 10 12 34 21 23 30 12;...
    18 35 17 10  9 21 23 15 10  0 13 17;...
    31 45 24 38 41 20 19 37 28 12 30 37;...
    23 20 23 15 10 22 18 30 28  7 15 10;...
    22 18 20 19 18 35  0 28 12 30 21 23];

workermax       = 450; % Modified from 390, otherwise infeasible
machinemax      = 850;
maxstorage      = 1000;

productioncost  = [100;80;110;90;200;140];
storagecost     = [25;28;25;27;10;20];
initialstock    = [50;20;0;15;0;10];
finalstock      = [10;10;10;10;10;10];
timeworker      = [3;3;3;2;4;4];
timemachine     = [2;1;4;8;11;9];
storagespace    = [4;5;5;6;4;9];

Prob = productionofdrinkingglasses(demand, workermax, machinemax, maxstorage...
    , productioncost, storagecost, initialstock, finalstock, timeworker...
    , timemachine, storagespace);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,

    [g,w] = size(demand); % g = glass_types, W = weeks

    temp = reshape(Result.x_k,g,2,w);

    for i = 1:w,
        disp(['results for week ' num2str(i) ':'])
        for j = 1:g,
            if temp(j,1,i) > 0,
                disp(['    ' num2str(temp(j,1,i)) ' batches of type ' num2str(j) ' should be produced' ])
            end
            if temp(j,2,i) > 0,
                disp(['    ' num2str(temp(j,2,i)) ' batches of type ' num2str(j) ' should be stored to next month'])
            end
        end
    end
end

% MODIFICATION LOG
%
```

% 051017 med Created.  
% 060109 per Added documentation.  
% 060125 per Moved disp to end

## 22.3 Material Requirement Planning

```

% function Result = materialrequirementplanningEx(PriLev)
%
% Creates a TOMLAB MIP problem for material requirement planning
%
% MATERIAL REQUIREMENT AND PLANNING
%
% The company Minorette produces two types of large toy lorries for
% children: blue removal vans and red tank lorries. Each of these
% lorries is assembled from thirteen items. See below for the
% breakdown of components and the table below for the prices of the
% components.
%
% Prices of components
%
%+-----+-----+-----+-----+-----+-----+
%| Wheel  |Steel bar  |Bumper  | Chassis | Cabin  |Door window|
%+-----+-----+-----+-----+-----+-----+
%| $ 0.30 | $ 1      |$ 0.20  | $ 0.80  | $ 2.75 | $ 0.10  |
%+-----+-----+-----+-----+-----+-----+
%|Windscreen|Blue container|Red tank|Blue motor|Red motor| Headlight |
%+-----+-----+-----+-----+-----+-----+
%| $ 0.29 | $ 2.60  | $ 3    | $ 1.65  | $ 1.65 | $ 0.15  |
%+-----+-----+-----+-----+-----+-----+
%
%
% Breakdown of components (Gozinto graph)
%
%
%                               Blue lorry
%                               (or red)
%                               |
%                               |
%                               |
%  +-----+-----+-----+-----+
%  |         |         |         |         |
%  1|         | 1|         | 1|         | 1|         | 2|
%  |         |         |         |         |         |         |
%  Assembled  Blue container  Assembled  Blue motor  Headlight
%  chassis    (or red tank)   cabin      (or red)
%  |         |         |         |         |
%  |         |         |         |         |
%  +-----+-----+-----+-----+
%  |         |  |  |         |  |  |         |
%  2|         | 2|  | 1|         | 1|  | 2|         | 1|
%  |         |  |  |         |  |  |         |  |
%  Bumper  Axle  Chassis  Cabin  Door  Windscreen

```

```

%           |                window
%           |
%           +---+---+
%           |       |
%           2|     1|
%           |       |
%           Wheel Steel bar
%

```

% The subsets (axles, chassis, blue or red cabin) may be assembled by the company using the components, or subcontracted. The following table lists the costs for assembling and for subcontracting these subsets, together with the capacities of the company. The assembly costs do not take into account the buying prices of their components.

% Subcontracting and assembly costs, assembly capacities

```

% +-----+-----+-----+-----+-----+-----+
% |           | Axle |Assembled chassis|Assembled cabin|Blue lorry|Red lorry|
% +-----+-----+-----+-----+-----+-----+
% |Subcontracting|$12.75|      $ 30      |      $ 3      |      -      |      -      |
% |Assembly      |$6.80 |      $ 3.55    |      $ 3.20    |      $ 2.20  |      $ 2.60  |
% |Capacity      | 600  |      4000     |      3000     |      4000   |      5000   |
% +-----+-----+-----+-----+-----+-----+
%

```

% For the next month, Minorette has the same demand forecast of 3000 pieces for the two types of lorries. At present, the company has no stock. Which quantities of the different items should Minorette buy or subcontract to satisfy the demand while minimizing the production cost?

% VARIABLES

```

% demand           The demand for tank and container lorries
% compprices       The price of the components
% subcontr         Cost for using a subcontracter to assemble
% assembly         Price for own assembly
% finalassembly    Price for final assembly
% capacity         Capacity to assemble components
% finalcapacity    Capacity to assemble final step
% assemmat         12 first columns: component prices
%                  next 3 columns: subcontracter price
%                  next 3 columns: own assembly price
%                  last 2 columns: final assembly
%

```

% RESULTS

%

```

% For an interpretation of the results, try:
% Result          = materialrequirementplanningEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 18, 2005.   Last modified Oct 18, 2005.

```

```
function Result = materialrequirementplanningEx(PriLev)
```

```

if nargin < 1
    PriLev = 1;
end

```

```

demand          = [3000;3000];
compprices      = [.3;1;.2;.8;2.75;.1;.29;2.6;3;1.65;1.65;.15];
subcontr        = [12.75;30;3];
assembly        = [6.8;3.55;3.20];
finalassembly   = [2.2;2.6];
capacity        = [600;4000;3000];
finalcapacity   = [4000;5000];

```

```
%Buy preprod (12), Buy subcontr (3), assemble (3), finalass (2)
```

```

assemmat        = [ 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 -2  0  0  0  0;...
    0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0 -1  0  0  0  0;...
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1 -2  0  0  0;...
    0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 -2  0  0  0;...
    0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0 -1  0  0  0;...
    0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0 -1  0  0;...
    0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0 -2  0  0;...
    0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0 -1  0  0;...
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0 -1 -1;...
    0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0 -1  0;...
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1 -1 -1;...
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1 -1 -1;...
    0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0 -1  0;...
    0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0 -1;...

```

```

    0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 -2 -2];

Prob = materialrequirementplanning(demand, compprices, assemmat...
    , subcontr, assembly, finalassembly, capacity, finalcapacity);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    1           = length(compprices);
    12          = length(subcontr);
    13          = length(assembly);
    14          = length(finalassembly);
    buycomponent = Result.x_k(1       : 1);
    buysubcontracter = Result.x_k(1+1   : 1+12);
    buyassembly    = Result.x_k(1+1+12 : 1+12+13);
    buyfinalassembly = Result.x_k(1+1+12+13 : 1+12+13+14);
    disp('Buy these components:')
    for i = 1:1,
        if buycomponent(i) > 0,
            disp(['    ' num2str(buycomponent(i)) ' units of type ' num2str(i) ])
        end
    end
    disp('Use these subcontracters:')
    for i = 1:12,
        if buysubcontracter(i) > 0,
            disp(['    ' num2str(buysubcontracter(i)) ' assemblies of type ' num2str(i) ])
        end
    end
    disp('Do this assembly:')
    for i = 1:13,
        if buyassembly(i) > 0,
            disp(['    ' num2str(buyassembly(i)) ' assemblies of type ' num2str(i) ])
        end
    end
    disp('Do this final assembly:')
    for i = 1:14,
        if buyfinalassembly(i) > 0,
            disp(['    ' num2str(buyfinalassembly(i)) ' assemblies of type ' num2str(i) ])
        end
    end
end

% MODIFICATION LOG
%
% 051018 med    Created.
% 060110 per    Added documentation.
% 060125 per    Moved disp to end

```

## 22.4 Planning the production of electronic components

```

% function Result = planningtheprodofoeleccompEx(PriLev)
%
% Creates a TOMLAB MIP problem for planning the production of
% electronic components
%
% PLANNING THE PRODUCTION OF ELECTRONIC COMPONENTS
%
% To augment its competitiveness a small business wishes to improve
% the production of its best selling products. One of its main
% activities is the production of cards with microchips and
% electronic badges. The company also produces the components for
% these cards and badges. Good planning of the production of these
% components therefore constitutes a decisive success factor for the
% company. The demand for components is internal in this case and
% hence easy to anticipate.
%
% For the next six months the production of four products with
% references X43-M1, X43-M2, Y54-N1, Y54-N2 is to be planned. The
% production of these components is sensitive to variations of the
% level of production, and every change leads to a non-negligible
% cost through controls and readjustments. The company therefore
% wishes to minimize the cost associated with these changes whilst
% also taking into account the production and storage costs.
%
% The demand data per time period, the production and storage costs,
% and the initial and desired final stock levels for every product
% are listed in the following table. When the production level
% changes, readjustments of the machines and controls have to be
% carried out for the current month. The cost incurred is
% proportional to the increase or reduction of the production
% compared to the preceding month. The cost for an increase of the
% production is $ 1 per unit but only $ 0.50 for a decrease of the
% production level.
%
% Data for the four products
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |           Demands           |           Cost           |           Stock           |
% +-----+-----+-----+-----+-----+-----+-----+-----+
% | Month| 1 | 2 | 3 | 4 | 5 | 6 | Production|Storage|Initial|Final|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |X43-M1|1500|3000|2000|4000|2000|2500|    20    |  0.4  |   10  |   50  |
% |X43-M2|1300| 800| 800|1000|1100| 900|    25    |  0.5  |    0  |   10  |
% |Y54-N1|2200|1500|2900|1800|1200|2100|    10    |  0.3  |    0  |   10  |
% |Y54-N2|1400|1600|1500|1000|1100|1200|    15    |  0.3  |    0  |   10  |

```

```

% +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
%
% What is the production plan that minimizes the sum of costs
% incurred through changes of the production level, production
% and storage costs?
%
% VARIABLES
%
% demand                the demand for each component and month
% prodcost               Cost to produce a component
% storagecost           Cost to store a component
% initialstock          Initial stock
% finalstock            Final stock
% increasecost, decreasecost  Increase or decrease in cost
%                        when producing more or less this month
%                        than last month.
%
% RESULTS
%
% for an interpretation of the results, try:
% Result      = planningtheprodoeleccompEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 18, 2005.  Last modified Feb 3, 2006.

function Result = planningtheprodoeleccompEx(PriLev)

if nargin < 1
    PriLev = 1;
end

demand      = [1500 3000 2000 4000 2000 2500;...
              1300  800  800 1000 1100  900;...
              2200 1500 2900 1800 1200 2100;...
              1400 1600 1500 1000 1100 1200];

```

```

prodcost      = [20;25;10;15];
storagecost   = [.4;.5;.3;.3];
initialstock  = [10;0;50;0];
finalstock    = [50;10;30;10];
increasecost  = 1;
decreasecost  = 0.5;

Prob = planningtheprodofeleccomp(demand, prodcost,...
    storagecost, initialstock, finalstock, increasecost, decreasecost);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    l1      = 10; % 4 components + 4 stockslots + increase + decrease
    l2      = 6; % the number of time segments
    temp    = reshape(Result.x_k,l1,l2);
    for month = 1:l2,
        disp(['Solution for month ' num2str(month)])
        disp(['produce   ' num2str(temp( 1: 4,month)')])
        disp(['stock    ' num2str(temp( 5: 8,month)')])
        disp(['increase ' num2str(temp( 9: 9,month)')])
        disp(['decrease ' num2str(temp(10:10,month)')])
        disp(' ')
    end
end

% MODIFICATION LOG
%
% 051018 med   Created.
% 060110 per   Added documentation.
% 060125 per   Moved disp to end
% 060203 med   Removed printing of temp

```

## 22.5 Planning the Production of Fiberglass

```

% function Result = planningtheprodooffiberglassEx(PriLev)
%
% Creates a TOMLAB MIP problem for planning the production of
% fiber glass
%
% PLANNING THE PRODUCTION OF FIBERGLASS
%
% A company produces fiberglass by the cubic meter and wishes to plan
% its production for the next six weeks. The production capacity is
% limited, and this limit takes a different value in every time
% period. The weekly demand is known for the entire planning period.
% The production and storage costs also take different values
% depending on the time period. All data are listed in the following
% table.
%
% Data per week
%
% +---+-----+-----+-----+-----+
% |   | Production |Demand|Production| Storage  |
% |Week|capacity(m3)| (m3) |cost($/m3)|cost ($/m3)|
% +---+-----+-----+-----+-----+
% | 1 |    140    | 100 |    5    |    0.2    |
% | 2 |    100    | 120 |    8    |    0.3    |
% | 3 |    110    | 100 |    6    |    0.2    |
% | 4 |    100    |  90 |    6    |    0.25   |
% | 5 |    120    | 120 |    7    |    0.3    |
% | 6 |    100    | 110 |    6    |    0.4    |
% +---+-----+-----+-----+-----+
%
% Which is the production plan that minimizes the total cost of
% production and storage?
%
% VARIABLES
%
% capacity          Production capacity over time
% demand           Demand over time
% prodcost         Cost to produce over time
% storcost         Cost to store over time
%
% RESULTS
%
% For an interpretation of the results, try:
% Result = planningtheprodooffiberglassEx(2);
%
% REFERENCES

```

```

%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 18, 2005.  Last modified Oct 18, 2005.

function Result = planningtheprodooffiberglassEx(PriLev)

if nargin < 1
    PriLev = 1;
end

capacity      = [140;100;110;100;120;100];
demand        = [100;120;100; 90;120;110];
prodcost      = [ 5; 8; 6; 6; 7; 6];
storcost      = [ .2; .3; .2;.25; .3; .4];

Prob = planningtheprodooffiberglass(capacity, demand,...
    prodcost, storcost);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    times = length(capacity);

    produce = Result.x_k(1:times);
    store   = [Result.x_k(times+1:end)' 0]';
    for t = 1:times,
        if produce(t) > 0 | store(t) > 0,
            disp(['during month ' num2str(t) ])
            if produce(t) > 0,
                disp(['  produce  ' num2str(produce(t))])
            end
            if store(t) > 0,
                disp(['  store   ' num2str(store(t)) ' to next month'])
            end
        end
    end
end
end
end

```

% MODIFICATION LOG

%

% 051018 med Created.

% 060110 per Added documentation.

% 060125 per Moved disp to end

## 22.6 Assignment of production batches to machines

```

% function Result = assignmentofprodbatchestomEx(PriLev)
%
% Creates a TOMLAB MIP problem for assignment of production batches to machines
%
% ASSIGNMENT OF PRODUCTION BATCHES TO MACHINES
%
% Having determined a set of ten batches to be produced in the next
% period, a production manager is looking for the best assignment of
% these batches to the different machines in his workshop. The five
% available machines in the workshop may each process any of these
% batches, but since they are all models from different years of
% manufacture the speed with which they process the batches changes
% from one machine to another. Furthermore, due to maintenance
% periods and readjustments, every machine may only work a certain
% number of hours in the planning period. The table below lists the
% processing times of the production batches on all machines and the
% capacities of the machines.
%
% Processing durations and capacities (in hours)
%
% +-----+-----+-----+-----+-----+
% |      |      |      |      |      |      |      |      |      |      |      |
% |      |      |      |      |      |      |      |      |      |      |      |
% |Machine| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|Capacity|
% +-----+-----+-----+-----+-----+
% |  1  | 8|15|14|23| 8|16| 8|25| 9|17|  18 |
% |  2  |15| 7|23|22|11|11|12|10|17|16|  19 |
% |  3  |21|20| 6|22|24|10|24| 9|21|14|  25 |
% |  4  |20|11| 8|14| 9| 5| 6|19|19| 7|  19 |
% |  5  | 8|13|13|13|10|20|25|16|16|17|  20 |
% +-----+-----+-----+-----+-----+
%
% The production cost of a batch depends on the machine that
% processes it. The hourly usage cost for every machine depends on
% its technology, its age, its consumption of consumables (such as
% electricity, machine oil) and the personnel required to operate it.
% These differences are amplified by the variation in the processing
% durations of a batch depending on the machine. The table below
% lists the production costs in thousand $. On which machine should
% each batch be executed if the production manager wishes to minimize
% the total cost of production?
%
% Production cost depending on the assignment (in k$)
%
% +-----+-----+-----+-----+-----+

```

```

% |           |   Production batches   |
% |           +-----+-----+-----+-----+
% |Machine| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|
% +-----+-----+-----+-----+
% |  1  |17|21|22|18|24|15|20|18|19|18|
% |  2  |23|16|21|16|17|16|19|25|18|21|
% |  3  |16|20|16|25|24|16|17|19|19|18|
% |  4  |19|19|22|22|20|16|19|17|21|19|
% |  5  |18|19|15|15|21|25|16|16|23|15|
% +-----+-----+-----+-----+
%
% VARIABLES
%
% batches           Hours needed to produce batches
% capacity          Hours available per machine
% costs             Cost to produce batches
%
% RESULTS
%
% For an interpretation of the results, use PriLev > 1, for example:
% Result = assignmentofprodbatchestomEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%
% OUTPUT PARAMETERS
% Result           Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 18, 2005.   Last modified Oct 18, 2005.

function Result = assignmentofprodbatchestomEx(PriLev)

if nargin < 1
    PriLev = 1;
end

batches      = [ 8 15 14 23  8 16  8 25  9 17;...
                15 7 23 22 11 11 12 10 17 16;...
                21 20 6 22 24 10 24  9 21 14;...
                20 11 8 14  9  5  6 19 19  7;...

```

```

    8 13 13 13 10 20 25 16 16 17];

capacity      = [18 ;19 ;25 ;19 ;20];

costs         = [17 21 22 18 24 15 20 18 19 18;...
    23 16 21 16 17 16 19 25 18 21;...
    16 20 16 25 24 16 17 19 19 18;...
    19 19 22 22 20 16 19 17 21 19;...
    18 19 15 15 21 25 16 16 23 15];

Prob = assignmentofprodbatchestom(batches, capacity, costs);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    [m,b] = size(batches);
    temp = reshape(Result.x_k,m,b);
    for i = 1:m,
        tempidx = find(temp(i,*)>0.5);
        disp([' machine ' num2str(i) ' deals with ' num2str(tempidx)])
    end
end

% MODIFICATION LOG
%
% 051018 med    Created.
% 060111 per    Added documentation.
% 060125 per    Moved disp to end

```

## 23 Loading and Cutting

### 23.1 Wagon load balancing

```
% function Result = wagonloadbalancingEx(PriLev)
%
% Creates a TOMLAB LP problem for wagon loading. Minimizes the max weight
%
% WAGON LOAD BALANCING
%
% Three railway wagons with a carrying capacity of 100 quintals (1
% quintal = 100 kg) have been reserved to transport sixteen boxes.
% The weight of the boxes in quintals is given in the following
% table. How shall the boxes be assigned to the wagons in order to
% keep to the limits on the maximum carrying capacity and to minimize
% the heaviest wagon load?
%
% Weight of boxes
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Box   | 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|12|13|14|15|16|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Weight|34| 6| 8|17|16| 5|13|21|25|31|14|13|33| 9|25|25|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% Before implementing a Mathematical Programming solution, one may
% wish to try to see whether it is possible to solve this problem
% instance with the following simple heuristic: until all boxes are
% distributed to the wagons we choose in turn the heaviest unassigned
% box and put it onto the wagon with the least load.
%
%
% VARIABLES
%
% minload           Minimal load accepted per wagon
% maxload           Maximal load accepted per wagon
% weights           Weight of each box
%
% RESULTS
%
% For an interpretation of the results, set PriLev > 1,
% Result = wagonloadbalancingEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
```

```

%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 7, 2005.  Last modified Feb 3, 2006.

function Result = wagonloadbalancingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

minload    = [ 0; 0; 0;];
maxload    = [100;100;100;];
weights    = [34;6;8;17;16;5;13;21;25;31;14;13;33;9;25;25];

Prob = wagonloadbalancing(minload, maxload, weights);
Prob.SolverInf = 'cplex';
Result = infLinSolve(Prob, PriLev);

if PriLev > 1,
    boxes = length(weights);
    wagons = length(minload);
    temp = reshape(Result.x_k,boxes,wagons);
    for w = 1:wagons,
        idx = find(temp(:,w) > 0.5);
        disp(['wagon ' num2str(w) ...
            ' has a total load of ' num2str(sum(weights(idx))) ...
            ' by carrying the boxes: ' num2str(idx') ])
    end
end

% MODIFICATION LOG
%
% 051007 med    Created.
% 060111 per    Added documentation.
% 060125 per    Moved disp to end
% 060203 med    Printing updated

```

## 23.2 Barge loading

```

% function Result = bargeloadingEx(PriLev)
%
% Creates a TOMLAB LP problem for barge loading, maximizing the profit
%
% BARGE LOADING
%
% A shipper on the river Rhine owns a barge of carrying capacity 1500
% m3. Over time he has specialized in the transport of wheat. He has
% seven regular customers who load and unload practically at the same
% places. The shipper knows his costs of transport from long
% experience and according to his personal preferences has concluded
% agreements with his clients for the price charged to them for the
% transport of their wheat. The following table summarizes the
% information about the seven clients. Every client wishes to
% transport a certain number of lots, deciding himself the size of
% his lots in m3. The table lists the price charged by the shipper
% for every transported lot. The last column of the table contains
% the cost incurred by the shipper per transported m3. This cost
% differs depending on the distance covered.
%
% Lots to transport
%
% +-----+-----+-----+-----+-----+
% |      |Available quantity|Lot size|Price per | Transport  |
% |Client| (no. of lots)    |(in m3) |lot (in $)|cost (in $/m3)|
% +-----+-----+-----+-----+-----+
% | 1  |      12          |   10  |  1000   |      80    |
% | 2  |      31          |    8  |   600   |      70    |
% | 3  |      20          |    6  |   600   |      85    |
% | 4  |      25          |    9  |   800   |      80    |
% | 5  |      50          |   15  |  1200   |      73    |
% | 6  |      40          |   10  |   800   |      70    |
% | 7  |      60          |   12  |  1100   |      80    |
% +-----+-----+-----+-----+-----+
%
% The objective of the shipper is to maximize his profit from
% transporting the wheat with lots that may be divided.
%
% Question 1:
% As a first step, assuming that every client has an unlimited
% quantity of wheat, which clients wheat should be transported?
%
% Question 2:
% If in addition the actual availability of wheat lots from the
% customers is taken into account, which strategy should the shipper

```

