Montogue

Tutorial MAT2
Linear Programming with MATLAB

Lucas Monteiro Nogueira

• Summary •	
Problem 1	Linear Programming - Minimization
Problem 2	Linear Programming - Maximization
Problem 3	Linear Programming – Changing to standard form 1
Problem 4	Linear Programming – Changing to standard form 2
Problem 5	An unbounded variable
Problem 6	Linear Programming – Problem-based approach 1
Problem 7	Linear Programming – Problem-based approach 2
Problem 8	Quadratic programming – Portfolio optimization
Problem 9	Quadratic programming 2

PROBLEMS

▶ Problem 1 – Linear programming - Minimization

Solve the following linear program using MATLAB's *linprog* function and Mathematica's *LinearOptimization* function.

$$\min f = x_1 + 2x_2 - x_4 + x_5$$

$$\text{s.t.} = \begin{cases} 2x_1 + x_2 + 3x_3 + 2x_4 = 5 \\ x_1 - x_2 + 2x_3 - x_4 + 3x_5 = 1 \\ x_1 - 2x_3 - 2x_5 = -1 \\ x_i \geq 0,\ i = 1, 2, ..., 5 \end{cases}$$

▶ Solution

In the present case, we first note that the objective function to be optimized is $f = x_1 + 2x_2 + 0x_3 - x_4 + x_5$, so we begin by typing the vector

```
f = [1,2,0,-1,1]
```

Next, notice that all constraints in the problem at hand are equalities, not inequalities. This implies that inequality constraints *A* and *B* should be assigned empty matrices,

```
A = []; B = []
```

The coefficient matrices Aeq and Beq of the equality constraints, in turn, can be easily gleaned from the three constraint equations,

```
Aeq = [2 1 3 2 0; 1 -1 2 -1 3; 1 0 -2 0 -2];
Beq = [5 1 -1]';
```

The lower bound vector xm is assembled by noting that all five variables must be greater than zero,

```
xm = zeros(5,1)
```

We can then implement our code:

```
f = [1,2,0,-1,1]; A = []; B = [];
Aeq = [2 1 3 2 0; 1 -1 2 -1 3; 1 0 -2 0 -2];
Beq = [5 1 -1]';
xm = zeros(5,1);
[x fval flag]=linprog(f,A,B,Aeq,Beq,xm)
```

The code returns the solution

```
x = (0.5714, 0, 0, 1.9286, 0.7857)
```

which means that x_1 = 0.571, x_2 = 0, x_3 = 0, x_4 = 1.93, x_5 = 0.786. The code also returns the optimized objective function fval = −0.571 and an exitflag of 1, which indicates, of course, that the solution has converged to some vector x.

▶ Problem 2 – Linear programming – Maximization

Solve the following linear program using MATLAB's *linprog* function.

$$\max f = 2x_1 + x_2 - 3x_3 + 5x_4$$

$$\text{s.t.} = \begin{cases} x_1 + 2x_2 + 2x_3 + 4x_4 \le 40 \\ 2x_1 - x_2 + x_3 + 2x_4 \le 8 \\ 4x_1 - 2x_2 + x_3 - x_4 \le 10 \\ x_i \ge 0,\ i = 1, 2, ..., 4 \end{cases}$$

▶ Solution

The first caveat to notice in this problem is that it involves a maximization of the objective function, but linear programming functions such as MATLAB's *linprog* and Mathematica's *LinearProgramming* can only minimize the objective function. This issue can be easily resolved if we multiply the objective function by −1 and proceed to minimize it; after all, finding the constrained maximum of the objective function is the same as finding the constrained minimum of −1 × the objective function, so long as we remember to swap the sign of the latter after carrying out the calculations. We proceed to type the objective function

```
f = [2 1 -3 5]
```

Notice that the constraints here are all inequalities, so they should be assigned to inputs A and B (unlike the previous problem, in which the constraints were *equalities* and hence had to be assigned to inputs A_{eq} and B_{eq}).

```
A = [1 2 2 4; 2 -1 1 2; 4 -2 1 -1];
B = [40 8 10]'
```

