Utah State University

DigitalCommons@USU

All ECSTATIC Materials

ECSTATIC Repository

Spring 2020

Lecture Notes: Nonlinear Optimization and Matlab Optimization Toolbox Example

Lina Sela

The University of Texas at Austin, linasela@utexas.edu

Recommended Citation

Sela, Lina, "Lecture Notes: Nonlinear Optimization and Matlab Optimization Toolbox Example" (2020). *All ECSTATIC Materials.* Paper 82.
https://digitalcommons.usu.edu/ecstatic_all/82

CE 385D Water Resources Planning and Management **Spring 2020**

Lecture 09: Nonlinear optimization and Matlab optimization toolbox

Instructor: Dr. Lina Sela

Model summary:

Here we will solve the example provided in Chapter 4.3.3, Daniel P. Loucks and Eelco van Beek 2017, using Lagrange multipliers and Matlab optimization toolbox. The problem is to find the operating and allocation policy that maximizes the net benefits. The optimization problem can be formulated as:

$$\begin{array}{ll}
\text{Maximize} & (12-p_1)p_1 + (20-1.5p_2)p_2 + (28-2.5p_3)p_3 \\
& -3p_1^{1.3} - 5p_2^{1.2} - 6p_3^{1.15} \\
\text{subject to} & p_1 \le 0.4x_1^{0.9} \\
& p_2 \le 0.5x_2^{0.8} \\
& p_3 \le 0.6x_3^{0.7} \\
& x_1 + x_2 + x_3 + R \le Q \\
& x_1, x_2, x_3 \ge 0 \\
& p_1, p_2, p_3 \ge 0
\end{array}$$

where x_i the amount of water allocated to each firm i, p_i is the amount of product produced by each firm i, R is the amount that must remain in the river, and Q is the total amount of available water.

Solution using Lagrange multipliers:

Assuming that available water is a binding constraint and that all variables are positive, we can write the modified optimization problem:

$$\begin{array}{lll}
\text{Maximize} & (12-p_1)p_1 + (20-1.5p_2)p_2 + (28-2.5p_3)p_3 & \\
& -3p_1^{1.3} - 5p_2^{1.2} - 6p_3^{1.15} & \\
\text{subject to} & p_1 - 0.4x_1^{0.9} = 0 & (\lambda_1) \\
& p_2 - 0.5x_2^{0.8} = 0 & (\lambda_2) \\
& p_3 - 0.6x_3^{0.7} = 0 & (\lambda_3) \\
& x_1 + x_2 + x_3 + R - Q = 0 & (\lambda_4)
\end{array}$$

There are four equality constraints and we add a Lagrange multiplier λ_i for each constraint i. The Lagrangian function is:

$$\begin{aligned}
\mathcal{L}(p,x,\lambda) = (12-p_1)p_1 + (20-1.5p_2)p_2 + (28-2.5p_3)p_3 \\
-3p_1^{1.3} - 5p_2^{1.2} - 6p_3^{1.15} \\
-\lambda_1\left(p_1 - 0.4x_1^{0.9}\right) - \lambda_2\left(p_2 - 0.5x_2^{0.8}\right) - \lambda_3\left(p_3 - 0.6x_3^{0.7}\right) \\
-\lambda_4\left(x_1 + x_2 + x_3 + R - Q\right)\}
\end{aligned}$$

Find the gradient of the Lagrange function and set it to zero, $\nabla\mathcal{L}(p, x, \lambda) = 0$.

$$\begin{array}{lcl}
\frac{\partial\mathcal{L}}{\partial p_1} = 0 & & 12 - 2p_1 - 1.3 \times 3 \times p_1^{0.3} - \lambda_1 = 0 \\
\frac{\partial\mathcal{L}}{\partial p_2} = 0 & & 20 - 3p_2 - 1.2 \times 5 \times p_2^{0.2} - \lambda_2 = 0 \\
\frac{\partial\mathcal{L}}{\partial p_3} = 0 & & 28 - 5p_3 - 1.15 \times 6 \times p_3^{0.15} - \lambda_3 = 0 \\
\frac{\partial\mathcal{L}}{\partial x_1} = 0 & & \lambda_1 \times 0.9 \times 0.4 \times x_1^{-0.1} - \lambda_4 = 0 \\
\frac{\partial\mathcal{L}}{\partial x_2} = 0 & \rightarrow & \lambda_2 \times 0.8 \times 0.5 \times x_2^{-0.2} - \lambda_4 = 0 \\
\frac{\partial\mathcal{L}}{\partial x_3} = 0 & & \lambda_3 \times 0.7 \times 0.6 \times x_3^{-0.3} - \lambda_4 = 0 \\
\frac{\partial\mathcal{L}}{\partial \lambda_1} = 0 & & p_1 - 0.4 \times x_1^{0.9} = 0 \\
\frac{\partial\mathcal{L}}{\partial \lambda_2} = 0 & & p_2 - 0.5 \times x_2^{0.8} = 0 \\
\frac{\partial\mathcal{L}}{\partial \lambda_3} = 0 & & p_3 - 0.6 \times x_3^{0.7} = 0 \\
\frac{\partial\mathcal{L}}{\partial \lambda_4} = 0 & & x_1 + x_2 + x_3 + R - Q = 0
\end{array}$$