```

% adopt?
%
% Question 3:
% What happens if the lots cannot be divided?
%
% VARIABLES
%
% capacity          = 1500;
% units             = [12;31;20;25;50;40;60];
% sizes             = [10;8;6;9;15;10;12];
% unitprice         = [10;6;6;8;12;8;11]*100;
% cost              = [80;70;85;80;73;70;80];
% var1              30 plus
% var2              30 minus
%
% RESULTS
%
% x_k is a long vector of values, and if we run
% Result           = bargeloadingEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%
% OUTPUT PARAMETERS
% Result           Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 7, 2005.   Last modified Oct 7, 2005.

function Result = bargeloadingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

capacity    = 1500;
units       = [12;31;20;25;50;40;60];
sizes       = [10;8;6;9;15;10;12];
unitprice   = [10;6;6;8;12;8;11]*100;
cost        = [80;70;85;80;73;70;80];

```

```
Prob = bargeloading(capacity, units, sizes, unitprice, cost);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    disp('Best buy')
    for i = 1:length(units),
        if Result.x_k(i) > 0,
            disp([' ' num2str(Result.x_k(i)) ' unit(s) from ' num2str(i)])
        end
    end
end

% MODIFICATION LOG
%
% 051007 med    Created.
% 060111 per    Added documentation.
% 060125 per    Moved disp to end
```

### 23.3 Tank loading

```

% function Result = tankloadingEx(PriLev)
%
% Creates a TOMLAB MIP problem for tank loading, maximizing the left over
% capacity
%
% TANK LOADING
%
% Five tanker ships have arrived at a chemical factory. They are
% carrying loads of liquid products that must not be mixed: 1200
% tonnes of Benzol, 700 tonnes of Butanol, 1000 tonnes of Propanol,
% 450 tonnes of Styrene, and 1200 tonnes of THF. Nine tanks of
% different capacities are available on site. Some of them are
% already partially filled with a liquid. The following table lists
% the characteristics of the tanks (in tonnes). Into which tanks
% should the ships be unloaded (question 1) to maximize the capacity
% of the tanks that remain unused, or (question 2) to maximize the
% number of tanks that remain free?
%
% Characteristics of tanks
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Tank      | 1|  2 | 3| 4| 5| 6| 7| 8| 9|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Capacity  |500| 400|400|600|600|900|800|800|800|
% |Current product| -|Benzol| -| -| -| -|THF| -| -|
% |Quantity  | 0| 100 | 0| 0| 0| 0|300| 0| 0|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% VARIABLES
%
% capacity          Capacity of Empty tanks
% products          Amount of incoming chemicals
%
% RESULTS
%
% In order to interpret the results, try the following:
% Result = tankloadingEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level

```

```

%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 10, 2005.  Last modified Dec 8, 2005.

function Result = tankloadingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

capacity    = [500;0;400;600;600;900;0;800;800];
products    = [1200-300;700;1000;450;1200-500];

Prob = tankloading(capacity, products);
Result = tomRun('cplex', Prob, PriLev);
Result.remaining = sum(capacity)-Result.f_k;

if PriLev > 1,
    disp('NB: Tank 2 and tank 7 are filled before the optimization starts.')
    temp    = reshape(Result.x_k,9,5);
    ships   = length(products);
    for ship = 1:ships,
        tank = find(temp(:,ship));
        disp(['Ship number ' num2str(ship) ' unloads in tank(s) ' num2str(tank)])
    end
end

% MODIFICATION LOG
%
% 051010 med    Created
% 051208 med    Added remaining the return
% 060112 per    Added documentation.
% 060124 per    Interpretation of results upgraded.
% 060125 per    Moved disp to end

```

## 23.4 Backing up files

```
% function Result = backupfilesEx(PriLev)
%
% Creates a TOMLAB MIP problem for backing up files
%
% BACKING UP FILES
%
% Before leaving on holiday, you wish to backup your most important
% files onto floppy disks. You have got empty disks of 1.44Mb
% capacity. The sixteen files you would like to save have the
% following sizes: 46kb, 55kb, 62kb, 87kb, 108kb, 114kb, 137kb,
% 164kb, 253kb, 364kb, 372kb, 388kb, 406kb, 432kb, 461kb, and 851kb.
%
% Assuming that you do not have any program at hand to compress the
% files and that you have got a sufficient number of floppy disks to
% save everything, how should the files be distributed in order to
% minimize the number of floppy disks used?
%
% VARIABLES
%
% maxuse           A maximal number of disks
% capacity         Storage available on each disk
% sizes           Filesizes
%
% RESULTS
%
% Results are explained by running:
% Result          = backupfilesEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.2.0$
% Written Oct 10, 2005. Last modified Mar 14, 2006.

function Result = backupfilesEx(PriLev)
```

```

if nargin < 1
    PriLev = 1;
end

maxuse = 10;
capacity = 1440;
sizes = [46;55;62;87;108;114;137;164;253;364;372;388;406;432;461;851];

Prob = backupfiles(maxuse, capacity, sizes);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    files = length(sizes);
    filepos = reshape(Result.x_k(1:files*maxuse),files,maxuse);
    disks_in_use = sum(Result.x_k(files*maxuse+1:files*maxuse+maxuse));
    disp(['a total of ' num2str(disks_in_use) ' disks are in use'])
    for d = 1:maxuse,
        files_on_disk = filepos(:,d); % what files on disk d
        if (abs(sum(files_on_disk)) >= 0.5), % disk d is not empty?
            file_ids = find(files_on_disk);
            disp([' files on one of them: ' num2str(file_ids') ])
        end
    end
end

% MODIFICATION LOG
%
% 051010 med Created.
% 060112 per Added documentation
% 060125 per Moved disp to end
% 060314 med checking for empty disc changed

```

## 23.5 Cutting sheet metal

```
% function Result = cuttingsheetmetalEx(PriLev)
%
% Creates a TOMLAB MIP problem for cutting smaller parts from big ones.
%
% CUTTING SHEET METAL
%
% A sheet metal workshop cuts pieces of sheet metal from large
% rectangular sheets of 48 decimeters 96 decimeters (dm). It has
% received an order for 8 rectangular pieces of 36 dm 50 dm, 13
% sheets of 24 dm 36 dm, 5 sheets of 20 dm 60 dm, and 15 sheets
% of 18 dm 30 dm. These pieces of sheet metal need to be cut from
% the available large pieces. How can this order be satisfied by
% using the least number of large sheets?
%
% VARIABLES
%
% patterns          Each column represent a possible sheet.
% demand           Wanted rectangles
%
% RESULTS
%
% To explain the results, run:
% Result = cuttingsheetmetalEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 10, 2005. Last modified Oct 10, 2005.

function Result = cuttingsheetmetalEx(PriLev)

if nargin < 1
    PriLev = 1;
end
```

```

% Have to manually evaluate all the possible patterns.

patterns = [1 1 1 0 0 0 0 0 0 0 0 0 0 0 0;...
            2 1 0 2 1 0 3 2 1 0 5 4 3 2 1 0;...
            0 0 0 2 2 2 1 1 1 1 0 0 0 0 0 0;...
            0 1 3 0 1 3 0 2 3 5 0 1 3 5 6 8];
demand = [8;13;5;15];

Prob = cuttingsheetmetal(demand, patterns);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    x      = Result.x_k;
    idx    = find(x);
    order  = [x(idx) idx];
    for i = 1:length(idx),
        disp(['cut ' num2str([order(i,1)]) ' sheet(s) in pattern ' num2str([order(i,2)])])
    end
end

% MODIFICATION LOG
%
% 051010 med    Created.
% 060112 per    Added documentation.
% 060125 per    Moved disp to end

```

## 23.6 Cutting steel bars for desk legs

```
% function Result = cuttingsteelbarsfordesklegsEx(PriLev)
%
% Creates a TOMLAB MIP problem for cutting steel bars for desk legs.
%
% CUTTING STEEL BARS FOR DESK LEGS
%
% The company SchoolDesk produces desks of different sizes for
% kindergartens, primary and secondary schools, and colleges. The
% legs of the desks all have the same diameter, with different
% lengths: 40 cm for the smallest ones, 60 cm for medium height, and
% 70 cm for the largest ones. These legs are cut from steel bars of
% 1.5 or 2 meters. The company has received an order for 108 small,
% 125 medium and 100 large desks. How should this order be produced
% if the company wishes to minimize the trim loss?
%
% VARIABLES
%
% patterns          The different cutting patterns
% demand           Demand of the different lengths
% loss             Loss for each pattern
% lengths          The lengths
%
% RESULTS
%
% To interpret the results run this:
% Result = cuttingsteelbarsfordesklegsEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 10, 2005.  Last modified Dec 8, 2005.

function Result = cuttingsteelbarsfordesklegsEx(PriLev)

if nargin < 1
```

```

    PriLev = 1;
end

patterns = [0 0 2 0 2 3 0 0 1 3 0 5;...
            0 1 0 2 1 0 1 2 0 0 3 0;...
            2 1 1 0 0 0 2 1 2 1 0 0];

demand = [108;125;100]*4;
loss = [10 20 0 30 10 30 0 10 20 10 20 0]';
lengths = [150;200];

Prob = cuttingsteelbarsfordesklegs(demand, patterns, lengths);
Prob.fConstant = -75280; % Constant to deduct from objective
Result = tomRun('cplex', Prob, PriLev);
Result.loss = sum(Result.x_k.*loss);

if PriLev > 1,
    x = Result.x_k;
    idx = find(x);
    order = [x(idx) idx];
    disp(['a minimal loss of ' num2str(Result.loss) ' is found with this combination:'] )
    for i = 1:length(idx),
        disp([' cut ' num2str([order(i,1)]) ' bar(s) in pattern ' num2str([order(i,2)])])
    end
end

end

% MODIFICATION LOG
%
% 051010 med Created
% 051208 med Loss added to Result
% 060112 per Added documentation.
% 060112 per Minor update.
% 060125 per Moved disp to end

```

## 24 Ground transport

### 24.1 Car rental

```
% function Result = carrentalEx(PriLev)
%
% Creates a TOMLAB MIP problem for car rental
%
% CAR RENTAL
%
% A small car rental company has a fleet of 94 vehicles distributed
% among its 10 agencies. The location of every agency is given by its
% geographical coordinates X and Y in a grid based on kilometers. We
% assume that the road distance between agencies is approximately 1.3
% times the Euclidean distance (as the crow flies). The following
% table indicates the coordinates of all agencies, the number of cars
% required the next morning, and the stock of cars in the evening
% preceding this day.
%
% Description of the vehicle rental agencies
%
% +-----+-----+-----+-----+-----+-----+
% |Agency      | 1| 2| 3| 4| 5| 6| 7| 8| 9|10|
% +-----+-----+-----+-----+-----+-----+
% |X coordinate | 0|20|18|30|35|33| 5| 5|11| 2|
% |Y coordinate | 0|20|10|12| 0|25|27|10| 0|15|
% +-----+-----+-----+-----+-----+-----+
% |Required cars|10| 6| 8|11| 9| 7|15| 7| 9|12|
% |Cars present | 8|13| 4| 8|12| 2|14|11|15| 7|
% +-----+-----+-----+-----+-----+-----+
%
% Supposing the cost for transporting a car is $ 0.50 per km,
% determine the movements of cars that allow the company to
% re-establish the required numbers of cars at all agencies,
% minimizing the total cost incurred for transport.
%
% VARIABLES
%
% demand          Required cars per agency
% stock           Cars present
% cost            Cost per km to transport a car
% xcord           The X-coordinate of agencies
% ycord           The Y-coordinate of agencies
% n              Number of agencies
% distance        A matrix of distances
%
% RESULTS
```

```

%
% For an interpretation of the results, try this:
% Result          = carrentalEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 19, 2005.   Last modified Oct 19, 2005.

function Result = carrentalEx(PriLev)

if nargin < 1
    PriLev = 1;
end

demand          = [10  6  8  11  9  7  15  7  9  12]';
stock           = [ 8  13  4  8  12  2  14  11  15  7]';
cost            = 0.50;

xcord           = [ 0  20  18  30  35  33  5  5  11  2]';
ycord           = [ 0  20  10  12  0  25  27  10  0  15]';
n               = length(xcord);

distance        = zeros(n,n);

for i=1:n
    for j=1:n
        distance(i,j) = 1.3*sqrt( (xcord(i) - xcord(j))^2 + (ycord(i) - ycord(j))^2);
    end
end

Prob = carrental(cost, distance, stock, demand);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    idx_excess   = find(stock-demand > 0);
    idx_need     = find(stock-demand < 0);
end

```

```

n_excess      = length(idx_excess);
n_need       = length(idx_need);
temp         = reshape(Result.x_k,n_excess,n_need);

disp('THE SENDING OF CARS')
for i = 1:n_excess,      % scan all positions, disp interpretation
    disp(['agency ' num2str(idx_excess(i)) ' sends: ' ])
    for j = 1:n_need,
        if temp(i,j) ~= 0
            disp(['    ' num2str(temp(i,j)) ' car(s) to agency ' ...
                num2str(idx_need(j))])
        end
    end
    disp(' ')
end

disp('THE GETTING OF CARS')
for j = 1:n_need,
    disp(['agency ' num2str(idx_need(j)) ' gets: ' ])
    for i = 1:n_excess,      % scan all positions, disp interpretation
        if temp(i,j) ~= 0
            disp(['    ' num2str(temp(i,j)) ' car(s) from agency ' ...
                num2str(idx_excess(i))])
        end
    end
    disp(' ')
end
end
% MODIFICATION LOG
%
% 051019 med    Created.
% 060112 per    Added documentation.
% 060125 per    Moved disp to end

```



```

%      \          +---+----- 10 (rail) \          /
%      \          /          /          \          /
%      \          /          /          \          /
%      +--- 4 -----+          /          /
%      (D3) \          11 (road) /
%      +-----+
%
%
% VARIABLES
%
% arcs_out/in          For an arc i arcs_out(i) is its starting node
%                      and arcs_in(i) its target node
% arcs_min             Minimal load for an arc
% arcs_max             Maximal load for an arc
% arcs_cost            Cost for an arc
% minflow              Flow from 1 to 15
%
% RESULTS
%
% For an interpretation of the results, try the following
% Result = choosingthemodeoftransportEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev              Print Level
%
% OUTPUT PARAMETERS
% Result              Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 19, 2005. Last modified Oct 19, 2005.

function Result = choosingthemodeoftransportEx(PriLev)

if nargin < 1
    PriLev = 1;
end

arcs_out      = [ 1  1  1  1  2  2  3  3  4  4  4  5  5  5  5 ...
                 6  7  8  9 10 11 12 13 14 ]';
arcs_in       = [ 2  3  4  5  7  9  6  7  9 10 11  8  9 10 11 ...
                 12 12 13 13 14 14 15 15 15 ]';

```

```

arcs_min      = [zeros(1,6) 10 0 0 10 0 10 0 10 0 ...
                zeros(1,9)]';
arcs_max      = [50 30 35 65 inf inf 50 inf inf 50 inf 50 inf 50 inf ...
                inf*ones(1,9)]';
arcs_cost     = [0 0 0 0 12 11 12 14 9 4 5 11 14 10 14 ...
                zeros(1,9) ]';
minflow       = 50+30+35+65;

Prob = choosingthemodeoftransport(arcs_out, arcs_in, arcs_min, arcs_max, arcs_cost, minflow);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    arcs      = Result.x_k;

    disp('transport as follows:')

    for i = 1:length(arcs),
        if arcs(i) ~= 0,
            disp(['      ' num2str(arcs(i)) ' units from node ' ...
                num2str(arcs_out(i)) ' to node ' num2str(arcs_in(i))])
        end
    end

end

end

% MODIFICATION LOG
%
% 051019 med   Created.
% 060112 per   Added documentation.
% 060125 per   Moved disp to end

```

### 24.3 Depot location

```

% function Result = depotlocationEx(PriLev)
%
% Creates a TOMLAB MIP problem for depot location
%
% DEPOT LOCATION
%
% A large company wishes to open new depots to deliver to its sales
% centers. Every new set-up of a depot has a fixed cost. Goods are
% delivered from a depot to the sales centers close to the site.
% Every delivery has a cost that depends on the distance covered. The
% two sorts of cost are quite different: set-up costs are capital
% costs which may usually be written off over several years, and
% transport costs are operating costs. A detailed discussion of how
% to combine these two costs is beyond the scope of this text we
% assume here that they have been put on some comparable basis,
% perhaps by taking the costs over a year.
%
% There are 12 sites available for the construction of new depots and
% 12 sales centers need to receive deliveries from these depots.
%
% The following table gives the costs (in thousand $) of satisfying
% the entire demand of each customer (sales center) from a depot (not
% the unit costs). So, for instance, the cost per unit of supplying
% customer 9 (who has a total demand of 30 tonnes according to the
% next table) from depot 1 is $ 60000/30t, i.e. $ 2000/t. Certain
% deliveries that are impossible are marked with "Inf".
%
% Delivery costs for satisfying entire demand of customers
%
% +-----+-----+-----+-----+-----+-----+-----+-----+-----+
% |      |           Customer           |
% |Depot| 1  2  3  4  5  6  7  8  9 10 11 12|
% +-----+-----+-----+-----+-----+-----+-----+-----+-----+
% |  1  |100| 80| 50| 50| 60|100|120| 90| 60| 70| 65|110|
% |  2  |120| 90| 60| 70| 65|110|140|110| 80| 80| 75|130|
% |  3  |140|110| 80| 80| 75|130|160|125|100|100| 80|150|
% |  4  |160|125|100|100| 80|150|190|150|130|Inf|Inf|Inf|
% |  5  |190|150|130|Inf|Inf|Inf|200|180|150|Inf|Inf|Inf|
% |  6  |200|180|150|Inf|Inf|Inf|100| 80| 50| 50| 60|100|
% |  7  |100| 80| 50| 50| 60|100|120| 90| 60| 70| 65|110|
% |  8  |120| 90| 60| 70| 65|110|140|110| 80| 80| 75|130|
% |  9  |140|110| 80| 80| 75|130|160|125|100|100| 80|150|
% | 10  |160|125|100|100| 80|150|190|150|130|Inf|Inf|Inf|
% | 11  |190|150|130|Inf|Inf|Inf|200|180|150|Inf|Inf|Inf|
% | 12  |200|180|150|Inf|Inf|Inf|100| 80| 50| 50| 60|100|

```

```

% +-----+-----+-----+-----+-----+-----+-----+-----+-----+
%
% In addition, for every depot we have the following information: the
% fixed cost for constructing the depot that needs to be included
% into the objective function and its capacity limit, all listed in
% the table below.
%
% Fix costs and capacity limits of the depot locations
%
%+-----+-----+-----+-----+-----+-----+-----+-----+-----+
%|Depot      | 1| 2| 3| 4| 5| 6| 7| 8| 9| 10| 11| 12|
%+-----+-----+-----+-----+-----+-----+-----+-----+-----+
%|Cost (k$)  |3500|9000|10000|4000|3000|9000|9000|3000|4000|10000|9000|3500|
%|Capacity (t)| 300| 250| 100| 180| 275| 300| 200| 220| 270| 250| 230| 180|
%+-----+-----+-----+-----+-----+-----+-----+-----+-----+
%
% The quantities demanded by the sales centers (customers), are
% summarized in the following table.
%
% Demand data
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Customer | 1| 2| 3| 4| 5| 6| 7| 8| 9| 10|11| 12|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Demand (t)|120|80|75|100|110|100|90|60|30|150|95|120|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% In every case, the demand of a customer needs to be satisfied but a
% sales center may be delivered to from several depots. Which depots
% should be opened to minimize the total cost of construction and of
% delivery, whilst satisfying all demands?
%
% VARIABLES
%
% delcosts           Costs to deliver to centre from depot
% idxinf            Impossible combination of centre/depot
% delcosts(idxinf)  A large number < Inf
% buildcosts        Costs to build a depot=
% capacity          Capacities per potential depot
% demand            The customers demand in tonnes
%
% RESULTS
%
% To interpret the results run this:
% Result = depotlocationEx(2);
%
% REFERENCES
%

```

```

% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 19, 2005.   Last modified Oct 19, 2005.

function Result = depotlocationEx(PriLev)

if nargin < 1
    PriLev = 1;
end

delcosts    = [ 100  80  50  50  60 100 120  90  60  70  65 110;...
    120  90  60  70  65 110 140 110  80  80  75 130;...
    140 110  80  80  75 130 160 125 100 100  80 150;...
    160 125 100 100  80 150 190 150 130 inf inf inf;...
    190 150 130 inf inf inf 200 180 150 inf inf inf;...
    200 180 150 inf inf inf 100  80  50  50  60 100;...
    100  80  50  50  60 100 120  90  60  70  65 110;...
    120  90  60  70  65 110 140 110  80  80  75 130;...
    140 110  80  80  75 130 160 125 100 100  80 150;...
    160 125 100 100  80 150 190 150 130 inf inf inf;...
    190 150 130 inf inf inf 200 180 150 inf inf inf;...
    200 180 150 inf inf inf 100  80  50  50  60 100];

idxinf = find(delcosts == inf);
delcosts(idxinf) = 1e6;

buildcosts  = [3500 9000 10000 4000 3000 9000 9000 3000 4000 10000 9000 3500]';
capacity    = [ 300  250  100  180  275  300  200  220  270  250  230  180]';
demand      = [ 120  80  75  100  110  100  90  60  30  150  95  120]';