Inputs Aeq and Beq are empty,

```
Aeq = []; Beq = []
```

Lower bound xm is included to ensure that all variables are nonnegative,

```
xm = zeros(4,1);
```

The final code is listed below,

```
f = [-2 -1 3 -5];
A = [1 2 2 4; 2 -1 1 2; 4 -2 1 -1];
B = [40 8 10]';
Aeq = []; Beq = [];
xm = zeros(4,1);
[x f0 flag]=linprog(f,A,B,Aeq,Beq,xm)
```

This code returns the solution

```
x = (0, 6, 0, 7)
```

which means that x_1 = 0, x_2 = 6, x_3 = 0, x_4 = 7. The code also returns the optimized objective function fval = −41.0, which, as mentioned above, must have its sign swapped in order to yield the "true" maximum fval, namely, 41.0.

▶ Problem 3 – Linear programming – Changing to standard form 1

Consider the following linear programming problem.

$$\min f = x_1 - 2x_2$$

$$\text{s.t. } -4x_1 + 6x_2 \le 9$$

$$x_1 + x_2 \le 4$$

$$x_1, x_2 \le 0$$

Convert the problem to standard form and solve it using MATLAB.

▶ Solution

In order to change the problem to standard form, we introduce slack variables s_1 and s_2 into the first two constraint inequalities so as to convert them to equations,

$$\begin{aligned}
\min f &= x_1 - 2x_2 \\
-4x_1 + 6x_2 + s_1 &= 9 \\
x_1 + x_2 + s_2 &= 4 \\
x_1, x_2, s_1, s_2 &\geq 0
\end{aligned}$$

Note that the standard form of the LP formulation requires the design variables to be nonnegative. We can solve the LP with the following MATLAB code,

```
f = [1 -2 0 0]; A = []; B = []';
Aeq = [-4 6 1 0; 1 1 0 1]; Beq = [9 4]';
xm = zeros(4,1);
[x f0 flag]=linprog(f,A,B,Aeq,Beq,xm)
```

This returns the solution $x_1 = 1.5$, $x_2 = 2.5$, $s_1 = 0$, $s_2 = 0$ and the objective function value $f0 = -3.5$.

▶ Problem 4 – Linear programming – Changing to standard form 2

Convert the following linear program to standard form and solve it using MATLAB.

$$\begin{aligned}
\max f &= 4x_1 + 5x_2 \\
\text{s.t. } 2x_1 + x_2 &\leq 6 \\
x_1 + 2x_2 &\leq 5 \\
x_1 + x_2 &\geq 1 \\
x_1 + 4x_2 &\geq 2 \\
x_1 \geq 0,\ x_2 &\geq 0
\end{aligned}$$

▶ Solution

In order to change the problem to standard form, we introduce slack variables s_1 and s_2 into the first two constraint inequalities so as to convert them to equations,

$$\begin{aligned}
\min f &= x_1 - 2x_2 \\
-4x_1 + 6x_2 + s_1 &= 9 \\
x_1 + x_2 + s_2 &= 4 \\
x_1, x_2, s_1, s_2 &\geq 0
\end{aligned}$$

Note that the standard form of the LP formulation requires the design variables to be nonnegative. We proceed to solve the LP problem with the following MATLAB code,

```
f = [1 -2 0 0]; A = []; B = []';
Aeq = [-4 6 1 0; 1 1 0 1]; Beq = [9 4]';
xm = zeros(4,1);
[x f0 flag]=linprog(f,A,B,Aeq,Beq,xm)
```

This returns the solution $x_1 = 1.5$, $x_2 = 2.5$, $s_1 = 0$, $s_2 = 0$ and the objective function value $f0 = -3.5$.