These ten equations are the necessary conditions for an optimal solution. They can be solved to obtain the values of the ten unknown variables. We can solve them using the `fsolve` function.

Steps to solve the equations:

1. `fsolve` function solves a system of nonlinear equations of the form: $F(x) = 0$, where x is a vector and F is a function that returns a vector value.
2. Syntax: `x = fsolve (fun,x0)`, where `x0` is the initial guess and `fun` is a function **handle** or a function **name** that accepts a vector `x` and returns a vector `F` , the nonlinear equations evaluated at `x` .
3. Combine all variables into one vector $x = [p_1, p_2, p_3, x_1, x_2, x_3, \lambda_1, \lambda_2, \lambda_3, \lambda_4]$.
4. Write a function that computes the left-hand side of these ten equations in the form: $F(x) = 0$. Here for simplicity `RQ` stands for `R-Q`.

```
function [F] = lagrange(x,RQ)
    F(1) = 12 - 2*x(1) - 1.3*3*x(1)^0.3 - x(7);
    F(2) = 20 - 3*x(2) - 1.2*5*x(2)^0.2 - x(8);
    F(3) = 28 - 5*x(3) - 1.15*6*x(3)^0.15 - x(9);
    F(4) = x(7)*0.9*0.4*x(4)^-0.1 - x(10);
    F(5) = x(8)*0.8*0.5*x(5)^-0.2 - x(10);
    F(6) = x(9)*0.7*0.6*x(6)^-0.3 - x(10);
    F(7) = x(1) - 0.4*x(4)^0.9;
    F(8) = x(2) - 0.5*x(5)^0.8;
    F(9) = x(3) - 0.6*x(6)^0.7;
    F(10) = x(4) + x(5) + x(6) - RQ;
end
```

5. Define the initial guess:

```
x0 = ones(10,1);
```

6. Define `RQ`:

```
RQ = 10;
```

7. Define function handle:

```
fun = @(x)lagrange(x,RQ);   % note the the function has two inputs but only x ...
    is the vector of variables and RQ is a parameter
```

8. Call the solver:

```
x = fsolve(fun,x0);
```

9. Compute the objective function: once we solved the equations we found the optimal solution. We can substitute the solution into the objective function and compute the optimal value.

```
% compute objective function
f = (12 - x(1))*x(1) + (20 - 1.5*x(2))*x(2) + (28 - 2.5*x(3))*x(3) ...
    -(3*x(1)^1.3 + 5*x(2)^1.2 + 6*x(3)^1.15);
```

10. The final code:

```
clc
clear
% solving 10 nonlinear equations 4.35-4.44
% x = [p1 p2 p3 x1 x2 x3 l1 l2 l3 l4]

n = 10;      % number of variables
RQ = 10;     % available water

fun = @(x)lagrange(x,RQ);
x0 = ones(n,1);
x = fsolve(fun,x0);

% compute objective function
f = (12 - x(1))*x(1) + (20 - 1.5*x(2))*x(2) + (28 - 2.5*x(3))*x(3) ...
    -(3*x(1)^1.3 + 5*x(2)^1.2 + 6*x(3)^1.15);

P = x(1:3);
X = x(4:6);
Lambda = x(7:10);

function [F] = lagrange(x,RQ)
    F(1) = 12 - 2*x(1) - 1.3*3*x(1)^0.3 - x(7);
    F(2) = 20 - 3*x(2) - 1.2*5*x(2)^0.2 - x(8);
    F(3) = 28 - 5*x(3) - 1.15*6*x(3)^0.15 - x(9);
    F(4) = x(7)*0.9*0.4*x(4)^-0.1 - x(10);
    F(5) = x(8)*0.8*0.5*x(5)^-0.2 - x(10);
    F(6) = x(9)*0.7*0.6*x(6)^-0.3 - x(10);
    F(7) = x(1) - 0.4*x(4)^0.9;
    F(8) = x(2) - 0.5*x(5)^0.8;
    F(9) = x(3) - 0.6*x(6)^0.7;
    F(10) = x(4) + x(5) + x(6) - RQ;
end
```