Prob = depotlocation(delcosts, buildcosts, capacity, demand, idxinf);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    c      = length(demand);           % number of customers
    d      = length(buildcosts);       % number of possible depots
    build  = find(Result.x_k(1:d)');    % the depots to build

```

```

deliver = reshape(Result.x_k(d+1:c*d+d),c,d);
deliver = deliver(:,build);
deliver(find(deliver<0.001)) = 0;
disp(['minimal total cost = ' num2str(Result.f_k) ])
disp(['build the depots      ' num2str(build)      ])
[jj,ii] = size(deliver);           % ii = depots, jj = customers
for j = 1:jj,
    disp(['let customer ' num2str(j) ' get '])
    for i = 1:ii,
        if deliver(j,i) ~= 0,
            if deliver(j,i) == 1,
                disp([' all of its demand from depot ' num2str(build(i)) ])
            else
                disp([' ' num2str(deliver(j,i)) ...
                    ' of its demand from depot ' num2str(build(i)) ])
            end
        end
    end
end
end
end
end

% MODIFICATION LOG
%
% 051019 med Created.
% 060112 per Added documentation.
% 060125 per Moved disp to end

```

## 24.4 Heating oil delivery

```

% function Result = heatingoildeliveryEx(PriLev)
%
% Creates a TOMLAB MIP problem for heating oil delivery
%
% HEATING OIL DELIVERY
%
% A transporter has to deliver heating oil from the refinery at
% Donges to a certain number of clients in the west of France. His
% clients are located at Brain-sur-lAuthion, Craquefou, Gurande, la
% Haie Fouassire, Msanger and les Ponts-de-C. The following table
% lists the demands in liters for the different sites.
%
% Demands by clients (in liters)
%
% +-----+-----+-----+-----+-----+
% | BslA| Craq| Gur| H Fo| Msa| P dC|
% +-----+-----+-----+-----+-----+
% |14000| 3000| 6000|16000|15000| 5000|
% +-----+-----+-----+-----+-----+
%
% The next table contains the distance matrix between the clients and
% the refinery.
%
% Distance matrix (in km)
% +-----+-----+-----+-----+-----+-----+
% |          |Dong|BslA|Craq|Gur|H Fo|Msa|P dC|
% +-----+-----+-----+-----+-----+-----+
% |Donges      |  0| 148|  55|  32|  70| 140|  73|
% |Brain-s.-lAuthion | 148|  0|  93| 180|  99|  12|  72|
% |Craquefou   |  55|  93|  0|  85|  20|  83|  28|
% |Gurande     |  32|  80|  85|  0| 100| 174|  99|
% |Haie Fouassire |  70|  99|  20| 100|  0|  85|  49|
% |Msanger     | 140|  12|  83| 174|  85|  0|  73|
% |Ponts-de-C  |  73|  72|  28|  99|  49|  73|  0|
% +-----+-----+-----+-----+-----+-----+
%
% The transport company uses tankers with a capacity of 39000 liters
% for the deliveries. Determine the tours for delivering to all
% clients that minimize the total number of kilometers driven.
%
% VARIABLES
%
% distance          The distance matrix
% demand           Demand
% capacity          Capacity of tanker

```

```

%
% RESULTS
%
% Run this for an interpretation of the results
% Result = heatingoildeliveryEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 20, 2005.  Last modified Oct 20, 2005.

function Result = heatingoildeliveryEx(PriLev)

if nargin < 1
    PriLev = 1;
end

distance    = [ 0 148 55 32 70 140 73;...
               148 0 93 180 99 12 72;...
               55 93 0 85 20 83 28;...
               32 180 85 0 100 174 99;...
               70 99 20 100 0 85 49;...
               140 12 83 174 85 0 73;...
               73 72 28 99 49 73 0];

demand      = [14000;3000;6000;16000;15000;5000];
capacity    = 39000;

Prob = heatingoildelivery(distance, demand, capacity);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    s      = 7;                                % number of sites
    names  = ['Dong'; 'Bsla'; 'Craq'; 'Gur'; 'H Fo'; 'Msa'; 'P dC'];
    tour   = reshape(Result.x_k(1:s*s),s,s);    % extract tour
    tour(find(tour<0.5)) = 0;                  % remove false zeros
end

```

```

first = find(tour(1,:)); % might be an array
disp(['min distance of ' num2str(Result.f_k) ' km by using'])

for init = 1:length(first), % there might be many first stops
    this_tour = [names(1,:)]; % start in Dong
    site = first(init);
    this_tour = [this_tour ' -> ' names(site,:)]; % add city 2
    loop_me = 1;
    next = find(tour(site,:)); % what city is next?

    if next == 1, % if we are going home:
        loop_me = 0; % do not enter while-loop
        this_tour = [this_tour ' -> ' names(1,:)]; % just add home and quit
        disp(['Tour ' num2str(init) ': ' num2str(this_tour) ])
    end

    while loop_me == 1, % if more than one stop
        this_tour = [this_tour ' -> ' names(next,:)]; % add them one at a time
        next = find(tour(next,:));

        if next == 1, % when we are going home
            loop_me = 0; % stop loop
            this_tour = [this_tour ' -> ' names(1,:)]; % finish names and quit
            disp([' tour ' num2str(init) ': ' num2str(this_tour) ])
        end
    end
end
end
end

% MODIFICATION LOG
%
% 051020 med Created.
% 060112 per Added documentation.
% 060125 per Moved disp to end

```

## 24.5 Combining different modes of transport

```

% function Result = combiningdiffmodesoftranspEx(PriLev)
%
% Creates a TOMLAB MIP problem for combining different modes of transport
%
% COMBINING DIFFERENT MODES OF TRANSPORT
%
% A load of 20 tonnes needs to be transported on a route passing
% through five cities, with a choice of three different modes of
% transport: rail, road, and air. In any of the three intermediate
% cities it is possible to change the mode of transport but the load
% uses a single mode of transport between two consecutive cities.
% The following table lists the cost of transport in $ per tonne
% between the pairs of cities.
%
% Transport costs with different modes
%
%
% Pairs of cities
% +-----+-----+-----+-----+
% |      |12|23|34|45|
% +-----+-----+-----+-----+
% |Rail| 30| 25| 40| 60|
% |Road| 25| 40| 45| 50|
% |Air | 40| 20| 50| 45|
% +-----+-----+-----+-----+
%
% The next table summarizes the costs for changing the mode of
% transport in $ per tonne. The cost is independent of location.
%
%
% Cost for changing the mode of transport
%
% +-----+-----+-----+-----+
% |from \ to|Rail|Road|Air|
% +-----+-----+-----+-----+
% |Rail    | 0 | 5 | 12|
% |Road    | 8 | 0 | 10|
% |Air     | 15| 10| 0 |
% +-----+-----+-----+-----+
%
% How should we organize the transport of the load at the least cost?
%
% VARIABLES
%
% transpcost          Transport costs

```

```

% changecost          Cost to change mode of transport
% demand             Load to transport
%
% RESULTS
%
% For an interpretation of the results, try this:
% Result = combiningdiffmodesoftranspEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev             Print Level
%
% OUTPUT PARAMETERS
% Result             Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 20, 2005. Last modified Oct 20, 2005.

```

```

function Result = combiningdiffmodesoftranspEx(PriLev)

if nargin < 1
    PriLev = 1;
end

transpcost = [ 30 25 40 60 ;...
              25 40 45 50 ;...
              40 20 50 45];

changecost = [ 0 5 12;...
              8 0 10;...
              15 10 0];

demand = 30;

Prob = combiningdiffmodesoftransp(transpcost, changecost, demand);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    modes = reshape(Result.x_k(1:3*4),3,4);
    means = ['rail';'road';'air '];
    disp(['the min cost (' num2str(Result.f_k) ') is found by,'])

```

```
for m = 1:size(modes,2),
    disp([' going from town ' num2str(m) ' to town ' num2str(m+1) ...
        ' by ' means(find(modes(:,m)),:)]))
end
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051020 med Created.
```

```
% 060112 per Added documentation.
```

```
% 060125 per Moved disp to end
```

## 24.6 Fleet planning for vans

```

% function Result = fleetplanningforvansEx(PriLev)
%
% Creates a TOMLAB MIP problem for fleet planning for vans
%
% FLEET PLANNING FOR VANS
%
% A chain of department stores uses a fleet of vans rented from
% different rental agencies. For the next six months period it has
% forecast the following needs for vans (table below):
%
% Requirements for vans for six months
%
% +---+---+---+---+---+---+
% |Jan|Feb|Mar|Apr|May|Jun|
% +---+---+---+---+---+---+
% |430|410|440|390|425|450|
% +---+---+---+---+---+---+
%
% At the 1st January, the chain has 200 vans, for which the rental
% period terminates at the end of February.
%
% To satisfy its needs, the chain has a choice among three types of
% contracts that may start the first day of every month: 3-months
% contracts for a total cost of $1700 per van, 4-months contracts at
% $2200 per van, and 5-months contracts at $2600 per van. How many
% contracts of the different types need to be started every month in
% order to satisfy the companys needs at the least cost and to have
% no remaining vans rented after the end of June?
%
% Running contracts in month 5 (May) .....
%
%                               +-----:-----:-----+
%                               | Rent34:      :          |
%                               +-----+-----:-----+
%                               |   Rent33      :          :
%                               +-----:-----:-----+
%                               |   Rent43      :          :          |
%                               +-----+-----:-----+
%                               |   Rent42      :          :          |
%                               +-----:-----:-----+
%                               |   Rent52      :          :          |
%                               +-----+-----:-----+
%                               |   Rent51      :          :          |
%                               +-----:-----:
%                               .           .           .           :           :           .

```

```

% Month   Jan : Feb : Mar : Apr : May : Jun :
%         :.....:.....:.....:.....:      :.....:
%
%
%
% VARIABLES
%
% demand           Demand of vans per month
% initialsupply    Vans per month
% contractlength   Contracts available
% contractcost     Cost of contracts
%
% RESULTS
%
% To interpret the result, use PriLev > 1, for example:
% Result = fleetplanningforvansEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Oct 21, 2005.   Last modified Oct 21, 2005.

function Result = fleetplanningforvansEx(PriLev)

if nargin < 1
    PriLev = 1;
end

demand          = [ 430; 410; 440; 390; 425; 450];

initialsupply   = [ 200; 200;  0;  0;  0;  0];

contractlength  = [ 3; 4; 5];

contractcost    = [ 1700; 2200; 2600];

Prob = fleetplanningforvans(demand, initialsupply, contractlength, contractcost);

```

```

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    months = length(demand);
    c      = length(contractlength);
    temp   = reshape(Result.x_k,c,months);
    disp(['minimal cost of ' num2str(Result.f_k) ' by '])
    for m = 1:months,
        if sum(temp(:,m)) > 0,
            rent = find(temp(:,m));
            disp(['renting ' num2str(temp(rent,m)) ' vans for ' ...
                num2str(contractlength(rent)) ' months in month ' num2str(m)])
        end
    end
end

end

% MODIFICATION LOG
%
% 051021 med Created.
% 060112 per Added documentation.
% 060125 per Moved disp to end

```

## 25 Air transport

### 25.1 Flight connections at a hub

```
% function Result = flightconnectionsatahubEx(PriLev)
%
% Creates a TOMLAB MIP problem for flight connections at a hub
%
% FLIGHT CONNECTIONS AT A HUB
%
% The airline SafeFlight uses the airport Roissy-Charles-de-Gaulle
% as a hub to minimize the number of flight connections to European
% destinations. Six Fokker 100 airplanes of this airline from
% Bordeaux, Clermont-Ferrand, Marseille, Nantes, Nice, and Toulouse
% are landing between 11am and 12:30pm. These aircraft leave for
% Berlin, Bern, Brussels, London, Rome, and Vienna between 12:30 pm
% and 13:30 pm. The numbers of passengers transferring from the
% incoming flights to one of the outgoing flights are listed in the
% table below.
%
% Numbers of passengers transferring between the different flights
% +-----+-----+-----+-----+-----+-----+
% |Origins      |           Destinations           |
% |            +-----+-----+-----+-----+-----+
% |            |Berlin|Bern|Brussels|London|Rome|Vienna|
% +-----+-----+-----+-----+-----+-----+
% |Bordeaux     | 35 | 12 | 16  | 38 | 5 | 2  |
% |Clermont-Ferrand| 25 | 8  | 9   | 24 | 6 | 8  |
% |Marseille    | 12 | 8  | 11  | 27 | 3 | 2  |
% |Nantes       | 38 | 15 | 14  | 30 | 2 | 9  |
% |Nice         |    | 9  | 8   | 25 | 10| 5  |
% |Toulouse     |    |    |     | 14 | 6 | 7  |
% +-----+-----+-----+-----+-----+-----+
%
% For example, if the flight incoming from Bordeaux continues on to
% Berlin, 35 passengers and their luggage may stay on board during
% the stop at Paris. The flight from Nice arrives too late to be
% re-used on the connection to Berlin, the same is true for the
% flight from Toulouse that cannot be used for the destinations
% Berlin, Bern and Brussels (the corresponding entries in the table
% are marked with ).
%
% How should the arriving planes be re-used for the departing
% flights to minimize the number of passengers who have to change
% planes at Roissy?
%
% VARIABLES
```

```

%
% origdest           People remaining in plane at transfer
% idximp            Impossible combinations
%
% RESULTS
%
% for an interpretation of the results, set PriLev > 1, for example:
% flightconnectionsatahubEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%
% OUTPUT PARAMETERS
% Result           Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 21, 2005. Last modified Feb 3, 2006.

function Result = flightconnectionsatahubEx(PriLev)

if nargin < 1
    PriLev = 1;
end

origdest      = [ 35 12 16 38  5  2;...
                25  8  9 24  6  8;...
                12  8 11 27  3  2;...
                38 15 14 30  2  9;...
                0  9  8 25 10  5;...
                0  0  0 14  6  7];

idximp = find(origdest == 0);

Prob = flightconnectionsatahub(origdest, idximp);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1
    p           = 6; % the number of planes
    in_names    = ['Bord'; 'Cler'; 'Mars'; ... % city names in
                  'Nant'; 'Nice'; 'Toul'];

```

```

out_names = ['Berl';'Bern';'Brus';... % city names out
            'Lond';'Rome';'Vien'];
transfers = reshape(Result.x_k,p,p); % reshape x_k

for i = 1:p, % in city has number i
    for o = 1:p, % out city has number o
        if transfers(i,o) == 1,
            disp(['plane from ' in_names(i,:) ...
                ' continues to ' out_names(o,:) ])
        end
    end
end
end
end

% MODIFICATION LOG
%
% 051021 med Created.
% 060113 per Added documentation.
% 060125 per Moved disp to end
% 060203 med Print level updated

```

## 25.2 Composing flight crews

```

% function Result = composingflightcrewsEx(PriLev)
%
% Creates a TOMLAB MIP problem for composing flight crews
%
% COMPOSING FLIGHT CREWS
%
% During the Second World War the Royal Air Force (RAF) had many
% foreign pilots speaking different languages and more or less
% trained on the different types of aircraft. The RAF had to form
% pilot/co-pilot pairs (crews) for every plane with a compatible
% language and a sufficiently good knowledge of the aircraft type. In
% our example there are eight pilots. In the following table every
% pilot is characterized by a mark between 0 (worst) and 20 (best)
% for his language skills (in English, French, Dutch, and Norwegian),
% and for his experience with different two-seater aircraft
% (reconnaissance, transport, bomber, fighterbomber, and supply
% planes).
%
% Ratings of pilots
%
% +-----+-----+-----+-----+-----+-----+-----+
% |          |Pilot          | 1| 2| 3| 4| 5| 6| 7| 8|
% +-----+-----+-----+-----+-----+-----+-----+
% |Language |English          |20|14| 0|13| 0| 0| 8| 8|
% |          |French           |12| 0| 0|10|15|20| 8| 9|
% |          |Dutch            | 0|20|12| 0| 8|11|14|12|
% |          |Norwegian        | 0| 0| 0| 0|17| 0| 0|16|
% +-----+-----+-----+-----+-----+-----+-----+
% |Plane type|Reconnaissance   |18|12|15| 0| 0| 0| 8| 0|
% |          |Transport        |10| 0| 9|14|15| 8|12|13|
% |          |Bomber           | 0|17| 0|11|13|10| 0| 0|
% |          |Fighter-bomber  | 0| 0|14| 0| 0|12|16| 0|
% |          |Supply plane     | 0| 0| 0| 0|12|18| 0|18|
% +-----+-----+-----+-----+-----+-----+-----+
%
% A valid flight crew consists of two pilots that both have each at
% least 10/20 for the same language and 10/20 on the same aircraft
% type.
%
% Question 1:
% Is it possible to have all pilots fly?
%
% Subsequently, we calculate for every valid flight crew the sum of
% their scores for every aircraft type for which both pilots are
% rated at least 10/20. This allows us to define for every crew the

```

```

% maximum score among these marks. For example, pilots 5 and 6 have
% marks 13 and 10 on bombers and 12 and 18 on supply planes. The
% score for this crew is therefore max(13 + 10, 12 + 18) = 30.
%
% Question 2:
% Which is the set of crews with maximum total score?
%
% Compatibility graph for pilots
% Nodes = pilots
% Arcs = possible combinations (with scores)
%
% 1 ----30--- 2 ----27--- 3
%
% | \      / |      / |
% | 24  28 |      / |
% | \    / 27    26 |
% |      |      / |
% |      4 |      / 30
% 25      |      / |
% | /    \ |      / |
% | 29   21 | /    |
% | /     \ | /    |
%
% 5 ---30----- 6 -----28--- 7
%
% \      |      /
% \      |      /
% \      |      /
% \      |      /
% \      |      /
% 30     |      25
% \      |      /
% \      |      /
%
%      8
%
% VARIABLES
%
% scores                Scores for language and piloting skills
% arcs_out/in          For an arc i arcs_out(i) is its starting node
%                      and arcs_in(i) is its target node
% arcs_score           Strength of an arc
%
% RESULTS
%

```

```

% Set PriLev to 2 for an interpretation of the results
% Result      = composingflightcrewsEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 21, 2005.   Last modified Feb 3, 2006.

function Result = composingflightcrewsEx(PriLev)

if nargin < 1
    PriLev = 1;
end

scores      = [ 20 14  0 13  0  0  8  8;...
               12  0  0 10 15 20  8  9;...
               0 20 12  0  8 11 14 12;...
               0  0  0  0 17  0  0 16;...
               18 12 15  0  0  0  8  0;...
               10  0  9 14 15  8 12 13;...
               0 17  0 11 13 10  0  0;...
               0  0 14  0  0 12 16  0;...
               0  0  0  0 12 18  0 18];

arcs_in     = [1 1 1 2 2 2 3 3 4 4 5 5 6 6 7]';
arcs_out    = [5 4 2 4 6 3 6 7 5 6 6 8 8 7 8]';

arcs_score  = [25 24 30 28 27 27 26 30 29 21 30 30 ...
               36 28 25]';

Prob = composingflightcrews(scores, arcs_in, arcs_out, arcs_score);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1
    idx      = find(Result.x_k);
    for crew = 1:idx,

```

```
        id = idx(crew);
        disp(['crew ' num2str(crew) ' has pilots ' ...
            num2str(arcs_in(id)) ' and ' num2str(arcs_out(id))])
    end
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051021 med    Created.
```

```
% 060113 per    Added documentation.
```

```
% 060125 per    Moved disp to end
```

```
% 060203 med    Updated print level
```

### 25.3 Scheduling flight landings

```

% function Result = schedulingflightlandingsEx(PriLev)
%
% Creates a TOMLAB MIP problem for scheduling flight landings
%
% SCHEDULING FLIGHT LANDINGS
%
% The plane movements in large airports are subject to numerous
% security constraints. The problem presented in this section consist
% of calculating a schedule for flight landings on a single runway.
% More general problems have been studied but they are fairly
% complicated (dynamic cases, for instance through planes arriving
% late, instances with several runways, etc.).
%
% Ten planes are due to arrive. Every plane has an earliest arrival
% time (time when the plane arrives above the zone if traveling at
% maximum speed) and a latest arrival time (influenced among other
% things by its fuel supplies). Within this time window the airlines
% choose a target time, communicated to the public as the flight
% arrival time. The early or late arrival of an aircraft with respect
% to its target time leads to disruption of the airport and causes
% costs. To take into account these cost and to compare them more
% easily, a penalty per minute of early arrival and a second penalty
% per minute of late arrival are associated with every plane. The
% time windows (in minutes from the start of the day) and the
% penalties per plane are given in the following table.
%
% Characteristics of flight time windows
% +-----+-----+-----+-----+-----+-----+-----+-----+-----+
% |Plane          | 1| 2| 3| 4| 5| 6| 7| 8| 9|10|
% +-----+-----+-----+-----+-----+-----+-----+-----+-----+
% |Earliest arrival|129|195| 89| 96|110|120|124|126|135|160|
% |Target time     |155|258| 98|106|123|135|138|140|150|180|
% |Latest Arrival  |559|744|510|521|555|576|577|573|591|657|
% |Earliness penalty| 10| 10| 30| 30| 30| 30| 30| 30| 30| 30|
% |Lateness penalty| 10| 10| 30| 30| 30| 30| 30| 30| 30| 30|
% +-----+-----+-----+-----+-----+-----+-----+-----+-----+
%
% Due to turbulence and the duration of the time during which a plane
% is on the runway, a security interval has to separate any two
% landings. An entry in line p of column q in the following table
% denotes the minimum time interval (in minutes) that has to lie
% between the landings of planes p and q, even if they are not
% consecutive. Which landing schedule minimizes the total penalty
% subject to arrivals within the given time windows and the required
% intervals separating any two landings?

```