▶ Problem 5 – An unbounded variable

Convert the following linear program to standard form and solve it using MATLAB.

$$\begin{aligned}
\min f &= x_1 - 2x_2 + 3x_3 \\
\text{s.t. } -4x_1 + 6x_2 + x_3 &\leq 9 \\
x_1 + x_2 - 2x_3 &\leq 4 \\
x_1, x_2 &\geq 0
\end{aligned}$$

▶ Solution

One crucial aspect that distinguishes this problem from previous ones is that variable x_3 is unbounded. One way to proceed in such cases is to replace the unbounded variable with a difference of two slack variables s_1 and s_2, that is, $x_3 = s_1 - s_2$. We must also include two other slack variables s_3 and s_4 to convert the two inequality constraints to equations,

$$-4x_1 + 6x_2 + (s_1 - s_2) + s_3 = 9$$
$$x_1 + x_2 - 2\times(s_1 - s_2) + s_4 = 4$$
$$x_1, x_2, s_1, s_2, s_3, s_4 \geq 0$$

Simplifying,

$$-4x_1 + 6x_2 + s_1 - s_2 + s_3 + 0s_4 = 9$$
$$x_1 + x_2 - 2s_1 + 2s_2 + 0s_3 + s_4 = 4$$
$$x_1, x_2, s_1, s_2, s_3, s_4 \geq 0$$

The equation to minimize is of course

$$\min f = x_1 - 2x_2 + 3(s_1 - s_2) = x_1 - 2x_2 + 3s_1 - 3s_2$$

The pertaining MATLAB code is shown below.

```
f = [1 -2 3 -3 0 0]; A = []; B = []';
Aeq = [-4 6 1 -1 1 0; 1 1 -2 2 0 1];
Beq = [9 4]';
xm = zeros(6,1);
[x f0 flag]=linprog(f,A,B,Aeq,Beq,xm)
```

This returns x_1 = 0, x_2 = 1.6923, s_1 = 0, s_2 = 1.1538, s_3 = 0, s_4 = 0.

▶ Problem 6 – Linear programming – Problem-based approach 1

Solve the following linear program using MATLAB's problem-based format.

$$\min -3x_1 + 4x_2 - 2x_3 + 5x_4$$
$$\text{s.t. } 4x_1 - x_2 + 2x_3 - x_4 = -2$$
$$x_1 + x_2 - x_3 + 2x_4 \leq 14$$
$$2x_1 - 3x_2 - x_3 - x_4 \geq -2$$
$$x_1, x_2, x_3 \geq -1;\ x_4 \text{ unconstrained}$$
$$x_1, x_2, x_3 \geq -1;\ x_4 \text{ unconstrained}$$

▶ Solution

The MATLAB code that solves this linear program is given below.

```
f = [-3 4 -2 5];
A = [1 1 -1 2; -2 3 1 1];
B = [14 2]';
Aeq = [4 -1 2 -1]; Beq = [-2]';
xm = [-1, -1, -1, -Inf]';
[x f0 flag]=linprog(f,A,B,Aeq,Beq,xm)
```

This returns the solution vector x_1 = -1.0, x_2 = 2.5, x_3 = -1.0, x_4 = -6.5 and the optimized objective function fval = -17.5.

The code works well, but we were asked to work out the problem with MATLAB's problem-based format. Starting with version R2017b, linear and quadratic programs can be solved with this new approach. In a problem-based framework, one way to proceed is to enter the variables using the function *optimvar*; lower or upper bounds are included as well.

```
x1 = optimvar('x1', 'LowerBound', -1);
x2 = optimvar('x2', 'LowerBound', -1);
x3 = optimvar('x3', 'LowerBound', -1);
x4 = optimvar('x4');
```

Then, we initiate a blank optimization problem with the command *optimproblem* and include the objective function:

```
linprob = optimproblem('Objective', -3*x1+4*x2-
2*x3+5*x4);
```

Then, we write down the constraints and append them to *linprob*:

```
linprob.Constraints.cons1 = 4*x1-x2+2*x3-x4 == -2;
linprob.Constraints.cons2 = 4*x1+x2-x3+2*x4 <= 14;
linprob.Constraints.cons3 = 2*x1-3*x2-x3-x4 >= -2;
```

Finally, we use the command *solve* to have MATLAB solve the problem at hand.

```
sols = solve(linprob)
>> Solving problem using linprog.
Optimal solution found.
     x0 =
        -1.0000
         2.5000
        -1.0000
        -6.5000
```

The solution is identical to the one obtained by employing *linprog* directly, as it should be.