Final solution: $p_1 = 0.46, p_2 = 1.44, p_3 = 1.88, x_1 = 1.16, x_2 = 3.74, x_3 = 5.10, \lambda_1 = 8.01, \lambda_2 = 9.24, \lambda_3 = 11.03, \lambda_4 = 2.84$ and $f = 53.56$.

Solution using Matlab optimization toolbox:

We will solve the **original** optimization problem, i.e. with the inequality constraints. This is a nonlinear optimization problem since the objective and constraint functions are nonlinear. For nonlinear problems we'll use the `fmincon` function.

$$
\begin{array}{ll}
\text{Maximize} & (12 - p_1)p_1 + (20 - 1.5p_2)p_2 + (28 - 2.5p_3)p_3 \\
 & -3p_1^{1.3} - 5p_2^{1.2} - 6p_3^{1.15} \\
\text{subject to} & p_1 \leq 0.4x_1^{0.9} \\
 & p_2 \leq 0.5x_2^{0.8} \\
 & p_3 \leq 0.6x_3^{0.7} \\
 & x_1 + x_2 + x_3 + R \leq Q \\
 & x_1, x_2, x_3 \geq 0 \\
 & p_1, p_2, p_3 \geq 0
\end{array}
$$

Recall: The `fmincon` function finds the minimum of constrained nonlinear problem of the form:

$$
\begin{array}{ll}
\text{Min} & f(x) \\
\text{s.t.} & c(x) \leq 0 \\
 & c_{eq} = 0 \\
 & Ax \leq b \\
 & A_{eq}x = b_{eq} \\
 & lb \leq x \leq ub
\end{array}
$$

Syntax: `[x,fval] = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)`

Steps to transform equations:

1. Separate bounds, nonlinear equalities, nonlinear inequalities, linear equalities, linear inequalities.
2. Combine all variables into one vector x.
3. Write vectors of lower and upper bounds: `lb` & `ub` such that $lb \leq x \leq ub$.
4. Write matrix and vector of inequalities: `A` & b such that $Ax \leq b$.
5. Write matrix and vector of equalities: $\mathtt{A}_{\mathrm{eq}}$ & b_{eq} such that $A_{eq}x = b_{eq}$.
6. Write the function of the nonlinear constraints that accepts a vector `x` and returns two arrays, `c(x)` and `ceq(x)`, where `c(x)` is the array of nonlinear inequality constraints at `(x` and `ceq(x)` is the array of nonlinear equality constraints at `x`.
7. Write the function of the nonlinear objective that needs to be minimized that accepts a vector `x` and returns a real scalar `f` at `x`.
8. Define initial guess `x0`.
9. Call the solver: `[x,fval] = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)`

Complete the code:

```
% Lagrange multiplier example 3.3 from the book
% solve the optimization problem
% Syntax: [x,fval,exitflag] = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)
clc
clear
% x = [p1 p2 p3 x1 x2 x3]
n = 6;                  % number of variables
RQ = 10;                % available water
% lower bounds


% linear inequality


% nonlinear constraints
nonlcon = @mycon;

% objective function
fun = @(x) -((12 - x(1))*x(1) + (20 - 1.5*x(2))*x(2) + (28 - 2.5*x(3))*x(3) ...
        -(3*x(1)^1.3 + 5*x(2)^1.2 + 6*x(3)^1.15));

x0 = ones(n,1);
options = optimoptions('fmincon','Display','iter');
[x,fval,exitflag, ,lambda] = fmincon( , , , , , , , , ,options)


% constraint function

function [c,ceq] = mycon(x)
    c(1) =
    c(2) =
    c(3) =
    ceq =
end
```

Final solution: $p_1 = 0.46, p_2 = 1.44, p_3 = 1.88, x_1 = 1.16, x_2 = 3.74, x_3 = 5.10$, the Lagrange multipliers are: $\lambda_1 = 8.00, \lambda_2 = 9.24, \lambda_3 = 11.03, \lambda_4 = 2.84$, and the optimal value is $f = 53.47$.