```

%
% Matrix of minimum intervals separating landings
%
% +---+---+---+---+---+---+---+---+---+---+
% |  | 1| 2| 3| 4| 5| 6| 7| 8| 9|10|
% +---+---+---+---+---+---+---+---+---+---+
% | 1|  | 3|15|15|15|15|15|15|15|15|15|
% | 2| 3|  |15|15|15|15|15|15|15|15|15|
% | 3|15|15|  | 8| 8| 8| 8| 8| 8| 8|
% | 4|15|15| 8|  | 8| 8| 8| 8| 8| 8|
% | 5|15|15| 8| 8|  | 8| 8| 8| 8| 8|
% | 6|15|15| 8| 8| 8|  | 8| 8| 8| 8|
% | 7|15|15| 8| 8| 8| 8|  | 8| 8| 8|
% | 8|15|15| 8| 8| 8| 8| 8|  | 8| 8|
% | 9|15|15| 8| 8| 8| 8| 8| 8|  | 8|
% |10|15|15| 8| 8| 8| 8| 8| 8| 8|  |
% +---+---+---+---+---+---+---+---+---+---+
%
% VARIABLES
%
% costs                Cost or being late
% mintimes             Minimal intervals between landings
% var1                 30 plus
% var2                 30 minus
%
% RESULTS
%
% For an interpretation of the results, set PriLev > 1, for example:
% Result = schedulingflightlandingsEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev              Print Level
%
% OUTPUT PARAMETERS
% Result              Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Oct 21, 2005.   Last modified Feb 3, 2006.

function Result = schedulingflightlandingsEx(PriLev)

```

```

if nargin < 1
    PriLev = 1;
end

costs      = [ 129 195  89  96 110 120 124 126 135 160;...
              155 258  98 106 123 135 138 140 150 180;...
              559 744 510 521 555 576 577 573 591 657;...
              10  10 30*ones(1,8);...
              10  10 30*ones(1,8)];

mintimes   = [  0  3 15 15 15 15 15 15 15 15;...
              3  0 15 15 15 15 15 15 15 15;...
              15 15  0  8  8  8  8  8  8  8;...
              15 15  8  0  8  8  8  8  8  8;...
              15 15  8  8  0  8  8  8  8  8;...
              15 15  8  8  8  0  8  8  8  8;...
              15 15  8  8  8  8  0  8  8  8;...
              15 15  8  8  8  8  8  0  8  8;...
              15 15  8  8  8  8  8  8  0  8;...
              15 15  8  8  8  8  8  8  8  0];

Prob = schedulingflightlandings(costs, mintimes);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1
    planes      = 10; % number of planes
    [times,idx] = sort(Result.x_k(1:planes));
    late        = Result.x_k(planes*planes+2*planes+1:planes*planes+3*planes);
    early       = Result.x_k(planes*planes+1*planes+1:planes*planes+2*planes);
    for p = 1:planes,
        time = times(p);
        plane = idx(p);
        if (late(plane) > 0.5),
            disp(['plane ' num2str(plane) ' will arrive minute ' ...
                 num2str(time) ' ( ' num2str(late(plane)) ' minute(s) late)'])
        elseif (early(plane) > 0.5),
            disp(['plane ' num2str(plane) ' will arrive minute ' ...
                 num2str(time) ' ( ' num2str(early(plane)) ' minute(s) early)'])
        else
            disp(['plane ' num2str(plane) ' will arrive minute ' ...
                 num2str(time) ' (on time)' ])
        end
    end
end
end

% MODIFICATION LOG
%
```

% 051021 med Created.  
% 060113 per Added documentation  
% 060125 per Moved disp to end  
% 060203 med Print level updated

## 25.4 Airline hub location

```

% function Result = airlinehublocationEx(PriLev)
%
% Creates a TOMLAB MIP problem for airline hub location
%
% AIRLINE HUB LOCATION
%
% FAL (the Flying Air Lines) specializes in freight transport. The
% company links the major French cities with cities in the United
% States, namely: Atlanta, Boston, Chicago, Marseille, Nice, and
% Paris. The average quantities in tonnes transported every day by
% this company between these cities are given in the following table.
%
% Average quantity of freight transported between every pair of cities
%
% +-----+-----+-----+-----+-----+-----+-----+
% |           |Atlanta|Boston|Chicago|Marseille|Nice|Paris|
% |Atlanta   |    0  | 500  | 1000  |   300   |400  |1500  |
% |Boston    |1500  |    0  |   250  |   630   |360  |1140  |
% |Chicago   | 400  | 510  |    0   |   460   |320  | 490  |
% |Marseille | 300  | 600  | 810   |    0    |820  | 310  |
% |Nice      | 400  | 100  | 420   |   730   |  0   | 970  |
% |Paris     | 350  |1020  | 260   |   580   |380  |  0   |
% +-----+-----+-----+-----+-----+-----+-----+
%
% We shall assume that the transport cost between two cities i and j
% is proportional to the distance that separates them. The distances
% in miles are given in the next table.
%
% Distances between pairs of cities
%
% +-----+-----+-----+-----+-----+-----+
% |           |Boston|Chicago|Marseille|Nice|Paris|
% +-----+-----+-----+-----+-----+-----+
% |Atlanta   |  945 |  605  | 4667   |4749| 4394|
% |Boston    |  866 |3726   | 3806   |3448|    |
% |Chicago   |4471 |4541   | 4152   |    |    |
% |Marseille | 109  | 415   |    |    |    |
% |Nice      | 431  |    |    |    |    |
% +-----+-----+-----+-----+-----+-----+
%
% The airline is planning to use two cities as connection platforms
% (hubs) to reduce the transport costs. Every city is then assigned
% to a single hub. The traffic between cities assigned to a given
% hub H1 to the cities assigned to the other hub H2 is all routed
% through the single connection from H1 to H2 which allows the

```

```

% airline to reduce the transport cost. We consider that the
% transport cost between the two hubs decreases by 20%. Determine the
% two cities to be chosen as hubs in order to minimize the transport
% cost.
%
% VARIABLES
%
% frights          Goods between cities
% distance         Distances
% hubs             Number of hubs
%
% RESULTS
%
% The result from this example is quite hard to interpret since
% Result.x_k should be split into a tensor of rank 4 with the indices
% i, j, k and l and also into a vector of length c (number of
% cities). The vector indicates the hubs. For all non-zero element
% (i,j,k,l) in the tensor the following interpretation is possible:
% the route from city i to j passes through the hubs k and l.
%
% In the example below temp(3,5,2,6) == 1 since the route from
% 3 (Chicago) to 5 (Nice) pass through 2 (Boston) and 6 (Paris).
% Also temp(4,5,6,6) == 1 since the route from 4 (Marseille) to
% 5 (Nice) pass through only 6 (Paris).
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2006 by Tomlab Optimization Inc., $Release: 5.1.0$
% Written Nov 7, 2005.   Last modified Feb 3, 2006.

function Result = airlinehublocationEx(PriLev)

if nargin < 1
    PriLev = 1;
end

frights      = [ 0  500  1000  300  400  1500;...

```

```

1500    0    250    630    360    1140;...
400    510    0    460    320    490;...
300    600    810    0    820    310;...
400    100    420    730    0    970;...
350    1020    260    580    380    0];

distance    = [    0    945    605    4667    4749    4394;...
945    0    866    3726    3806    3448;...
605    866    0    4471    4541    4152;...
4667    3726    4471    0    109    415;...
4749    3806    4541    109    0    431;...
4394    3448    4152    415    431    0];

hubs        = 2;

Prob = airlinehublocation(frights, distance, hubs);

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1
    c        = 6;    % number of cities
    c_names = ['Atla';'Bost';'Chic';'Mars';'Nice';'Pari'];
    temp     = reshape(Result.x_k(1:c^4),c,c,c,c);
    hubs     = find(Result.x_k(c^4+1:c^4+c));

    disp('Put hubs in ')
    for h = 1: length(hubs),
        disp([' ' num2str(c_names(hubs(h),:)) ])
    end

    % displaying the routes between cities
    for i = 1:c,
        for j = i+1:c,
            for k = 1:c,
                for l = 1:c,
                    if temp(i,j,k,l) == 1,
                        % route involving one hub
                        if k==l & i~=k & j~=k & i~=j,
                            disp(['the route ' num2str([c_names(i,:), ' ', ...
                                c_names(j,:)]) ' requires hub ' num2str(c_names(k,:))])
                        % route involving two hubs
                        elseif i~=j & i~=k & i~=l & j~=k & j~=l & k~=l,
                            disp(['the route ' num2str([c_names(i,:), ' ', ...
                                c_names(j,:)]) ' requires hubs ' ...
                                num2str([c_names(k,:), ' ', c_names(l,:)])])
                        % route between hubs
                        elseif ( (i==k & j==l) | (i==l & j==k) ) & k~=l,

```

```
                disp(['the route ' num2str([c_names(i,:), ' ', ...
                c_names(j,:)]) ' is between hubs' ])
            end
        end
    end
end
end
end
end
end
```

```
% MODIFICATION LOG
```

```
%
% 051107 med    Created.
% 060113 per    Added documentation.
% 060116 per    Minor changes.
% 060125 per    Moved disp to end
% 060203 med    Updated print level
```

## 25.5 Planning a flight tour

```
% function Result = planningaflighttourEx(PriLev)
%
% Creates a TOMLAB MIP problem for planning a flight tour
%
% PLANNING A FLIGHT TOUR
%
% A country in south-east Asia is experiencing widespread flooding.
% The government, with international help, decides to establish a
% system of supply by air. Unfortunately, only seven runways are
% still in a usable state, among which is the one in the capital.
%
% The government decides to make the planes leave from the capital,
% have them visit all the other six airports and then come back to
% the capital. The following table lists the distances between the
% airports. Airport A1 is the one in the capital. In which order
% should the airports be visited to minimize the total distance
% covered?
%
% Distance matrix between airports (in km)
%
% +---+---+---+---+---+---+---+
% | | A2| A3| A4| A5| A6| A7|
% +---+---+---+---+---+---+---+
% |A1|786|549|657|331|559|250|
% |A2|668|979|593|224|905|  |
% |A3|316|607|472|467|  |  |
% |A4|890|769|400|  |  |  |
% |A5|386|559|  |  |  |  |
% |A6|681|  |  |  |  |  |
% +---+---+---+---+---+---+---+
%
% VARIABLES
%
% distance          The distance matrix
%
% RESULTS
%
% For an interpretation of the results set PriLev > 1, for example:
% Result = planningaflighttourEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
```

```

% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Nov 7, 2005.  Last modified Nov 7, 2005.

function Result = planningaflighttourEx(PriLev)

if nargin < 1
    PriLev = 1;
end

distance = [ 0 786 549 657 331 559 250;...
            786 0 668 979 593 224 905;...
            549 668 0 316 607 472 467;...
            657 979 316 0 890 769 400;...
            331 593 607 890 0 386 559;...
            559 224 472 769 386 0 681;...
            250 905 467 400 559 681 0];

Prob = planningaflighttour(distance);

stop = 0;
n = size(distance,1);
while stop<100
    Result = tomRun('cplex', Prob, PriLev);
    % Find a subtour
    idx = zeros(n,n);
    idx(1,1:n) = [1:7];
    for i=1:n
        for j=2:n
            idx(j,i) = find(Result.x_k( (idx(j-1,i)-1)*n+1: idx(j-1,i)*n) ~= 0);
            if idx(j,i) == i
                break
            end
        end
    end
    shortmat = spones(idx);
    len = full(sum(shortmat,1));
    [val, idx2] = min(len);
    idx3 = idx(1:val, idx2);

    Aextra = zeros(1,Prob.N);

```

```

for k=1:val-1
    Aextra(1,idx3(k)+n*(idx3(k+1)-1)) = 1;
end

Prob.A = [Prob.A;Aextra];
Prob.b_L = [Prob.b_L;-inf];
Prob.b_U = [Prob.b_U;1];
Prob.mLin = Prob.mLin+1;
Result.tour = idx(:,1);
if val == n
    break
end
stop = stop + 1;
end

if PriLev > 1,
    a                = 7;                % number of airports
    temp             = reshape(Result.x_k,a,a); % reshape results
    not_home_again = true;                % help to loop
    current_pos      = 1;                % we start in 1
    route            = [];                % initially empty
    route            = [route current_pos]; % add position
    while not_home_again,                % loop if not home
        current_pos = find(temp(current_pos,:)); % go to next pos
        route = [route current_pos];      % add to route
        if current_pos == 1,              % if home:
            not_home_again = false;       % stop loop
        end
    end
    end
    disp(['Shortest route is: ' num2str(route)])
    disp(['                   or: ' num2str(route(end:-1:1))])

end

% MODIFICATION LOG
%
% 051107 med    Created.
% 060116 per    Added documentation.
% 060125 per    Moved disp to end

```

## 26 Telecommunication

### 26.1 Network reliability

```
% function Result = networkreliabilityEx(PriLev)
%
% Creates a TOMLAB MIP problem for network reliability
%
% NETWORK RELIABILITY
%
% We consider the military telecommunications network represented
% below. This network consists of eleven sites connected by
% bidirectional lines for data transmission. For reliability reasons
% in the case of a conflict, the specifications require that the two
% sites (nodes) 10 and 11 of the network remain able to communicate
% even if any three other sites of the network are destroyed. Does
% the network satisfy this requirement?
%
%
%          1 --- 2 ----- 8
%
%      / | / |           / |
%      / | / |           / |
%      / | / |           / |
%          |               |
% 11 --- 3 ---+----- 10   |
%          |               |
% | \ / \ | / | \ | |
% | X   \ | / | | |
% | / \   \ | / | | |
%          | | | |
% 4 --- 5 --- 9   | | |
%          | | | |
%      \   | /   | |
%      \   | /   \ |
%      \   | /   \ |
%          \
%          ----- 6 ----- 7
%
%
%
% VARIABLES
%
% arcs_out/in           For an arc i arcs_out(i) is its starting node
%                       and arcs_in(i) ts target node
%
% RESULTS
```

```

%
% for an interpretation of the results, use a PriLev > 1, for example
% Result = networkreliabilityEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Nov 7, 2005.   Last modified Nov 7, 2005.

function Result = networkreliabilityEx(PriLev)

if nargin < 1
    PriLev = 1;
end

arcs_in   = [1 1  1 2 2 2 3 3  3  3 4 4  4 5  5 6 6  6 7  7 8  9]';
arcs_out  = [2 3 11 3 8 9 4 9 10 11 5 6 11 9 11 7 9 10 8 10 10 10]';

Prob = networkreliability(arcs_in, arcs_out);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    f      = - Result.f_k;
    x      = - Result.x_k;
    disp(['The network can manage the loss of ' num2str(f-1) ' nodes.'])
    disp(['There are ' num2str(f) ' disjunctive paths.'])
    [arcs,direction] = find(reshape(x,length(arcs_in),2));
    disp('For a maximal number of disjunctive paths, activate...')
    for arc = 1:length(arcs),
        if direction(arc) == 1,
            disp([' arc ' num2str([arcs_out(arcs(arc))]) '->' ...
                num2str([arcs_in(arcs(arc))])])
        end
        if direction(arc) == 2,
            disp([' arc ' num2str([arcs_in(arcs(arc))]) '->' ...
                num2str([arcs_out(arcs(arc))])])
        end
    end
end

```

```
end  
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051107 med Created.
```

```
% 060116 per Added documentation.
```

```
% 060126 per Moved disp to end
```

## 26.2 Dimensioning of a mobile phone network

```
% function Result = dimofamobilephonenetworkEx(PriLev)
%
% Creates a TOMLAB MIP problem for dimensioning of a mobile phone network
%
% DIMENSIONING OF A MOBILE PHONE NETWORK
%
% The figure below represents the typical architecture of a mobile
% phone network. Every elementary geographical zone or cell is served
% by a transmitter-receiver called a relay. The calls originating
% from a mobile phone first pass through these relays. Every relay is
% connected by cable or electro-magnetic waves to a transit node
% (hub). One of the hubs controls the network, this is the MTSO
% (Mobile Telephone Switching Office). A very reliable ring of fiber
% optic cable connects the hubs and the MTSO with high capacity
% links. It is capable of re-establishing itself in the case of a
% breakdown (self-healing ring) and there is no need to replicate it.
%
% At the present state of technology, there are no dynamic
% connections between the relays and the MTSO. The connections are
% fixed during the design phase, so it is necessary to choose the
% nodes of the ring that a relay should be connected to. The number
% of links between a cell  $c$  and the ring is called the diversity of
% the cell  $c$ , denoted by  $CNCTc$ . A diversity larger than 1 is
% recommended for making the system more reliable.
%
% The traffic in this kind of system is entirely digitized, expressed
% in values equivalent to bidirectional circuits at 64kbps (kilobit
% per second). This capacity corresponds to the number of
% simultaneous calls during peak periods. The ring has edges of known
% capacity  $CAP$ . The traffic  $TRAFc$  from a cell  $c$  is divided into equal
% parts ( $TRAFc / CNCTc$ ) among the connections to the ring. This
% traffic is transmitted via the ring to the MTSO, that then routes
% it to another cell or to a hub that serves as the interface to the
% fixed-line telephone network. A relay may be connected directly to
% the MTSO that also has the functions of an ordinary hub.
%
% We consider a network of 10 cells and a ring of 5 nodes with a
% capacity of  $CAP = 48$  circuits. The MTSO is at node 5. The following
% table indicates the traffic, the required number of connections and
% the cost per connection in thousand $ per cell. For example, cell 1
% is connectable with node 1 for a cost of $15,000. Its diversity is
% 2, which means it must be connected to at least two nodes of the
% ring. Its traffic capacity is of 22 simultaneous circuits. The
% objective is to define the connections of the cells to the ring
% that minimize the connection costs whilst remaining within the
```

```

% capacity limits and satisfying the constraints on the number of
% connections.
%
% Structure of a mobile phone network
%
%   Cell 1
%   2 connections
%
%   Relay -----\
%   |              \
%   |              \
%   |              |
%   V              V
%
%   hub2 ===== hub3 <-- Relay
%   cell 2
%   1 connection
%   ||             ||
%   ||             ||
%   ||             ||
%
%   hub1 === MTSO === hub4
%
% Connection costs, traffic and number of connections per cell
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Cell      | 1| 2| 3| 4| 5| 6| 7| 8| 9|10|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Hub 1      |15| 9|12|17| 8| 7|19|20|21|25|
% |Hub 2      | 8|11| 6| 5|22|25|25| 9|22|24|
% |Hub 3      | 7| 8| 7| 9|21|15|21|15|14|13|
% |Hub 4      |11| 5|15|18|19| 9|20|18|16| 4|
% |Hub 5 (MTSO)|10|14|15|24| 6|17|22|25|20|11|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Traffic    |22|12|20|12|15|25|15|14| 8|22|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Connections| 2| 2| 2| 2| 3| 1| 3| 2| 2| 2|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% VARIABLES
%
% hub_mat      Matrix describing the hubs
% traffic      Traffic from cells
% connections   Possible connections per cell
% capacity     Capacity
%
% RESULTS

```

```

%
% For an interpretation of the results, try:
% Result = dimofamobilephonenetworkEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Nov 8, 2005.   Last modified Nov 8, 2005.

function Result = dimofamobilephonenetworkEx(PriLev)

if nargin < 1
    PriLev = 1;
end

hub_mat      = [15  9 12 17  8  7 19 20 21 25;...
                8 11  6  5 22 25 25  9 22 24;...
                7  8  7  9 21 15 21 15 14 13;...
                11  5 15 18 19  9 20 18 16  4;...
                10 14 15 24  6 17 22 25 20 11]*1000;

traffic      = [22 12 20 12 15 25 15 14  8 22]';
connections  = [ 2  2  2  2  3  1  3  2  2  2]';
capacity     = 48;

Prob = dimofamobilephonenetwork(hub_mat, traffic, connections,...
    capacity);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    [h,c] = size(hub_mat) ;
    temp = reshape(Result.x_k,c,h);
    for cell = 1:c,
        idx = find(temp(cell,:));
        disp(['cell ' num2str(cell) ' connects to hub(s) ' num2str(idx)])
    end
end
end

```

% MODIFICATION LOG

%

% 051108 med Created.

% 060116 per Added documentation.

% 060126 per Moved disp to end

## 26.3 Routing telephone calls

```
% function Result = routingtelephonedcallsEx(PriLev)
%
% Creates a TOMLAB MIP problem for routing telephone calls
%
% ROUTING TELEPHONE CALLS
%
% A private telephone company exploits a network, represented in the
% figure below, between five cities: Paris, Nantes, Nice, Troyes, and
% Valenciennes.
%
% Structure of the network of the company
%
% Nantes --(300)-- Paris --(200)-- Valenciennes
%
%      \          /          |
%      (120)    (300)        (70)
%      \          /          |
%
%           Nice  -----(80)---- Troyes
%
% The number beside each edge (connection) is the capacity of the
% link in terms of circuits. This issue is worth some further
% explanation. Suppose that a person A at Nantes calls a person B at
% Nice. The company needs to find a path formed by non-saturated
% edges from Nantes to Nice to route the binary flow corresponding to
% the digitized voice of A. But the conversation is only possible if
% a flow from Nice to Nantes is established so that B may answer A.
% In digital telephone networks, the binary flows all have the same
% throughput standard (often 64 kbps). The associated capacity is
% called a channel. The return channel uses the same edges as the
% first channel, but in the opposite direction. This linked pair of
% channels necessary for a telephone conversation is a circuit.
%
% The routing of the return channel through the same edges is due to
% the fact that the call could fail if one waited until the callee
% picked up the phone before searching for a non-saturated return
% path. This is why, at the moment of the call, the routing system
% constructs the path edge by edge and immediately reserves the edges
% of the return channel. As a consequence, as the capacity of an edge
% is consumed in increments of a bidirectional circuit, we do not
% consider any directed flows. For example, there may be 10 persons
% calling from Nantes to Nice and 20 from Nice to Nantes, or the
% opposite: in both cases, 30 circuits are used.
%
% At a given moment, the company is facing demands for circuits given
```