▶ Problem 7 – Linear programming – Problem-based approach 2

Solve the following linear program using MATLAB's problem-based format.

$$\max f = 4x_1 + 5x_2$$

$$\text{s.t. } 2x_1 + x_2 \le 6$$
$$x_1 + 2x_2 \le 5$$
$$x_1 + x_2 \ge 1$$
$$x_1 + 4x_2 \ge 2$$
$$x_1 \ge 0,\ x_2 \ge 0$$

▶ Solution

This linear program can be easily solved with the problem-based format introduced in Problem 6; note that maximizing $f = 4x_1 + 5x_2$ is essentially the same as minimizing $f = -4x_1 - 5x_2$.

```
x1 = optimvar('x1', 'LowerBound', 0);
x2 = optimvar('x2', 'LowerBound', 0);
linprob = optimproblem('Objective', -4*x1 - 5*x2);
linprob.Constraints.cons1 = 2*x1 + x2 <= 6;
linprob.Constraints.cons2 = x1 + 2*x2 <= 5;
linprob.Constraints.cons3 = x1 + x2 >= 1;
linprob.Constraints.cons4 = x1 + 4*x2 >= 2;
sols = solve(linprob);
>> Optimal solution found.
sols =
     2.333
     1.333
```

The optimal values are $x_1 = 2.333$ and $x_2 = 1.333$.

▶ Problem 8 – Quadratic programming – Portfolio optimization

Consider three securities with the expected returns given in the following table.

Expected return	Security 1 ($i = 1$)	Security 2 ($i = 2$)	Security 3 ($i = 3$)
μ_i	9.73%	6.57%	5.37%

The covariances of returns are tabulated below.

Covariance σ_{ij}	i = 1	i = 2	i = 3
i = 1	0.0256	0.0033	0.00019
i = 2		0.0135	−0.00027
i = 3			0.00125

Propose a portfolio with minimum variance and short selling that achieves an expected return of at least 5.5%.

▶ Solution

We wish to form a portfolio with minimum variance with short selling allowed that achieves an expected return of at least 5.5%. The corresponding model is

$$\begin{gathered}\min f = 0.0256x_1^2 + 0.0135x_2^2 + 0.00125x_3^2 + 2\times 0.0033x_1x_2 \\ +2\times 0.00019x_1x_3 + 2\times\left(-0.00027\right)x_2x_3 \\ \text{s.t. } 0.0973x_1 + 0.0657x_2 + 0.0537x_3 \geq 0.055 \\ x_1 + x_2 + x_3 = 1\end{gathered}$$

Turning to MATLAB, a quadratic programming exercise can be solved with the general syntax

```
[x,fval,exitflag] =
quadprog(Q,f,A,B,Aeq,Beq,xm,xM,options)
```

In the input arguments, Q is the matrix of the quadratic terms of the objective function; f is the matrix of the linear terms of the objective function; A and B are the coefficient matrix and right-hand side coefficients vector in the inequality constraints; A_{eq} and B_{eq} are the coefficient matrix and right-hand side coefficients in the equality constraints; and xm and xM are the decision variable lower and upper bounds, respectively. After the optimization process, results are returned as vector x, the optimized objective function value is returned as fval, and exitflag is an index that should equal unity if the problem solution converged to a solution x.

The pertaining code is shown in continuation.

```
Q = [0.0256 0.0033 0.00019; 0.0033, 0.0135, -0.00027;
0.00019, -0.00027, 0.00125];
f = [0 0 0];
A = [-0.0973 -0.0653 -0.0537];
B = [-0.055];
Aeq = [1 1 1];
Beq = [1];
xm = []; xM = [];
[x, fval] = quadprog(Q,f,A,B,Aeq,Beq,xm,xM)
```