```

% in the following table. Is it possible to satisfy the demands
% entirely? If this is not possible, try to transmit as much as
% possible. In every case, indicate the corresponding routing,
% that is, the access paths used.
%
% Demand of circuits
%
% +-----+-----+
% |Cities          |Circuits|
% +-----+-----+
% |Nantes-Nice     | 100  |
% |Nantes-Troyes   |  80  |
% |Nantes-Valenciennes|  75  |
% |Nice-Valenciennes | 100  |
% |Paris-Troyes    |  70  |
% +-----+-----+
%
%
% VARIABLES
%
% arcs_out/in          For an arc i arcs_out(i) is its starting node
%                      and arcs_in(i) ts target node
% capacity             Capacity of arcs
% demand_out/in       For a demand i demand_out(i) is its starting node
%                      and demand_in(i) its target node
% demands             Demand
% path_mat            Possible paths
% paths              In/Out of possible paths
%
% RESULTS
%
% For an interpretation of the results, try:
% Result = routingtelephonenumberEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$

```

```
% Written Nov 22, 2005. Last modified Nov 22, 2005.
```

```
function Result = routingtelephonecallsEx(PriLev)
```

```
if nargin < 1  
    PriLev = 1;  
end
```

```
% Nantes, Paris, Nice, Valenciennes, Troyes
```

```
arcs_in    = [1 1 2 2 3 4]';  
arcs_out   = [2 3 3 4 5 5]';  
capacity   = [300 120 300 200 80 70]';
```

```
demand_in  = [1 1 1 3 2]';  
demand_out = [3 5 4 4 5]';  
demands    = [100 80 75 100 70]';
```

```
path_mat    = [1 3 0 0 0;...  
               1 2 3 0 0;...  
               1 2 4 5 3;...  
               1 2 4 5 0;...  
               1 2 3 5 0;...  
               1 3 5 0 0;...  
               1 3 2 4 5;...  
               1 2 4 0 0;...  
               1 3 2 4 0;...  
               1 2 3 5 4;...  
               1 3 5 4 0;...  
               3 1 2 4 0;...  
               3 2 4 0 0;...  
               3 5 4 0 0;...  
               2 4 5 0 0;...  
               2 1 3 5 0;...  
               2 3 5 0 0];
```

```
paths       = [1 3;...  
               1 3;...  
               1 5;...  
               1 5;...  
               1 5;...  
               1 5;...  
               1 4;...  
               1 4;...  
               1 4;...  
               1 4;...]
```

```

3 4;...
3 4;...
3 4;...
2 5;...
2 5;...
2 5];

```

```

Prob = routingtelephonenumbercalls(arcs_in, arcs_out, capacity,...
    demand_in, demand_out, demands, path_mat, paths);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    paths = Result.x_k;
    idx = find(paths);
    load = [];
    cities = ['Nant'; 'Pari'; 'Nice'; 'Vale'; 'Troy'];
    disp('INCOMING AND OUTGOING CIRCUITS')
    for i = 1:length(idx),
        id = idx(i);
        temp_path = path_mat(id,:);
        temp_path = temp_path(find(temp_path));
        city_path = [];
        for city = 1:length(temp_path),
            city = temp_path(city);
            city_path = [city_path cities(city,:) ' '];
        end
        disp(['    ' num2str(paths(id)) ' circuits ' ...
            cities(temp_path(1),:) '-' cities(temp_path(end),:) ...
            ' using the following path: ' num2str(city_path)])
        for j = 2:length(temp_path),
            out = temp_path(j-1);
            in = temp_path(j);
            if in < out,
                in_temp = in;
                in = out;
                out = in_temp;
            end
            if (size(load) > [out in]),
                load(out,in) = load(out,in) + paths(id);
            else
                load(out,in) = paths(id);
            end
        end
    end
end
disp('LOAD BETWEEN CITIES')
for i = 1:length(cities)-1,
    for j = 1:length(cities),

```

```
        if load(i,j) > 0,
            disp(['  between ' cities(i,:) ' and ' cities(j,:) ': ' ...
                num2str(load(i,j)) ' circuits.' ])
        end
    end
end
end
end
```

```
% MODIFICATION LOG
```

```
%
% 051122 med   Created.
% 060116 per   Added documentation.
% 060126 per   Moved disp to end
```

## 26.4 Construction of a cabled network

```
% function Result = constructionofacablednetworkEx(PriLev)
%
% Creates a TOMLAB MIP problem for construction of a cabled network
%
% CONSTRUCTION OF A CABLED NETWORK
%
% A university wishes to connect six terminals located in different
% buildings of its campus. The distances, in meters, between the
% different terminals are given in the table below.
%
% Distances between the different terminals (in meters)
%
% +-----+---+---+---+---+---+
% |           | T1| T2| T3| T4| T5| T6|
% +-----+---+---+---+---+---+
% |Terminal 1| 0|120| 92|265|149|194|
% |Terminal 2|120| 0|141|170| 93|164|
% |Terminal 3| 92|141| 0|218|103|116|
% |Terminal 4|265|170|218| 0|110|126|
% |Terminal 5|149| 93|103|110| 0| 72|
% |Terminal 6|194|164|116|126| 72| 0|
% +-----+---+---+---+---+---+
%
% These terminals are to be connected via underground cables. We
% suppose the cost of connecting two terminals is proportional to the
% distance between them. Determine the connections to install to
% minimize the total cost.
%
% VARIABLES
%
% distances                a distance matrix
%
% RESULTS
%
% for an interpretation of the results, let PriLev > 1, for example:
% Result = constructionofacablednetworkEx(2); % call solver
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
```

```

% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Nov 22, 2005.   Last modified Nov 22, 2005.

function Result = constructionofacablednetworkEx(PriLev)

if nargin < 1
    PriLev = 1;
end

distances    = [ 0 120  92 265 149 194;...
                120  0 141 170  93 164;...
                92 141  0 218 103 116;...
                265 170 218  0 110 126;...
                149  93 103 110  0  72;...
                194 164 116 126  72  0];

Prob = constructionofacablednetwork(distances);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    t      = size(distances,1);           % number of terminals
    s      = sum(1:t-1);                 % possible connections
    arcs   = [];                         % empty set of arcs
    count  = 1;                          % counter
    for i = 1:t-1,                       % we catch true arcs
        for j = i+1:t,
            if Result.x_k(count) == 1,
                arcs = [arcs ; i j];
            end
            count = count + 1;
        end
    end
    for arc = 1:length(arcs),
        arc = arcs(arc,:);
        disp(['connect the terminals ' num2str(arc(1)) ' and ' num2str(arc(2)) ])
    end
end

% MODIFICATION LOG
%
% 051122 med   Created.
% 060116 per   Added documentation.
% 060126 per   Moved disp to end

```

## 26.5 Scheduling of telecommunications via satellite

```

% function Result = schedulingofteleviasatelliteEx(PriLev)
%
% Creates a TOMLAB MIP problem for scheduling of telecommunications via
% satellite
%
% SCHEDULING OF TELECOMMUNICATIONS VIA SATELLITE
%
% A digital telecommunications system via satellite consists of a
% satellite and a set of stations on earth which serve as interfaces
% with the terrestrial network. With SS-TDMA (Satellite-Switch, Time
% Division Multiple Access) access technology, the satellite divides
% its time among the stations. Consider for example the transmissions
% from four transmitter stations in the US to four receiver stations
% in Europe. The following table gives a possible 4 4 traffic
% matrix. TRAFtr is the quantity of data transmitted from station t
% to station r. It is expressed in seconds of transmission duration,
% because all lines have the same constant transmission rate.
%
% Matrix TRAF with its lower bound LB
%
% +-----+-----+-----+-----+
% |TRAF| 1| 2| 3| 4| rowt |
% +-----+-----+-----+-----+
% | 1 | 0| 7|11|15| 33 |
% | 2 |15| 8|13| 9| 45 |
% | 3 |17|12| 6|10| 45 |
% | 4 | 6|13|15| 4| 38 |
% +-----+-----+-----+-----+
% |colr|38|40|45|38|LB = 45|
% +-----+-----+-----+-----+
%
% The satellite has a switch that allows any permutation between the
% four transmitters and the four receivers. The figure below gives an
% example of a permutation connecting the transmitters 1 to 4 to the
% receivers 3, 4, 1, and 2 respectively. These connections allow
% routing a part of the traffic matrix, called a mode. A part of a
% matrix entry transmitted during a mode is a packet of data. A mode
% is thus a 4 4 matrix M with at most one non-zero packet per row
% and column and such that Mtr <= TRAFtr for all t, r. To every
% mode corresponds a slice of a schedule as shown in the figure.
%
% +-----+-----+-----+-----+           +-----+-----+-----+-----+
% |TRAF| 1| 2| 3| 4|           |Stations | Packets           |
% +-----+-----+-----+-----+           +-----+-----+-----+-----+
% | 1 | 0| 0|11| 0|           | 1 --> 3 |-----11---->   |

```

```

% | 2 | 0| 0| 0| 9|          | 2 --> 4 |-----9->      |
% | 3 |15| 0| 0| 0|          | 3 --> 1 |-----15----->|
% | 4 | 0|13| 0| 0|          | 4 --> 2 |-----13-----> |
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |colr|38|40|45|38|LB = 45
% +-----+-----+-----+-----+
%
%
% Example of a mode and associated schedule
%
% A valid schedule of transmissions defines a sequence of
% permutations for the on-board switch that routes all the traffic of
% the matrix TRAF. In other words, this boils down to decomposing
% TRAF into a sum of mode matrices. An element of TRAF may be
% fragmented, like TRAF31 that is only partially transmitted in the
% mode represented in the figure above. A fragmented element will
% appear in several packets and several modes of the final
% decomposition. The duration of a mode is the length of its longest
% packet. The objective is to find a schedule with minimal total duration.
%
% VARIABLES
%
% traffic          A matrix describing the traffic
%
% RESULTS
%
% For an interpretation of the results, let PriLev > 1,
% schedulingofteleviasatelliteEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Nov 22, 2005.   Last modified Nov 22, 2005.

function Result = schedulingofteleviasatelliteEx(PriLev)

if nargin < 1

```

```

    PriLev = 1;
end

if PriLev > 1, % if: PriLev > 1
    sequence = []; % then: let us catch all x_k
end

traffic      = [ 0  7 11 15;...
    15  8 13  9;...
    17 12  6 10;...
    6 13 15  4];

TQBS = schedulingofteleviasatellite(traffic);

while sum(sum(TQBS)) > 0.01
    n1 = size(TQBS,1);
    n2 = n1^2;
    n = n2 + 1;
    x_L = zeros(n,1);
    x_U = [ones(n2,1);inf];
    IntVars = [ones(n2,1);0];
    b_L1 = ones(n1,1);
    b_U1 = ones(n1,1);
    b_L2 = ones(n1,1);
    b_U2 = ones(n1,1);
    b_L3 = zeros(n1,1);
    b_U3 = inf*ones(n1,1);
    A1 = zeros(n1,n);
    A2 = zeros(n1,n);
    A3 = zeros(n1,n);
    for i=1:n1
        idx1 = [(i-1)*n1+1:i*n1];
        idx2 = find(TQBS(:,i) == 0);
        idx1(idx2) = [];
        idx3 = [i:n1:n2-n1+i];
        idx4 = find(TQBS(i,:) == 0);
        idx3(idx4) = [];
        A1(i,idx1) = ones(1,length(idx1));
        A2(i,idx3) = ones(1,length(idx3));
        A3(i,(i-1)*n1+1:i*n1) = TQBS(:,i)';
        A3(i,end) = -1;
    end
    c = [zeros(n2,1);-1];
    Prob = mipAssign(c, [A1;A2;A3], [b_L1;b_L2;b_L3], [b_U1;b_U2;b_U3], x_L, x_U, [], 'Satellite Scheduling',
        [], [], IntVars);
    Result = tomRun('cplex', Prob, 0);
    x_k = Result.x_k;
end

```

```

x_k = reshape(x_k(1:end-1), n1, n1);
TQBS = TQBS - x_k*(-Result.f_k);

if PriLev > 1,
    sequence = [sequence Result.x_k];
end

end

PrintResult(Result,PriLev);

if PriLev > 1,
    for t = 1:size(sequence,2),
        disp(['Transmission ' num2str(t) ' transfers ' num2str(sequence(end,t)) ' packet(s) of data' ])
        this = reshape(sequence(1:end-1,t),n1,n1);
        this(find(this < 0.5)) = 0; % remove bad zeros
        for j = 1:n1,
            disp([' from station ' num2str(j) ' to ' num2str(find(this(j,:))) ])
        end
    end
end

end

% MODIFICATION LOG
%
% 051122 med Created.
% 060116 per Added documentation.
% 060126 per Moved disp to end
% 060131 per cleaned up help text

```

## 26.6 Location of GSM transmitters

```

% function Result = locationofgsmtransmittersEx(PriLev)
%
% Creates a TOMLAB MIP problem for location of gsm transmitters
%
% LOCATION OF GSM TRANSMITTERS
%
% A mobile phone operator decides to equip a currently uncovered
% geographical zone. The management allocates a budget of $ 10
% million to equip this region. A study shows that only 7 locations
% are possible for the construction of the transmitters and it is
% also known that every transmitter only covers a certain number of
% communities. The figure below represents a schematic map of the
% region with the division into communities and the possible
% locations for transmitters. Every potential site is indicated by
% four stars and a number, every community is represented by a
% polygon. The number in the center of a polygon is the number of the
% community.
%
% Certain geographical and topological constraints add to the
% construction cost and reduce the reach of the GSM transmitters. The
% table below lists the communities covered and the cost for every
% site.
%
% For every community the number of inhabitants is known (see table).
% Where should the transmitters be built to cover the largest
% population with the given budget limit of $ 10M?
%
% Map of the region to cover
%
% +-----+-----+-----+-----+
% |          |          * 7  /          |
% |  1      |  4  *3*      /          |
% |          *      * /--\  /*      11 |
% | +-----*1*      | / 10 \ /*6*      |
% |          * \      / \      /  *---+-----+
% |          \      / \  \--/      \      |
% |          \ / \      |          \  15 |
% |  2      /      \ 8  |*          \      |
% |          /      \ *5*  12  * \      +
% |          /      \ |*          *7* \ / |
% |          /  5      \ |          *  \ / |
% | +-----*--+      *  \ |-----+----- / |
% |          *2* \      -*4*---+      |          \ 14 |
% |          *  \ /      * /          |          \ |
% |          \ /      /  9  |          \ |

```

```

% | 3      \ 6 /      | 13  \ |
% |      \ /      |      +
% +-----+-----+
%
% Inhabitants of the communities
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Community      | 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|12|13|14|15|
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Population (in 1000)| 2| 4|13| 6| 9| 4| 8|12|10|11| 6|14| 9| 3| 6|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% Cost and communities covered for every site
%
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Site            | 1 | 2 | 3  | 4  | 5  | 6  | 7  |
% +-----+-----+-----+-----+-----+-----+-----+-----+
% |Cost (in million $)| 1.8 | 1.3 | 4.0  | 3.5  | 3.8  | 2.6  | 2.1  |
% |Communities covered|1,2,4|2,3,5|4,7,8,10|5,6,8,9|8,9,12|7,10,11,12,15|12,13,14,15|
% +-----+-----+-----+-----+-----+-----+-----+-----+
%
% VARIABLES
%
% budget            Money available
% cost              Cost to build site
% connections       Communities covered by a site
% population        People living in communities
% var1              30 plus
% var2              30 minus
%
% RESULTS
%
% To interpret the results try the following:
% Result = locationofgsmtransmittersEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%
% OUTPUT PARAMETERS
% Result           Result structure.
%
% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz

```

```

% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Nov 30, 2005. Last modified Nov 30, 2005.

```

```

function Result = locationofgsmttransmittersEx(PriLev)

```

```

if nargin < 1
    PriLev = 1;
end

```

```

budget      = 10e6;
cost        = [1.8 1.3 4.0 3.5 3.8 2.6 2.1]'*1e6;

```

```

connections = [ 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0;...
    0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0;...
    0 0 0 1 0 0 1 1 0 1 0 0 0 0 0 0;...
    0 0 0 0 1 1 0 1 1 0 0 0 0 0 0 0;...
    0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0;...
    0 0 0 0 0 0 1 0 0 1 1 1 0 0 1;...
    0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1];

```

```

population = [ 2 4 13 6 9 4 8 12 10 11 6 ...
    14 9 3 6]'*1000;

```

```

Prob = locationofgsmttransmitters(budget, cost, connections, ...
    population);
Result = tomRun('cplex', Prob, PriLev);

```

```

if PriLev > 1,
    [loc,com] = size(connections);
    build     = Result.x_k(1:loc);
    cover     = Result.x_k(loc+1:end);
    disp(['Build at locations ' num2str(find(build))])
    disp(['Cover communities ' num2str(find(cover))])
end

```

```

% MODIFICATION LOG
%
% 051130 med Created.
% 060116 per Added documentation.
% 060126 per Moved disp to end

```

## 27 Economics

### 27.1 Choice of loans

```
% function Result = choiceofloansEx(PriLev)
%
% Creates a TOMLAB MIP problem for choice of loans
%
% CHOICE OF LOANS
%
% Mr Chic, director of a chain of shops selling clothes, wishes to
% open three new shops: one in London, one in Munich, and one in
% Rome. To open a new shop costs respectively $ 2.5 million, $ 1
% million and $ 1.7 million. To finance his project, he calls at
% three different banks.
%
% Rates offered by the banks for the different projects
%
% +-----+-----+-----+-----+
% |          |London|Munich|Rome|
% +-----+-----+-----+-----+
% |Bank 1| 5.0% | 6.5% |6.1%|
% |Bank 2| 5.2% | 6.2% |6.2%|
% |Bank 3| 5.5% | 5.8% |6.5%|
% +-----+-----+-----+-----+
%
% Depending on the location of the shops and the evaluation of the
% related risks, each bank decides to finance at most $ 3 million
% over 8 years with different interest rates for the shops. Determine
% the amount to borrow from each bank for financing each shop in
% order to minimize the total expenses of Mr Chic.
%
% VARIABLES
%
% rates                The rates
% costs                Cost per site
% maxloan              Max loan per bank
% loanlength           Length of the loan in years
%
% RESULTS
%
% for an interpretation of the results, let PriLev > 1:
% Result = choiceofloansEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
```

```

% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Nov 30, 2005.  Last modified Nov 30, 2005.

function Result = choiceofloansEx(PriLev)

if nargin < 1
    PriLev = 1;
end

rates      = [ 5.0 6.5 6.1 ;...
              5.2 6.2 6.2 ;...
              5.5 5.8 6.5 ];

costs      = [2.5 1.0 1.7]'*1e6;
maxloan    = 3e6;
loanlength = 8;

Prob = choiceofloans(rates, costs, maxloan, loanlength);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    [b,s] = size(rates); % banks/sites
    temp  = Result.x_k;
    temp  = reshape(temp,b,s)';
    for i = 1:s,
        site = temp(:,i);
        idx  = find(site);
        disp(['To finance shop at site ' num2str(i)])
        for j = 1:length(idx),
            disp([' take a loan of ' num2str(site(idx(j))) ...
                ' in bank ' num2str(idx(j))])
        end
    end
end

% MODIFICATION LOG
%
% 051130 med    Created.

```

% 060116 per Added documentation.

## 27.2 Publicity campaign

```

% function Result = publicitycampaignEx(PriLev)
%
% Creates a TOMLAB MIP problem for publicity campaign
%
% PUBLICITY CAMPAIGN
%
% The small company, Pronuevo, launches a new product into a regional
% market and wishes to have a publicity campaign using different
% media. It therefore contacts a regional publicity agency, PRCo,
% that specializes in this type of regional campaign and completely
% hands over this task for a total budget of $ 250,000. The agency
% knows the market well, that is, the impact of publicity in a local
% magazine or over the radio, or as a TV spot on the regional
% channel. It suggests addressing the market for two months through
% six different media. For each medium, it knows the cost and the
% number of people on which this medium has an impact. An index of
% the quality of perception of the campaign is also known for every
% medium.
%
% The publicity agency establishes a maximum number of uses of every
% medium (for instance, not more than eight transmissions of a TV
% spot). The table below lists all this information. Pronuevo wants
% the impact of the publicity campaign to reach at least 100,000
% people. Which media should be chosen and in which proportions to
% obtain a maximum index of perception quality?
%
% Data for the publicity campaign
% +-----+-----+-----+-----+-----+-----+
% |      |      | People      | Unit |Maximum      |Perception|
% |Number|Media type      |potentially reached| cost | use          | quality  |
% +-----+-----+-----+-----+-----+-----+
% |  1  |Free weekly newspaper| 12,000          | 1,500| 4 weeks      | 3        |
% |  2  |Monthly magazine    | 1,500           | 8,000| 2 months     | 7        |
% |  3  |Weekly magazine     | 2,000           |12,000| 8 weeks      | 8        |
% |  4  |Radio spot          | 6,000           | 9,000|60 broadcasts| 2        |
% |  5  |Billboard 4x3 m     | 3,000           |24,000| 4 boards     | 6        |
% |  6  |TV spot             | 9,000           |51,000| 8 broadcasts| 9        |
% +-----+-----+-----+-----+-----+-----+
%
% VARIABLES
%
% budget          Budget
% people          Persons potentially reached per media
% costs           Unit costs
% maxuse         Maximum use

```

```

% quality                Perception quality
% minpeople
%
% RESULTS
%
% For an interpretation of the results, set PriLev > 1, for example:
% Result = publicitycampaignEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~seveaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 1, 2005.  Last modified Dec 1, 2005.

function Result = publicitycampaignEx(PriLev)

if nargin < 1
    PriLev = 1;
end

budget      = 250000;
people      = [12000 1500 2000 6000 3000 9000]';
costs       = [1500 8000 12000 9000 24000 51000]';
maxuse     = [4 2 8 60 4 8]';
quality    = [3 7 8 2 6 9]';
minpeople   = 100000;

Prob = publicitycampaign(budget, people, costs, maxuse, quality, minpeople);
Result = tomRun('cplex', Prob, PriLev);

if PriLev >1,
    invest = Result.x_k;
    for i = 1:length(invest),
        if invest(i) ~= 0,
            disp(['invest in ' num2str(invest(i)) ' uses of media number ' num2str(i)])
        end
    end
end
end

```

% MODIFICATION LOG

%

% 051201 med Created.

% 060116 per Added documentation.

% 060125 per Moved disp to end

### 27.3 Portfolio selection

```
% function Result = portfolioselectionEx(PriLev)
%
% Creates a TOMLAB MIP problem for portfolio selection
%
% PORTFOLIO SELECTION
%
% A consultant in finance has to choose for one of his wealthy female
% clients a certain number of shares in which to invest. She wishes
% to invest $ 100,000 in 6 different shares. The consultant estimates
% for her the return on investment that she may expect for a period
% of six months. The following table gives for each share its country
% of origin, the category (T: technology, N: non-technology) and the
% expected return on investment (ROI). The client specifies certain
% constraints. She wishes to invest at least $ 5,000 and at most
% $ 40,000 into any share. She further wishes to invest half of her
% capital in European shares and at most 30% in technology. How
% should the capital be divided among the shares to obtain the
% highest expected return on investment?
%
% List of shares
%
% +---+-----+-----+-----+
% |Nr|Origin |Category|Expected ROI|
% +---+-----+-----+-----+
% | 1|Japan  |  T    |  5.3%    |
% | 2|UK      |  T    |  6.2%    |
% | 3|France  |  T    |  5.1%    |
% | 4|USA     |  N    |  4.9%    |
% | 5|Germany |  N    |  6.5%    |
% | 6|France  |  N    |  3.4%    |
% +---+-----+-----+-----+
%
% VARIABLES
%
% budget                Budget
% mininvest             Minimal investment
% maxinvest             Maximal investment
% catinvest1min        Minimal investment in category One (N)
% idx1cat               Index of category One
% catinvest2max        Maximal investment in category Two (T)
% idx2cat               Index of category Two
% returns               Expected ROI
%
% RESULTS
%
```

```

% For an interpretation of the results, try the following:
% Result = portfolioselectionEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 1, 2005.   Last modified Dec 1, 2005.

function Result = portfolioselectionEx(PriLev)

if nargin < 1
    PriLev = 1;
end

budget      = 100000;
mininvest   = 5000;
maxinvest   = 40000;
catinvest1min = 0.5;
idx1cat     = [0 1 1 0 1 1]';

catinvest2max = 0.3;
idx2cat     = [1 1 1 0 0 0]';

returns     = [5.3 6.2 5.1 4.9 6.5 3.4]';

Prob = portfolioselection(budget, mininvest, maxinvest, catinvest1min, idx1cat,...
    catinvest2max, idx2cat, returns);

Prob.MIP.SC = 1:Prob.N; % All variables are semi-continuous

Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    invest = Result.x_k;
    for i = 1:length(invest),
        if invest(i) ~= 0,
            disp(['invest $ ' num2str(invest(i)) ' in share ' num2str(i)])
        end
    end
end

```

```
    end
  end
end
```

```
% MODIFICATION LOG
```

```
%
```

```
% 051201 med   Created.
```

```
% 060116 per   Added documentation.
```

```
% 060125 per   Moved disp to end
```

## 27.4 Financing an early retirement

```

% function Result = financinganearlyretirementEx(PriLev)
%
% Creates a TOMLAB MIP problem for financing an early retirement scheme
%
% FINANCING AN EARLY RETIREMENT SCHEME
%
% The National Agricultural Bank (NAB) decides to establish an early
% retirement scheme for fifteen employees who are taking early
% retirement. These employees will retire over a period of seven
% years starting from the next year. To finance this early retirement
% scheme, the bank decides to invest in bonds for this seven-year
% period. The necessary amounts to cover the pre-retirement leavers
% are given in the following table; they have to be paid at the
% beginning of every year.
%
% Amounts required every year
%
% +-----+-----+-----+-----+-----+-----+
% |Year          | 1| 2| 3| 4| 5| 6| 7|
% +-----+-----+-----+-----+-----+-----+
% |Amount (in 1000 $)|1000|600|640|480|760|1020|950|
% +-----+-----+-----+-----+-----+-----+
%
% For the investments, the bank decides to take three different types
% of bonds, SNCF bonds, Fujitsu bonds, and Treasury bonds. The money
% that is not invested in these bonds is placed as savings with a
% guaranteed yield of 3.2%. The table below lists all information
% concerning the yield and the durations of the bonds and also the
% value of a bond. In this type of bond, it is only possible to buy
% an integer number of bonds, and the invested capital remains locked
% in for the total duration of the bond. Every year, only the
% interest on the capital is distributed. The person in charge of the
% retirement plan decides to buy bonds at the beginning of the first
% year, but not in the following years. How should he organize the
% investments in order to spend the least amount of money to cover
% the projected retirement plan?
%
% Information about loans
%
% +-----+-----+-----+-----+
% |Loan      |Value of bonds (in k$)| Interest|Duration|
% +-----+-----+-----+-----+
% |SNCF      |      1.0              |  7.0%  | 5 years|
% |Fujitsu   |      0.8              |  7.0%  | 4 years|
% |Treasury  |      0.5              |  6.5%  | 6 years|

```

```

% +-----+-----+-----+-----+
%
% VARIABLES
%
% amounts          Retirement costs
% baseinterest     Interest from money not invested in bonds
% values           Values of a bond
% interest         Interest from bonds
% duration         Years to place the money
%
% RESULTS
%
% For an interpretation of the results, try the following:
% Result = financinganearlyretirementEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure.

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 1, 2005.   Last modified Dec 1, 2005.

function Result = financinganearlyretirementEx(PriLev)

if nargin < 1
    PriLev = 1;
end

amounts      = [1000 600 640 480 760 1020 950]*1000;
baseinterest = 3.2;
values       = [1000 800 500]';
interest     = [7.0 7.0 6.5]';
duration     = [5.0 4.0 6.0]';

Prob = financinganearlyretirement(amounts, baseinterest, values, interest, duration);
Prob.MIP.cpxControl.EPGAP = 1e-10;
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,

```

```

b          = 3; % number of bonds
y          = 7; % number of years
buy_bonds = Result.x_k(1:b);
buy_other = Result.x_k(b+1:b+1+y-1);

for i = 1:length(buy_bonds),
    if buy_bonds(i) ~= 0,
        disp(['buy ' num2str(buy_bonds(i)) ' bonds of type ' ...
              num2str(i)])
    end
end

for i = 1:length(buy_other),
    if buy_other(i) ~= 0,
        disp(['buy "other things" for $ ' ...
              num2str(buy_other(i)) ' year ' num2str(i)])
    end
end
end

% MODIFICATION LOG
%
% 051201 med    Created.
% 060116 per    Added documentation.
% 060125 per    Moved disp to end

```

## 27.5 Family budget

```
% function Result = familybudgetEx(PriLev)
%
% Creates a TOMLAB MIP problem for family budget
%
% FAMILY BUDGET
%
% The mother of a family wishes to use her sons computer. On the
% internet she has found optimization software and would now like to
% formulate a mathematical model to help plan their annual budget.
% She has prepared a list of the monthly expenses and receipts of
% money. Every month the expenses for living are $ 550. The monthly
% rent for the apartment is $ 630. She also budgets $ 135 for
% telephone every two months, $ 850 for gas and electricity bills
% every six months, $ 340 per month for the car and $ 100 of tax
% every four months. Her receipts of money are a monthly payment of
% $ 150 of state allowance for families with dependent children and a
% net salary of $ 1900 per month. She knows that she pays at least
% $ 165 for leisure every month (subscription to the swimming pool
% for the older children, football club for the youngest, gym for
% herself) but she would like to spend more (restaurant, cinema,
% holidays). How should the budget be balanced all through the year
% to maximize the money available for leisure?
%
% VARIABLES
%
% costs                Various costs
% costfreq             Frequency of the costs
% income               Various incomes
% incomefreq           Frequency of the income
% minhobby             Mimial cost for hobbies
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = familybudgetEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev             Print Level
%
% OUTPUT PARAMETERS
```

```

% Result      Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 1, 2005.   Last modified Dec 1, 2005.

function Result = familybudgetEx(PriLev)

if nargin < 1
    PriLev = 1;
end

costs      = [550 630 135 850 340 100]';
costfreq   = [ 1  1  2  6  1  4]';
income     = [150 1900]';
incomefreq = [ 1  1]';
minhobby   = 165;

Prob = familybudget(costs, costfreq, income, incomefreq, minhobby);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    m      = 12 ; % number of months
    leisure = Result.x_k(1:m);
    save    = Result.x_k(m+1:2*m);
    disp('Money for leisure and for savings per month')
    for i = 1:m,
        disp(['month ' num2str(i) ', leisure ' num2str(leisure(i)) ', save ' num2str(save(i))])
    end
    disp(['total leisure this year: ' num2str(sum(leisure))])
    disp(['total to save this year: ' num2str(sum(save))  ])
end

% MODIFICATION LOG
%
% 051201 med   Created.
% 060116 per   Added documentation.
% 060117 per   Minor change of documentation.
% 060125 per   Moved disp to end

```

## 27.6 Choice of expansion projects

```

% function Result = choiceofexpansionprojectsEx(PriLev)
%
% Creates a TOMLAB MIP problem for choice of expansion projects
%
% CHOICE OF EXPANSION PROJECTS
%
% The large company Tatayo in the north of Italy has specialized in
% the construction of cars for more than ten years. The company
% wishes to expand and has issued internally a call for proposals for
% expansion projects for a planning period of five years. Among the
% many, often cranky, propositions the management has retained five
% projects. Every project has an annual cost and is designed to
% produce a benefit after five years. The first table below gives a
% list of the projects with short descriptions and the expected
% benefit after five years. The forecast annual costs of the projects
% for the next five years are detailed in the second table below,
% together with the funds available. Which project(s) should the
% management choose now to maximize the total benefit after five
% years?
%
% Estimated benefits of the projects (in million $)
%
% +-----+-----+-----+-----+-----+
% |Project|Description                |Expected benefit|
% +-----+-----+-----+-----+-----+
% |  1  |Expand assembly line        |    10.8        |
% |  2  |Reorganize the main shop    |     4.8        |
% |  3  |New painting facilities     |     3.2        |
% |  4  |Research for a new concept car|    4.44        |
% |  5  |Reorganize the logistics chain|   12.25        |
% +-----+-----+-----+-----+-----+
%
% Annual costs of projects and available funds (in million $)
%
% +-----+-----+-----+-----+-----+
% |Project|Year 1|Year 2|Year 3|Year 4|Year 5|
% +-----+-----+-----+-----+-----+
% |  1  | 1.8 | 2.4 | 2.4 | 1.8 | 1.5 |
% |  2  | 1.2 | 1.8 | 2.4 | 0.6 | 0.5 |
% |  3  | 1.2 | 1.0 | 0.0 | 0.48| 0.0 |
% |  4  | 1.4 | 1.4 | 1.2 | 1.2 | 1.2 |
% |  5  | 1.6 | 2.1 | 2.5 | 2.0 | 1.8 |
% +-----+-----+-----+-----+-----+
% |Funds | 4.8 | 6.0 | 4.8 | 4.2 | 3.5 |
% +-----+-----+-----+-----+-----+

```

```

%
% VARIABLES
%
% benefit          Expected benefit
% budget           Funds available each year
% costmat          Cost per project and year
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = choiceofexpansionprojectsEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 1, 2005.   Last modified Dec 1, 2005.

function Result = choiceofexpansionprojectsEx(PriLev)

if nargin < 1
    PriLev = 1;
end

benefit      = [10.8 4.8 3.2 4.44 12.25]*1e6;
budget       = [ 4.8 6.0 4.8  4.2  3.5]*1e6;

costmat      = [1.8 2.4 2.4 1.8 1.5;...
                1.2 1.8 2.4 0.6 0.5;...
                1.2 1.0 0.0 .48 0.0;...
                1.4 1.4 1.2 1.2 1.2;...
                1.6 2.1 2.5 2.0 1.8]*1e6;

Prob = choiceofexpansionprojects(benefit, budget, costmat);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    names = [' Expand assembly line          '];

```

```
    ' Reorganize the main shop      ' ;  
    ' New painting facilities      ' ;  
    ' Research for a new concept car' ;  
    ' Reorganize the logistics chain'];  
idx = find(Result.x_k);  
disp('The management should choose the following projects:')  
disp(names(idx,:))  
end  
  
% MODIFICATION LOG  
%  
% 051201 med   Created.  
% 060117 per   Added documentation.
```

## 27.7 Mean variance portfolio selection

```

% function Result = meanvarianceportfolioselecEx(PriLev)
%
% Creates a TOMLAB MIQP problem for mean variance portfolio selection
%
% MEAN VARIANCE PORTFOLIO SELECTION
%
% An investor wishes to invest a certain amount of money. He is
% evaluating four different securities (assets) for his investment.
% The securities are US Treasury Bills (T-bills), a computer
% hardware company, a computer software company, and a high-risk
% investment in a theater production. He estimates the mean yield on
% each dollar he invests in each of the securities, and also adopts
% the Markowitz idea of getting estimates of the variance/covariance
% matrix of estimated returns on the securities. (For example,
% hardware and software company worths tend to move together, but are
% oppositely correlated with the success of theatrical production, as
% people go to the theater more when they have become bored with
% playing with their new computers and computer games.) The return on
% theatrical productions are highly variable, whereas the T-bill
% yield is certain. The estimated returns and the variance/covariance
% matrix are given in the table below.
%
% Estimated returns and variance/covariance matrix
%
% +-----+-----+-----+-----+-----+
% |           |Hardware|Software|Show-biz|T-bills|
% +-----+-----+-----+-----+-----+
% |Estimated return|    8    |    9    |    12   |    7    |
% +-----+-----+-----+-----+-----+
% |Hardware        |    4    |    3    |    -1   |    0    |
% |Software         |    3    |    6    |    1    |    0    |
% |Show-biz         |   -1    |    1    |    10   |    0    |
% |T-bills          |    0    |    0    |    0    |    0    |
% +-----+-----+-----+-----+-----+
%
% Question 1: Which investment strategy should the investor adopt to
% minimize the variance subject to getting some specified minimum
% target yield?
%
% Question 2: Which is the least variance investment strategy if the
% investor wants to choose at most two different securities (again
% subject to getting some specified minimum target yield)?
%
% VARIABLES
%

```

```

% estreturn          Estimated return of securities
% covmat            The variance/covariance matrix
% target            Minimum target yield
% maxassets         Number of securities
%
% RESULTS
%
% For an interpretation of the results, try the following:
% [Result1, Result2] = meanvarianceportfolioselecEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev           Print Level
%
% OUTPUT PARAMETERS
% Result           Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 2, 2005.   Last modified Dec 2, 2005.

function [Result1, Result2] = meanvarianceportfolioselecEx(PriLev)

if nargin < 1
    PriLev = 1;
end

estreturn    = [8 9 12 7]';
covmat       = [ 4  3 -1  0;...
                3  6  1  0;...
                -1  1 10  0;...
                0  0  0  0];
target       = 8.5;

Prob = meanvarianceportfolioselec(estreturn, covmat, target);
Result1 = tomRun('cplex', Prob, PriLev);

maxassets    = 2;

Prob = meanvarianceportfolioselec(estreturn, covmat, target, maxassets);
Result2 = tomRun('cplex', Prob, PriLev);

if PriLev >1,

```

```

names = ['Hardware';
        'Software';
        'Show-biz';
        'T-bills '];
disp('Answer to Question 1:')
for i = 1:length(Result1.x_k),
    disp(['  invest ' num2str(Result1.x_k(i)*100) ...
         '% of the capital in ' names(i,:) ])
end
disp('Answer to Question 2 (limited number of securities):')
x = Result2.x_k(1:length(Result2.x_k)/2);
x(find(x < 1e-6)) = 0;
for i = 1:(length(Result2.x_k)/2),
    if x(i) ~= 0,
        disp(['  invest ' num2str(x(i)*100) ...
             '% of the capital in ' names(i,:) ])
    end
end
end
end

% MODIFICATION LOG
%
% 051202 med   Created.
% 060117 per   Added documentation.
% 060125 per   Moved disp to end

```

## 28 Timetabling

### 28.1 Assigning personnel to machines

```
% function Result = assigningpersonneltomachinesEx(PriLev)
%
% Creates a TOMLAB MILP problem for assigning personnel to machines
%
% ASSIGNING PERSONNEL TO MACHINES
%
% An operator needs to be assigned to each of the six machines in a
% workshop. Six workers have been pre-selected. Everyone has
% undergone a test of her productivity on every machine. The table
% below lists the productivities in pieces per hour. The machines run
% in parallel, that is, the total productivity of the workshop is the
% sum of the productivities of the people assigned to the machines.
%
% Productivity in pieces per hour
%
% +-----+-----+
% |      | Machines |
% +-----+-----+
% |Workers| 1| 2| 3| 4| 5| 6|
% +-----+-----+
% |  1   |13|24|31|19|40|29|
% |  2   |18|25|30|15|43|22|
% |  3   |20|20|27|25|34|33|
% |  4   |23|26|28|18|37|30|
% |  5   |28|33|34|17|38|20|
% |  6   |19|36|25|27|45|24|
% +-----+-----+
%
% The objective is to determine an assignment of workers to machines
% that maximizes the total productivity. We may start by calculating
% a (non-optimal) heuristic solution using the following fairly
% natural method: choose the assignment p -> m with the highest
% productivity, cross out the line p and the column m (since the
% person has been placed and the machine has an operator), and
% restart this process until we have assigned all persons. The
% problem should then be solved to optimality using Mathematical
% Programming. And finally, solve the same problem to optimality,
% but for machines working in series.
%
% VARIABLES
%
% prodmat          The productivity matrix
%
```

```

% RESULTS
%
% For an interpretation of the results, run:
% [Result1, Result2] = assigningpersonneltomachinesEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 2, 2005.   Last modified Dec 2, 2005.

function [Result1, Result2] = assigningpersonneltomachinesEx(PriLev)

if nargin < 1
    PriLev = 1;
end

prodmat      = [ 13 24 31 19 40 29;...
                18 25 30 15 43 22;...
                20 20 27 25 34 33;...
                23 26 28 18 37 30;...
                28 33 34 17 38 20;...
                19 36 25 27 45 24];

flag = 0; % Parallel

Prob = assigningpersonneltomachines(prodmat, flag);
Result1 = tomRun('cplex', Prob, PriLev);

flag = 1; % Series

Prob = assigningpersonneltomachines(prodmat, flag);
Result2 = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    m = size(prodmat,1); % number of machines
    w = m;                % number of workers
    x1 = reshape(Result1.x_k,m,w);

```

```

disp(['Best parallel work (' num2str(-Result1.f_k) ') when '])
[worker,machine] = find(x1);
for i = 1:length(worker)
    disp([' worker ' num2str(worker(i)) ...
         ' operates machine ' num2str(machine(i))])
end
x2 = reshape(Result2.x_k(1:m*w),m,w);
disp(['Best serial work (' num2str(-Result2.f_k) ') when '])
[worker,machine] = find(x2);
for i = 1:length(worker)
    disp([' worker ' num2str(worker(i)) ...
         ' operates machine ' num2str(machine(i))])
end
end

```

```
% MODIFICATION LOG
```

```

%
% 051202 med Created.
% 060117 per Added documentation.
% 060126 per Moved disp to end

```

## 28.2 Scheduling nurses

```

% function Result = schedulingnursesEx(PriLev)
%
% Creates a TOMLAB MILP problem for scheduling nurses
%
% SCHEDULING NURSES
%
% Mr. Schedule has been asked to organize the schedule of nurses for
% the Cardiology service at St. Josephs hospital. A working day in
% this service is subdivided into twelve periods of two hours. The
% personnel requirement changes from period to period: for instance,
% only a few nurses are required during the night, but the total
% number must be relatively high during the morning to provide
% specific care to the patients. The following table lists the
% personnel requirement for every time period.
%
% Question 1:
% Determine the minimum number of nurses required to cover all the
% requirements, knowing that a nurse works eight hours per day and
% that she is entitled to a break of two hours after she has worked
% for four hours.
%
% Question 2:
% The service only has 80 nurses, which is not sufficient with the
% given requirements. Mr. Schedule therefore proposes that part of
% the personnel works two additional hours per day. These two
% additional hours follow immediately after the last four hours,
% without any break. Determine the schedule of the nurses in this
% service that minimizes the number of nurses working overtime.
%
% Personnel requirement per time period
%
% +-----+-----+-----+-----+
% |Number|Time interval|Minimum number of nurses|
% +-----+-----+-----+-----+
% |  0  | 00am - 02am |          15             |
% |  1  | 02am - 04am |          15             |
% |  2  | 04am - 06am |          15             |
% |  3  | 06am - 08am |          35             |
% |  4  | 08am - 10am |          40             |
% |  5  | 10am - 12pm |          40             |
% |  6  | 12pm - 02pm |          40             |
% |  7  | 02pm - 04pm |          30             |
% |  8  | 04pm - 06pm |          31             |
% |  9  | 06pm - 08pm |          35             |
% | 10  | 08pm - 10pm |          30             |

```