This code returns x_1 = 2.39%, x_2 = 9.22%, and x_3 = 88.39%. The objective function can be evaluated with these optimal inputs to yield 5.42×10^{-4}. Note that fval needs to be doubled to equal the variance of the portfolio since the quadratic term (variance) in the objective has a coefficient of 0.5. The expected return of the portfolio is

$$r_p = 0.0973\times 0.0239 + 0.0657\times 0.0922 + 0.0537\times 0.8839 = \boxed{5.58\%}$$

We conclude that the optimal portfolio meets the return goal of 5.5% and it will not be possible to have another portfolio that achieves an expected return of 5.58% with lower variance. It is noteworthy that the optimal portfolio allocates most of the investment into the third asset, which is associated with returns (5.37%) close to the desired 5.5%. Less investment is apportioned to assets 1 and 2, which are comparably riskier.

Much like the linear programming problems 6 and 7, MATLAB can also solve a quadratic programming problem with the problem-based approach. The pertaining code is shown below.

```
P = optimproblem;
x = optimvar('x',3,1,'LowerBound',0);
```

```
P.Objective=0.0256*x(1)^2+0.0135*x(2)^2+0.00125*x(3)^2
+2*0.0033*x(1)*x(2)+2*0.00019*x(1)*x(3)-
2*0.00027*x(2)*x(3);
P.Constraints.cons1=0.0973*x(1)+0.0657*x(2)+0.0537*x(3
)>=0.055;
P.Constraints.cons2=x(1)+x(2)+x(3)==1;
sols=solve(P); x0=sols.x
```

This returns the following solution vector,

```
x0 =

    0.0239
    0.0922
    0.8839
```

which is identical to the vector obtained in the direct solution approach, as it should be.

▶ Problem 9 – Quadratic programming 2

Use MATLAB to solve the following quadratic optimization problem.

$$\min x_1^2 + 10x_2^2 + 2.5x_1x_2 + 2x_1 + 3x_2$$
$$\text{s.t. } x_1 - x_2 \le 1$$
$$x_1 + 2x_2 \ge 100$$
$$x_{1,2} \ge 0$$

▶ Solution

The problem is restated below.

$$\min 1\times x_1^2 + 10x_2^2 + 2.5x_1x_2 + 2x_1 + 3x_2$$
$$\text{s.t. } x_1 - x_2 \le 1$$
$$-x_1 - 2x_2 \le -100$$
$$x_{1,2} \ge 0$$

Matrix ***Q*** is shown next:

$$\mathbf{Q} = \begin{bmatrix} 2\times 1 & 2.5 \\ 2.5 & 2\times 10 \end{bmatrix} = \begin{bmatrix} 2 & 2.5 \\ 2.5 & 20 \end{bmatrix}$$

The pertaining MATLAB code is listed below.

```
Q = [2 2.5; 2.5 20];
f = [2 3];
A = [1 -1; -1 -2];
B = [1; -100];
Aeq = [];
Beq = [];
xm = [0 0]; xM = [];
[x, fval] = quadprog(Q,f,A,B,Aeq,Beq,xm,xM)
```

This returns the solution vector x_1 = 34.0, x_2 = 33.0. The optimized objective function value is fval = 15,018. Alternatively, we may use the problem-based approach:

```
xm = [0 0];
P = optimproblem('ObjectiveSense','min');
x = optimvar('x',2,1,'LowerBound',xm);
P.Objective =
x(1)^2+10*x(2)^2+2.5*x(1)*x(2)+2*x(1)+3*x(2);
P.Constraints.cons1 = x(1)-x(2) <= 1;
P.Constraints.cons2 = x(1)+2*x(2) >= 100;
sols = solve(P);
x0=sols.x
```

The resulting solution vector is identical to the one obtained via *quadprog*.

REFERENCES

- MISHRA, S.K. and RAM, B. (2018). *Introduction to Linear Programming with MATLAB*. Boca Raton: CRC Press.
- THIE. P.R. and KEOUGH, G.E. (2008). *An Introduction to Linear Programming and Game Theory*. 3rd edition. Hoboken: John Wiley and Sons.
- XUE, D. (2020). *Solving Optimization Problems with MATLAB*. Boston/Berlin: Walter De Gruyter.