```

% | 11 | 10pm - 12am |          20          |
% +-----+-----+-----+-----+
%
%
% VARIABLES
%
% demand                nurses needed per shift
% hoursperday           hours of work per day
% breakinterval         work this long before break
% breaklength           length of break
% intervallength        length of an interval
% maxdemand             max number of nurses in Q 2
%
% RESULTS
%
% For an interpretation of the results, run:
% [Result1, Result2] = schedulingnursesEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev                Print Level
%
% OUTPUT PARAMETERS
% Result                Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 2, 2005.   Last modified Dec 2, 2005.

function [Result1, Result2] = schedulingnursesEx(PriLev)

if nargin < 1
    PriLev = 1;
end

demand          = [15 15 15 35 40 40 40 30 31 35 30 20]';
hoursperday     = 8;
breakinterval   = 4;
breaklength     = 2;
intervallength  = 2;
maxdemand       = 80;

flag = 0; % No max on nurses

```

```

Prob = schedulingnurses(demand, hoursperday, breakinterval, breaklength, ...
    intervallength, maxdemand, flag);
Result1 = tomRun('cplex', Prob, PriLev);

flag = 1; % Max on nurses
Prob = schedulingnurses(demand, hoursperday, breakinterval, breaklength, ...
    intervallength, maxdemand, flag);
Result2 = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    x1      = Result1.x_k;                % nurses working
    intervals = length(x1);              % number of intervals
    work     = [x1 circshift(x1,-1) circshift(x1,-3) circshift(x1,-4)];
    work_1   = sum(work)';                % total nurses
    n        = Result2.x_k(1:intervals); % regular shifts
    o        = Result2.x_k(intervals+1:end); % overtime
    work1    = [n circshift(n,-1) ...     % regular -> 4 intervals
                circshift(n,-3) circshift(n,-4)];
    work2    = [o circshift(o,-1) ...     % overtime -> 5 intervals
                circshift(o,-3) circshift(o,-4) circshift(o,-5)];
    work_2   = sum(work1)' + sum(work2)'; % total work

    disp(['Without overtime we need ' num2str(Result1.f_k) ' nurses.'])
    disp([' This is the schedule:'])
    for i = 1:intervals,
        disp([' ' num2str((i-1)*2) '.00: ' num2str(x1(i)) ...
            ' start and ' num2str(work_1(i)) ' are working.' ...
            ' ( demand is ' num2str(demand(i)) ')' ])
    end

    disp(['With overtime we have ' num2str(sum(n)) ...
        ' nurses, and ' num2str(sum(o)) ...
        ' of them will work overtime.'])
    disp([' This is the schedule:'])
    for i = 1:intervals,
        disp([' ' num2str((i-1)*2) '.00: ' num2str(n(i)) ...
            ' start a regular shift, ' num2str(o(i)) ...
            ' start an overtime shift and ' num2str(work_2(i)) ...
            ' are working.' ...
            ' ( demand is ' num2str(demand(i)) ')' ])
    end
end

% MODIFICATION LOG
%
% 051202 med Created.

```

% 060117 per Added documentation.  
% 060126 per Moved disp to end

### 28.3 Establishing a college timetable

```

% function Result = establishingacollegetimetableEx(PriLev)
%
% Creates a TOMLAB MILP problem for establishing a college time table
%
% ESTABLISHING A COLLEGE TIMETABLE
%
% Mr. Miller is in charge of establishing the weekly timetable for
% two classes of the last year in a college. The two classes have
% the same teachers, except for mathematics and sport. In the
% college all lessons have a duration of two hours. Furthermore,
% all students of the same class attend exactly the same courses.
% From Monday to Friday, the slots for courses are the following:
% 8:0010:00, 10:1512:15, 14:0016:00, and 16:1518:15. The
% following table lists the number of two-hour lessons that every
% teacher has to teach the students of the two classes per week.
%
% Number of 2-hour lessons per teacher and class
%+-----+-----+-----+-----+
%|Teacher      |Subject          |Lessons for class 1|Lessons for class 2|
%+-----+-----+-----+-----+
%|Mr Cheese    |English          |1                  |1                  |
%|Mrs Insulin  |Biology          |3                  |3                  |
%|Mr Map       |History-Geography|2                  |2                  |
%|Mr Effoicks  |Mathematics      |0                  |4                  |
%|Mrs Derivate |Mathematics      |4                  |0                  |
%|Mrs Electron |Physics          |3                  |3                  |
%|Mr Wise      |Philosophy       |1                  |1                  |
%|Mr Muscle    |Sport            |1                  |0                  |
%|Mrs Biceps   |Sport            |0                  |1                  |
%+-----+-----+-----+-----+
%
% The sport lessons have to take place on Thursday afternoon from
% 14:00 to 16:00. Furthermore, the first time slot on Monday morning
% is reserved for supervised homework. Mr Effoicks is absent every
% Monday morning because he teaches some courses at another college.
% Mrs Insulin does not work on Wednesday. And finally, to prevent
% students from getting bored, every class may only have one two-hour
% lesson per subject on a single day. Write a mathematical program
% that allows Mr Miller to determine the weekly timetable for the two
% classes.
%
% VARIABLES
%
% lessons1/2      Number of lessons per subject and class
% subject         Subject indices

```

```

% slots                Possible slots
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = establishingacolletimetableEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev            Print Level
%
% OUTPUT PARAMETERS
% Result            Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 5, 2005.   Last modified Dec 5, 2005.

function Result = establishingacolletimetableEx(PriLev)

if nargin < 1
    PriLev = 1;
end

lessons1    = [ 1 3 2 0 4 3 1 1 0]';
lessons2    = [ 1 3 2 4 0 3 1 0 1]';
subject     = [ 1 2 3 4 4 5 6 7 7]';
slots       = 4*5;

Prob = establishingacolletimetable(lessons1, lessons2, subject, slots);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    t_names = ['Cheese   '; 'Insulin  '; 'Map      '];
             ['Effoacks'; 'Derivate '; 'Electron '];
             ['Wise     '; 'Muscle   '; 'Biceps   '];
    s_names = ['English  '; 'Biology  '; 'Histo-Geo '];
             ['Mathematics'; 'Physics  ']; 'Philosophy '];
             ['Sport    '];
    subject = [ 1 2 3 4 4 5 6 7 7]';
    d_names = ['Mon'; 'Tue'; 'Wed'; 'Thu'; 'Fri' ];
    l_times = ['0800-1000'; '1015-1215'; '1400-1600'; '1615-1815'; ];
    t_nr    = length(t_names);

```

```

s_nr = 20 ; % number of slots
temp = reshape(Result.x_k,t_nr*2,s_nr);
class1 = temp(1:t_nr,:);
class2 = temp(t_nr+1:end,:);
classes= [class1 class2];
for c = 1:2,
    this_class = class1;
    if c == 2,
        this_class = class2;
    end
    disp(' ')
    disp(['TIMETABLE FOR CLASS ' num2str(c)])
    counter = 0;
    for i = 1:length(d_names),
        day = d_names(i,:);
        disp(['--' day '--'])
        for j = 1:size(l_times,1),
            time = l_times(j,:);
            counter = counter + 1;
            if sum(this_class(:,counter)) == 1,
                teacher = find(this_class(:,counter));
                disp([' ' time ' ' s_names(subject(teacher),:) ...
                    ' (' t_names(teacher,:) ')'])
            end
        end
    end
end
end
end
end
end

% MODIFICATION LOG
%
% 051205 med Created.
% 060117 per Added documentation.
% 060126 per Moved disp to end

```

## 28.4 Exam scheduling

```

% function Result = examschedulingEx(PriLev)
%
% Creates a TOMLAB MILP problem for exam scheduling
%
% EXAM SCHEDULING
%
% At a technical university every term the third-year students choose
% eight modules from the eleven modules that are taught, depending on
% the option they wish to choose in the fourth year (there are two
% possible choices: "Production planning" and "Quality and security
% management"). In the current term, certain modules are obligatory
% for students who wish to continue with one of these options. The
% obligatory courses are Statistics (S), Graph models and algorithms
% (GMA), Production management (PM), and Discrete systems and events
% (DSE). The optional modules are: Data analysis (DA), Numerical
% analysis (NA), Mathematical programming (MP), C++, Java (J), Logic
% programming (LP), and Software engineering (SE).
%
% Incompatibilities between different exams
%
% +---+---+---+---+---+---+---+---+---+---+---+---+
% |   | DA| NA|C++| SE| PM| J |GMA| LP| MP| S |DSE|
% +---+---+---+---+---+---+---+---+---+---+---+---+
% |DA | - | X | - | - | X | - | X | - | - | X | X |
% |NA | X | - | - | - | X | - | X | - | - | X | X |
% |C++| - | - | - | X | X | X | X | - | X | X | X |
% |SE | - | - | X | - | X | X | X | - | - | X | X |
% |PM | X | X | X | X | - | X | X | X | X | X | X |
% |J  | - | - | X | X | X | - | X | - | X | X | X |
% |GMA| X | X | X | X | X | X | - | X | X | X | X |
% |LP | - | - | - | - | X | - | X | - | - | X | X |
% |MP | - | - | X | - | X | X | X | - | - | X | X |
% |S  | X | X | X | X | X | X | X | X | X | - | X |
% |DSE| X | X | X | X | X | X | X | X | X | X | - |
% +---+---+---+---+---+---+---+---+---+---+---+---+
%
% Mrs Edeetee needs to schedule the exams at the end of the term.
% Every exam lasts two hours. Two days have been reserved for the
% exams with the following time slices: 8:00-10:00, 10:15-12:15,
% 14:00-16:00, and 16:15-18:15. For every exam she knows the set of
% incompatible exams that may not take place at the same time because
% they have to be taken by the same students. These incompatibilities
% are summarized in the table above.
%
% Help Mrs Edeetee construct a timetable so that no student has more

```

```

% than one exam at a time.
%
% VARIABLES
%
% incompatmat          Matrix of incompatibilities
% slots                Timeslots (2 days * 4 per day)
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = examschedulingEx(2);

%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev              Print Level
%
% OUTPUT PARAMETERS
% Result              Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 5, 2005.   Last modified Dec 5, 2005.

function Result = examschedulingEx(PriLev)

if nargin < 1
    PriLev = 1;
end

incompatmat      = [ 0 1 0 0 1 0 1 0 0 1 1;...
    1 0 0 0 1 0 1 0 0 1 1;...
    0 0 0 1 1 1 1 0 1 1 1;...
    0 0 1 0 1 1 1 0 0 1 1;...
    1 1 1 1 0 1 1 1 1 1 1;...
    0 0 1 1 1 0 1 0 1 1 1;...
    1 1 1 1 1 1 0 1 1 1 1;...
    0 0 0 0 1 0 1 0 0 1 1;...
    0 0 1 0 1 1 1 0 0 1 1;...
    1 1 1 1 1 1 1 1 1 0 1;...
    1 1 1 1 1 1 1 1 1 1 0];

slots            = 8;

```

```

Prob = examscheduling(incompatmat, slots);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    e_names = ['DA '; 'NA '; 'C++ '; 'SE ';
              'PM '; 'J  '; 'GMA '; 'LP  ';
              'MP '; 'S  '; 'DSE '];
    exams   = length(e_names); % number of exams
    days    = 2;                % number of days
    times   = ['0800-1000'; '1015-1215'; '1400-1600'; '1615-1815'; ];
    slots   = size(times,1);    % time slots per day
    temp    = reshape(Result.x_k,exams,days*slots);

    for d = 1:days,
        disp([' day number ' num2str(d)])
        for s = 1:slots,
            i = s + (slots)*(d-1);
            exams_now = find(temp(:,i));
            e_list = [];
            for e = 1:length(exams_now),
                e_list = [e_list e_names(exams_now(e),:)];
            end
            if ~isempty(e_list),
                disp([' during ' times(s,:) ': ' num2str(e_list)])
            end
        end
    end
end

% MODIFICATION LOG
%
% 051205 med    Created.
% 060117 per    Added documentation.
% 060126 per    Moved disp to end

```

## 28.5 Production planning with personnel assignment

```

% function Result = prodplanwithpersonnelassignEx(PriLev)
%
% Creates a TOMLAB MILP problem for production planning with personnel assignment
%
% PRODUCTION PLANNING WITH PERSONNEL ASSIGNMENT
%
% The company Line Production decides to plan the production of four
% of its products (P1, P2, P3, P4) on its five production lines (L1
% to L5). The company gains net profits of $ 7 for the products P1
% and P4, $ 8 for P2, and $ 9 for P3. The maximum times during
% which the five production lines may operate are different during
% the planning period. The maximum capacities for L1 to L5 are 4500
% hours, 5000 hours, 4500 hours, 1500 hours, and 2500 hours
% respectively. The table below lists the processing time in hours
% necessary for the production of one unit of every product on every
% production line. Which quantities of P1 to P4 should be produced to
% maximize the total profit? If subsequently a transfer of personnel
% (and hence of working hours) is authorized between production lines
% during the planning period as shown in the second table below,
% which is the maximum profit? How many hours are transferred and
% under what conditions?
%
% Unitary processing times
%
% +-----+-----+
% |          | Lines          |
% +-----+-----+
% |Products| L1| L2| L3| L4| L5|
% +-----+-----+
% |  P1    |1.3|0.9|2.0|0.3|0.9|
% |  P2    |1.8|1.7|1.4|0.6|1.1|
% |  P3    |1.3|1.2|1.3|1.0|1.4|
% |  P4    |0.9|1.1|1.0|0.9|1.0|
% +-----+-----+
%
% Possible transfers of personnel
%
% +-----+-----+
% |          | Destination          |
% +-----+-----+Maximum number of |
% |Origin| L1| L2| L3| L4| L5|transferable hours|
% +-----+-----+
% | L1   | -|yes|yes|yes| no|    400    |
% | L2   | no| -|yes| no|yes|    800    |
% | L3   |yes|yes| -|yes| no|    500    |

```

```

% | L4 | no| no| no| -|yes|      200      |
% | L5 |yes|yes|yes| no| -|      300      |
% +-----+-----+-----+-----+-----+
%
% VARIABLES
%
% profit                Profit per product
% capacity              Hours at each site
% timemat               Time to produce products
% transfermat           Transfer matrix
% maxtransfer           Maximum hours to transfer
%
% RESULTS
%
% For an interpretation of the results, run:
% [Result1, Result2] = prodplanwithpersonnelassignEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev                Print Level
%
% OUTPUT PARAMETERS
% Result                Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 5, 2005.   Last modified Dec 5, 2005.

function [Result1, Result2] = prodplanwithpersonnelassignEx(PriLev)

if nargin < 1
    PriLev = 1;
end

profit      = [7 8 9 7]';
capacity    = [4500 5000 4500 1500 2500]';

timemat     = [ 1.3 0.9 2.0 0.3 0.9;...
               1.8 1.7 1.4 0.6 1.1;...
               1.3 1.2 1.3 1.0 1.4;...
               0.9 1.1 1.0 0.9 1.0];

transfermat = [0 1 1 1 0;...

```

```

0 0 1 0 1;...
1 1 0 1 0;...
0 0 0 0 1;...
1 1 1 0 0];

maxtransfer = [400 800 200 500 300]';

flag = 0;

Prob = prodplanwithpersonnelassign(profit, capacity, timemat,...
    transfermat, maxtransfer, flag);
Result1 = tomRun('cplex', Prob, PriLev);

flag = 1;

Prob = prodplanwithpersonnelassign(profit, capacity, timemat,...
    transfermat, maxtransfer, flag);
Result2 = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    disp('without transfer')
    x1 = Result1.x_k;
    for i = 1:length(x1),
        if x1(i) > 0,
            disp(['    produce ' num2str(x1(i)) ' units of P' num2str(i) ])
        end
    end
    disp('with transfer')
    x2 = Result2.x_k;
    length(x2);
    n1 = 5;
    n2 = 4;
    n = n1 + n2*n2 + n2; %amount, transfers, hours
    x2 = Result2.x_k;
    for i = 1:length(x1),
        if x2(i) > 0,
            disp(['    produce ' num2str(x2(i)) ' units of P' num2str(i) ])
        end
    end
    transfer = reshape(x2(n1:n1+n1*n1-1),n1,n1);
    for i = 1:size(transfer,1),
        idx = find(transfer(i,:));
        for j = 1: length(idx),
            disp(['    transfer ' num2str(transfer(i,idx(j))) ...
                ' hours from ' num2str(i) ' to ' num2str(idx(j)) ])
        end
    end
end

```

```
end
hours = x2(n1+n1*n1:end);
for j = 1:length(hours),
    disp(['    work ' num2str(hours(j)) ' hours at L' num2str(j) ])
end
end
```

```
% MODIFICATION LOG
```

```
%
% 051205 med    Created.
% 060117 per    Added documentation.
% 060126 per    Moved disp to end
% 060131 per    Really moved disp to end
```

## 28.6 Plan the personnel at a construction site

```

% function Result = planthepersonnelataconstrEx(PriLev)
%
% Creates a TOMLAB MILP problem for planning the personnel at a construction site
%
% PLANNING THE PERSONNEL AT A CONSTRUCTION SITE
%
% Construction workers who erect the metal skeleton of skyscrapers
% are called steel erectors. The following table lists the
% requirements for steel erectors at a construction site during a
% period of six months. Transfers from other sites to this one are
% possible on the first day of every month and cost $100 per person.
% At the end of every month workers may leave to other sites at a
% transfer cost of $160 per person. It is estimated that
% understaffing as well as overstaffing cost $200 per month per post
% (in the case of unoccupied posts the missing hours have to be
% filled through overtime work).
%
% Monthly requirement for steel erectors
%
% +---+---+---+---+---+---+
% |Mar|Apr|May|Jun|Jul|Aug|
% +---+---+---+---+---+---+
% | 4 | 6 | 7 | 4 | 6 | 2 |
% +---+---+---+---+---+---+
%
% Overtime work is limited to 25% of the hours worked normally. Every
% month, at most three workers may arrive at the site. The departure
% to other sites is limited by agreements with labor unions to 1/3 of
% the total personnel of the month. We suppose that three steel
% erectors are already present on site at the end of February, that
% nobody leaves at the end of February and that three workers need to
% remain on-site at the end of August. Which are the number of
% arrivals and departures every month to minimize the total cost?
%
% VARIABLES
%
% transferin           Cost to transfer in
% transferout         Cost to transfer out
% staffingdevcost     Cost for over or under employment
% overtimemax         Maximum overtime
% maxtransferin       Maximum amount to transfer in
% maxtransferout      Maximum amount to transfer out
% startstaff          Starting staff
% endstaff            Staff required at the end of the period
% demands             Staff required each month

```

```

%
% RESULTS
%
% For an interpretation of the results, run:
% Result = planthepersonnelataconstrEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev      Print Level
%
% OUTPUT PARAMETERS
% Result      Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 5, 2005.   Last modified Dec 5, 2005.

function Result = planthepersonnelataconstrEx(PriLev)

if nargin < 1
    PriLev = 1;
end

transferin      = 100;
transferout     = 160;
staffingdevcost = 200;

overtimemax    = 0.25;
maxtransferin  = 3;
maxtransferout = 1/3;

startstaff     = 3;
endstaff       = 3;

demands        = [4 6 7 4 6 2]';

Prob = planthepersonnelataconstr(transferin, transferout,...
    staffingdevcost, overtimemax, maxtransferin, maxtransferout, ...
    startstaff, endstaff, demands);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    months = ['Mar'; 'Apr'; 'May'; 'Jun'; 'Jul'; 'Aug'];

```

```

temp = reshape(Result.x_k,size(months,1),5);
for m = 1:size(months,1),
    disp(['-- ' months(m,:) ' --' ])
    disp([' ' num2str(temp(m,1)) ' worker(s) on site'])
    disp([' ' num2str(temp(m,2)) ' worker(s) have arrived'])
    disp([' ' num2str(temp(m,3)) ' worker(s) will leave'])
    disp([' ' num2str(temp(m,4)) ' worker(s) too many'])
    disp([' ' num2str(temp(m,5)) ' worker(s) too few'])
end
end

% MODIFICATION LOG
%
% 051205 med Created.
% 060118 per Added documentation.
% 060126 per Moved disp to end

```

## 29 Public Services

### 29.1 Water conveyance

```

% function Result = waterconveyanceEx(PriLev)
%
% Creates a TOMLAB MILP problem for water conveyance
%
% WATER CONVEYANCE / WATER SUPPLY MANAGEMENT
%
% The graph in the figure below shows a water transport network. The
% nodes, numbered from 1 to 10, represent the cities, the reservoirs,
% and the pumping stations connected by a network of pipes. The three
% cities Gotham City, Metropolis, and Spider Ville are supplied from
% two reservoirs. The availabilities of water from these reservoirs
% in thousands of m3/h are 35 for reservoir 1 and 25 for reservoir 2.
% The capacity of each pipe connection is indicated in thousands of
% m3/h in the graph.
%
% A study is undertaken to find out whether the existing network will
% be able to satisfy the demands of the cities in ten years time,
% that is 18, 15, and 20 thousand m3/h. Determine the maximum flow in
% the current network. Will it be sufficient in ten years from now?
%
% Water transport network
%
%
%
%      35          20          15          7          18
% Reservoir-1 ----> 3 -----> 4 -----> 8-Gotham city
%
%          | \15 |          \          ----+
%          | \  |10          ----\ /
%      12| \  |          10          / \10
%          \ \ |          / \10
%          \ V V -----/ -----+
%      25          \ /          15          V 15
% Reservoir-2 ---+> 5 -----> 9-Metropolis
%          | 6 \          15          ^
%          \ \          -----+
%          \ \          \ / 10
%          \ \12          X
%      22 \ \          / \
%          V V          / -----+
%
%          22          10          V 20
%          6 -----> 7 -----> 10-Spider Ville
%
%
%
%

```

```

% VARIABLES
%
% arcs_out/in          For an arc i arcs_out(i) is its target node
%                      and arcs_in(i) its starting node
% capacity            Capacity of an arc
% source              Artificial node acting as a source
% sink                Artificial node acting as a sink
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = waterconveyanceEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev             Print Level
%
% OUTPUT PARAMETERS
% Result             Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 5, 2005.   Last modified Dec 5, 2005.

function Result = waterconveyanceEx(PriLev)

if nargin < 1
    PriLev = 1;
end

arcs_in      = [ 1  1  1  2  2  3  3  4  4  5  5  5  6  7  7  8  9 10 11 11]';
arcs_out     = [ 3  5  6  5  6  4  5  8  9  8  9 10  7  9 10 12 12 12  1  2]';

capacity     = [20 15 12  6 22 15 10  7 10 10 15 15 22 10 10 18 15 20 35 25]';
source       = 11;
sink         = 12;

Prob = waterconveyance(arcs_in, arcs_out, capacity, source, sink);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    x      = Result.x_k;
    names  = ['Reservoir 1 ']; ...

```

```

'Reservoir 2 '; ...
'Node 3      '; ...
'Node 4      '; ...
'Node 5      '; ...
'Node 6      '; ...
'Node 7      '; ...
'Gotham City '; ...
'Metropolis  '; ...
'Spider Ville'; ...
'Production  '; ... % this is the source node
'Consumption']; % this is the sink node
disp('Maximum flow of the network is as follows:')
for i = 1:length(arcs_in),
    if x(i) ~= 0,
        disp([names(arcs_in(i),:) ' -> ' names(arcs_out(i),:) ': ' num2str(x(i)) ])
    end
end
end
end

% MODIFICATION LOG
%
% 051205 med    Created.
% 060117 per    Added documentation.
% 060125 per    Moved disp to end

```

## 29.2 CCTV surveillance

```

% function Result = cctvsurveillanceEx(PriLev)
%
% Creates a TOMLAB MILP problem for cctv surveillance
%
% CCTV SURVEILLANCE
%
% In the course of the last few months, the industrial zone of
% Billston has suffered from a series of break-ins and thefts during
% the night. The zone is watched by security men but there are too
% few of them. The town council in charge of security in this zone
% decides to install surveillance cameras to aid the security men
% with their task. These cameras can be directed and pivot through
% 360 degrees. By installing a camera at the intersection of several
% streets, it is possible to survey all adjoining streets. The map in
% the figure below shows the industrial zone with the limits of the
% zone to be covered by closed circuit TV (CCTV) surveillance and the
% 49 possible locations where to install the cameras. What is the
% minimum number of cameras that have to be installed to survey all
% the streets of this zone and where should they be placed?
%
% The industrial zone in Billston
%
% 13 -- 14 -- 18 -- 17      28 -- 29      35 -- 36
%
% |      |      |          |          |
% |      |      |          |          |
% |
% |      15 -- 19          26 -- 27      34      48
% |
% | / |      |          |          |      |
% | / |      |          |          |      |
%
% 12      16 -- 20      24 -- 25 -- 30      33      47 -- 45 -- 46
%
% | /          |          /          | / |          |
% | /          |          /          | / |          |
%
% 3 -- 11 -- 21 -- 22          31      37 -- 43 -- 44 -- 49
%
% | \          \          |      |
% | \          \          |      |
% |
% |      4 -- 9 -- 10          23 -- 32 -- 38
% |
% |      | \          |      |

```

```

% | | \ | |
% |
% | 6 5 39 -- 40 -- 41 -- 42
% |
% | | \ | /
% | | \ | /
% | 8 7 | /
% | | /
% | | /
% |
% 1 ----- 2
%
% VARIABLES
%
% arcs_out/in These variables describe the network
% of streets
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = cctvsurveillanceEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev Print Level
%
% OUTPUT PARAMETERS
% Result Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 5, 2005. Last modified Dec 5, 2005.

function Result = cctvsurveillanceEx(PriLev)

if nargin < 1
    PriLev = 1;
end

arcs_in = [1 1 2 2 3 3 3 3 4 4 4 6 6 9 11 12 12 13 14 14 15 15 16 ...

```

```

17 18 19 20 21 22 22 23 24 25 25 26 26 28 30 31 31 32 32 ...
33 33 34 35 37 37 38 39 40 41 43 44 44 45 45 47]';

arcs_out = [2 3 39 41 4 11 12 16 5 6 9 7 8 10 21 13 15 14 15 18 16 ...
19 20 18 19 20 21 22 23 25 32 25 26 30 27 28 29 31 32 33 ...
38 39 34 37 35 36 38 43 40 40 41 42 44 49 45 46 47 48]';

Prob = cctvsurveillance(arcs_in, arcs_out);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    disp('Put cameras in nodes ')
    disp(num2str(find(Result.x_k)))
end

% MODIFICATION LOG
%
% 051205 med Created.
% 060118 per Added documentation.
% 060125 per Moved disp to end

```

### 29.3 Rigging elections

```

% function Result = riggingelectionsEx(PriLev)
%
% Creates a TOMLAB MILP problem for rigging elections
%
% RIGGING ELECTIONS
%
% In a country not very far away, the party of the duke Sark Mevo has
% finally beaten the ruling coalition headed by the princess Reguel
% Tekris. Mevo wishes to consolidate his position in the capital, the
% fourteen quarters of which need to be grouped to electoral
% districts. A schematic map of the capital is given in the figure
% below. The quarters are numbered from 1 to 14. The two other
% numbers are the forecast number of favorable votes for Mevo and the
% total number of electors per quarter. All electors must vote and
% the winner needs to have the absolute majority. A valid electoral
% district is formed by several adjacent quarters and must have
% between 30,000 and 100,000 voters. Two quarters that touch each
% other just at a point like 12 and 13 are not considered adjacent.
% Electoral districts consisting of a single quarter are permitted if
% it has at least 50,000 voters. Nevertheless, Mevo may not decently
% define an electoral district solely with quarter 10, since this
% contains his residence. Determine a partitioning into five
% electoral districts that maximizes the number of seats for Mevo.
% Should this cause any difficulties, try a partitioning into six
% districts. Snirp, the mathematical jester, suggests Mevo uses
% Mathematical Programming...
%
% +-----+-----+-----+-----+-----+
% | 1       | | 6 | 7 12000/30000 | |
% | 17500/30000 | | 9000/+-----+-----+
% +-----+ |40000 | 8       | |
% | 2       | | | | 10000/30000 | 9 |
% | 15000/50000 | +-----+-----+-----+ 26000/40000 |
% +-----+ 5 | | | 11 | |
% | 3       | 18000/ | 10 | 2500/ +-----+
% | 14200/20000 | 20000 | 34000/| 10000 | 12 27000/60000|
% +-----+-----+ 60000 +-----+-----+
% | 4       | | | 13 | 14 |
% | 42000/70000 | | | 29000/| 15000/40000 |
% + | | | 40000 | |
% +-----+-----+-----+-----+
%
% Map of the capital and its quarters.
% Legend: quarter number, supporters/electorate
%

```

```

% VARIABLES
%
% in/out           Quarter in(i) borders to quarter out(i)
% population       Population of each quarter
% votes           Supporters er quarter
% minpop           Min size of an electoral district
% maxpop           Max size of an electoral district
% minsingle        Min size of a quarter to be a district
% districtsnum     Number of districts wanted
% illegalsingle    This quarter may not be single
% districts        The possible partition of the quarters
%                 into desired number of districts.
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = riggingelectionsEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 5, 2005. Last modified Dec 5, 2005.

function Result = riggingelectionsEx(PriLev)

if nargin < 1
    PriLev = 1;
end

in = [1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 8 9 9 10 10 11 11 12 13]';
out = [2 5 3 5 4 5 5 10 6 10 7 8 8 9 9 10 11 11 12 11 13 12 13 14 14]';

population = [30 50 20 70 20 40 30 30 40 60 10 60 40 40]';
votes      = [17.5 15 14.2 42 18 9 12 10 26 34 2.5 27 29 15]';
minpop     = 30;
maxpop     = 100;
minsingle  = 50;

```

```

districtsnum = 6;
illegalsingle = 10;

districts = [];
rowlen = length(population);
maxq = rowlen;

for q=1:maxq
    if (population(q) >= minsingle & q ~= illegalsingle)
        districts = [districts; zeros(1,maxq)];
        districts(end, q) = 1;
    end
    idx = find(in == q);
    for i = 1:length(idx)
        p = out(idx(i));
        if (population(q) + population(p) <= maxpop)
            if (population(q) + population(p) >= minpop)
                districts = addNeighbor(districts, q, p, rowlen);

                idx2 = find(in == p);
                for j = 1:length(idx2)
                    r = out(idx2(j));
                    if (population(q) + population(p) + population(r) <= maxpop)
                        if (population(q) + population(p) + population(r) >= minpop)
                            districts = addNeighbor(districts, q, p, rowlen, r);
                        end
                    end
                end
            end
        end
    end
end
end

Prob = riggingelections(districts, votes, population, districtsnum);
Result = tomRun('cplex', Prob, PriLev);
Result.districts = districts;

if PriLev > 1,
    temp = Result.districts(find(Result.x_k),:);
    disp('to rig the elections:')
    for d = 1:size(temp,1),
        qs = find(temp(d,:));
        disp([' merge the quarters ' num2str(qs) ' into a district.'])
    end
end

function districts = addNeighbor(districts, q, p, rowlen, r)

```

```
if nargin <5
    districts = [districts; zeros(1,rowlen)];
    districts(end, [q, p]) = [1 1];
else
    districts = [districts; zeros(1,rowlen)];
    districts(end, [q, p, r]) = [1 1 1];
end
```

```
% MODIFICATION LOG
```

```
%  
% 051205 med    Created.  
% 060118 per    Added documentation.  
% 060125 per    Moved disp to end
```

## 29.4 Gritting roads

```

% function Result = grittingroadsEx(PriLev)
%
% Creates a TOMLAB MILP problem for gritting roads
%
% GRITTING ROADS
%
% In the case of ice, all the streets of a village need to be
% gritted. A schematic map of the streets is given in the figure
% below. The nodes correspond to street intersections, the arcs to
% the streets that need to be gritted. The arcs are labeled with the
% length of the street in meters.
%
% Graph of the village streets
%
%      --150->   --130->   --100->
%    1         2         3         4
%           <-140-- <-100--
%  | ^       /| ^       ^       | ^
%  | 165   / | 170     |       | 180
%  | |   / | |       200     | |
% 165| 230 160|       |       190|
%  | | /   | |       |       | |
%  | | /   | |       |       | |
%  V |V     V |       |       V |
%      --144->   --128->
%    5         6         7 --109-> 8
%      <-144-- <-122--
%  ^       /| ^       ^| ^       / |
% 194   / |174   / | |       / |
%  |   / | |   / 185|185 /   |
%  | 218 174| 233 | | /   190
%  | /   | | /   | | /   140 |
%  | /   | | /   | | /   |
%  | V     V | /     V |V     V
%
%    9 --148-> 10 <-135- 11 -110-> 12
%
% The highway maintenance depot with the gritting truck is located at
% intersection number 1. The truck has a sufficiently large capacity
% to grit all the streets during a single tour. Due to the one-way
% streets, it may be forced to pass several times through the same
% street that has already been gritted. Determine a tour for the
% gritting truck of minimum total length that passes through all
% streets. For bidirectional streets, both directions need to be
% gritted separately.

```

```

%
% VARIABLES
%
% in/out          Road i starts in in(i) and
%                goes out to out(i)
% lengths        Lengths of the roads
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = grittingroadsEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev         Print Level
%
% OUTPUT PARAMETERS
% Result         Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 6, 2005.   Last modified Dec 6, 2005.

function Result = grittingroadsEx(PriLev)

if nargin < 1
    PriLev = 1;
end

in = [1 1 2 2 2 3 3 4 4 5 5 6 6 6 6 6 7 7 7 7 8 8 8 9 9 10 10 11 11 12]';
out = [2 5 3 5 6 2 4 3 8 1 6 2 5 7 9 10 3 6 8 11 4 11 12 5 10 6 7 7 10 11]';

lengths = [150 165 130 230 160 140 100 100 190 165 144 170 144 128 218 174 ...
           200 122 109 185 180 141 190 194 148 174 233 185 135 110]';

Prob = grittingroads(in, out, lengths);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    twice = find(Result.x_k > 1);
    disp('all roads travelled once, except:')
    for i = 1:length(twice),
        idx = twice(i);

```

```
        disp([' road from ' num2str(in(idx)) ' to ' num2str(out(idx)) ...  
            ' ( that is travelled ' num2str(Result.x_k(idx)) ' times)'])  
    end  
end
```

```
% MODIFICATION LOG
```

```
%  
% 051206 med    Created.  
% 060118 per    Added documentation.  
% 060125 per    Moved disp to end
```

## 29.5 Location of income tax offices

```

% function Result = locationofincometaxofficesEx(PriLev)
%
% Creates a TOMLAB MILP problem for location of income tax offices
%
% LOCATION OF INCOME TAX OFFICES
%
% The income tax administration is planning to restructure the
% network of income tax offices in a region. The graph in the figure
% below shows the cities in the region and the major roads. The
% numbers within ( ) close to the cities indicate the population in
% thousands of inhabitants. The arcs are labeled with the distances
% in kilometers. The income tax administration has determined that
% offices should be established in three cities to provide sufficient
% coverage. Where should these offices be located to minimize the
% average distance per inhabitant to the closest income tax office?
%
% Graph of towns and roads of the region
%
%
%   (15)   (10)   (12)   (18)
%   1 -15- 2 -22- 3 -18- 4
%
%   | \       / |   |
%   | 24   16 |   |
%   18  \   /  |   |
%   |           20  12
%   |     5(5) |   |
%   |           |   |
%   |   | \   |   |
%   |   12 24 |   |
%   |   | \ |   |
%
%   7 -15- 8 -30- 9 -12- 6 (24)
% (11)   (16)   (13)
%   |   | / | /
%   22  25 19 19 22
%   |   | / | /
%
%   10 -19- 11 -21- 12 (20)
% (22)   (19)
%
% VARIABLES
%
% population           Population of each town
% numloc               Number of offices to start

```

```

% lengths          The length of the roads
% in/out           A road i goes between towns
%                 in(i) and out(i)
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = locationofincometaxofficesEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 6, 2005.   Last modified Dec 6, 2005.

function Result = locationofincometaxofficesEx(PriLev)

if nargin < 1
    PriLev = 1;
end

population = [15 10 12 18 5 24 11 16 13 22 19 20]';
numloc     = 3;
in         = [1 1 1 2 3 3 3 4 5 5 6 6 7 7 8 8 9 9 10 11]';
out        = [2 5 7 3 4 5 9 6 8 9 9 12 8 10 9 11 11 12 11 12]';
lengths    = [15 24 18 22 18 16 20 12 12 24 12 22 15 22 30 ...
              25 19 19 19 21]';

Prob = locationofincometaxoffices(population, numloc, in, out, lengths);
Result = tomRun('cplex', Prob, PriLev);

if PriLev > 1,
    cities = length(population);
    temp   = Result.x_k(1:cities);
    build  = find(temp);
    goto   = reshape(Result.x_k(1+cities:end),cities,cities);
    disp(['Build the offices in towns ' num2str(build) ' and let'])
    for i = 1:length(build),

```

```
        disp(['  people from ' num2str(find(goto(build(i),:))) ...  
            ' travel to ' num2str(build(i)) ])  
    end  
end
```

```
% MODIFICATION LOG
```

```
%  
% 051206 med    Created.  
% 060118 per    Added documentation.  
% 060125 per    Moved disp to end
```

## 29.6 Efficiency of hospitals

```

% function Result = efficiencyofhospitalsEx(PriLev)
%
% Creates a TOMLAB MILP problem for efficiency of hospitals
%
% EFFICIENCY OF HOSPITALS
%
% The administration of the hospitals in Paris decides to measure the
% efficiency of the surgery departments in four major hospitals with
% a desire to improve the service to the public. To keep this study
% anonymous, the hospitals are named H1 to H4. The method suggested
% to measure the efficiency is DEA (Data Envelopment Analysis). This
% method compares the performance of a fictitious hospital with the
% performances of the four hospitals.
%
% Three initial indicators (resources) are taken into account: the
% number of non-medical personnel, the general expenses, and the
% available number of beds. In addition, four final indicators
% (services) are analyzed: the number of hospital admissions per day,
% the number of consultations in the outpatients clinic, the number
% of nurses on duty every day, and the number of interns and doctors
% on duty every day. The corresponding data have been analyzed over a
% period of two years and the numbers representing a day of average
% activity in every hospital are given in the following two tables.
%
% Resource indicators
%
% +-----+-----+-----+-----+-----+
% |           | H1 | H2 | H3 | H4 |
% +-----+-----+-----+-----+-----+
% |Non-medical personnel| 90 | 87 | 51 | 66 |
% |General expenses (k$)|38.89|109.48|40.43|48.41|
% |Number of beds      | 34 | 33 | 20 | 33 |
% +-----+-----+-----+-----+-----+
%
% Service indicators
%
% +-----+-----+-----+-----+-----+
% |           | H1 | H2 | H3 | H4 |
% +-----+-----+-----+-----+-----+
% |Admissions          |30.12|18.54|20.88|10.42|
% |Consultations       |13.54|14.45| 8.52|17.74|
% |Interns and doctors| 13 | 7 | 8 | 26 |
% |Nurses on duty     | 79 | 55 | 47 | 50 |
% +-----+-----+-----+-----+-----+
%

```

```

% Justify through the DEA method how hospital H2 is performing
% compared to the others.
%
% VARIABLES
%
% resources          A matrix describing the resources
% services          A matrix describing the services
%
% RESULTS
%
% For an interpretation of the results, run:
% Result = efficiencyofhospitalsEx(2);
%
% REFERENCES
%
% Applications of optimization... Gueret, Prins, Seveaux
% http://web.univ-ubs.fr/lester/~sevaux/pl/index.html
%
% INPUT PARAMETERS
% PriLev          Print Level
%
% OUTPUT PARAMETERS
% Result          Result structure

% Marcus Edvall, Tomlab Optimization Inc, E-mail: tomlab@tomlab.biz
% Copyright (c) 2005-2005 by Tomlab Optimization Inc., $Release: 5.0.0$
% Written Dec 6, 2005.  Last modified Dec 6, 2005.

function Result = efficiencyofhospitalsEx(PriLev)

if nargin < 1
    PriLev = 1;
end

resources = [90 87 51 66;...
            38.89 109.48 40.43 48.41;...
            34 33 20 33];
services = [30.12 18.54 20.88 10.42;...
           13.54 14.45 8.52 17.74;...
           13 7 8 26;...
           79 55 47 50];

indices = zeros(size(resources,2),1);

for i=1:size(resources,2)
    Prob = efficiencyofhospitals(resources, services, i);
    Result = tomRun('cplex', Prob, PriLev);
end

```

```
    indices(i,1) = Result.x_k(end,1);
    Result.indices = indices;
end

if PriLev > 1,
    temp = Result.indices;
    for i = 1:length(temp),
        disp(['H' num2str(i) ' is ' num2str(100*temp(i)) '% efficient'])
    end
end
```

```
% MODIFICATION LOG
```

```
%
% 051206 med    Created.
% 060118 per    Added documentation.
% 060125 per    Moved disp to end
```

## Part III

# Additional downloads: lp\_prob, milp\_prob and qp\_prob.

## 30 Introduction to additional downloads

At the [TOMLAB downloads page](#) three zip files with more problems can be found. Many of these are quite large and may be good as test cases for benchmarking. A package of about 250 MB with mixed LP/MILP are also available from your TOMLAB representative.

## 31 Additional linear programming test problems: lp\_prob.

Download [lp\\_prob.zip](#) and extract to a folder, for example `tomlab/testprob2/lp_prob`.

The folder will then contain about 90 test problems, of which one is `cre-c.mps`. To solve this problem with TOMLAB enter the following in Matlab:

```
[F, c, A, b_L, b_U, x_L, x_U, IntVars] = cpx2mat('cre-c.mps',0);
Prob = lpAssign(c, A, b_L, b_U, x_L, x_U, [], 'Additional lp_prob: cre-c');
Result = tomRun('cplex', Prob, 1);
```

The problems in this zip-file have up to almost 3700 variables and more than 3000 constraints.

## 32 Additional mixed-integer linear programming test problems: milp\_prob.

Download [milp\\_prob.zip](#) and extract to a folder, for example `tomlab/testprob2/milp_prob`.

The folder will then contain about 90 test problems, of which one is `blp-ic97.mps`. To solve this problem with TOMLAB enter the following in Matlab:

```
[F, c, A, b_L, b_U, x_L, x_U, IntVars] = cpx2mat('blp-ic97.mps',0);
Prob = mipAssign(c, A, b_L, b_U, x_L, x_U, [], 'Additional milp-problem: blp-ic97');
Result = tomRun('cplex', Prob, 1);
```

The problems in this zip-file have up to some 14000 variables and about 16000 constraints.

## 33 Additional quadratic programming test problems: qp\_prob.

Download [qp\\_prob.zip](#) and extract to a folder, for example `tomlab/testprob2/qp_prob`.

The folder will then contain about 130 test problems, of which one is `liswet1.qps`. To solve this problem with TOMLAB enter the following in Matlab:

```
[F, c, A, b_L, b_U, x_L, x_U] = cpx2mat('liswet1.qps',0);  
Prob = qpAssign(F, c, A, b_L, b_U, x_L, x_U, [], 'Additional qp-problem: liswet1.qps');  
Result = tomRun('cplex', Prob, 1);
```

The problems in this zip-file have up to some 20000 variables and about 12000 constraints.