Ladders, Moats, and Lagrange Multipliers

The functions we present here implement the classical method of Lagrange multipliers for solving constrained optimization problems. The moat problem that we employ to motivate the development of these functions generalizes a standard calculus problem, and shows how symbolic algebra systems expand our horizons in the formulation and analysis of geometric problems.

C. Henry Edwards

A Ladder Problem

A standard elementary calculus problem has a ladder leaning across a five-foot fence and just touching a high wall three feet behind the fence. We ask what is the minimum possible length L of the ladder?

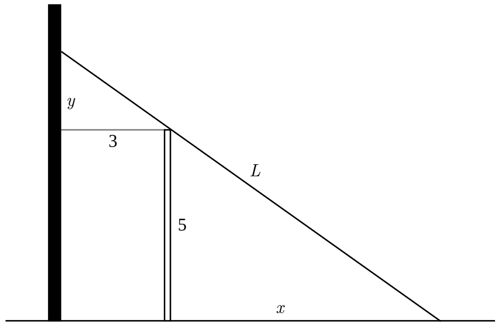


FIGURE 1. A ladder leans against a five-foot fence, just touching a wall that is three feet behind the fence.

The Pythagorean formula applied to the large right triangle in Figure 1 yields the equation

$$(x + 3)^2 + (y + 5)^2 = L^2 \qquad (1)$$

that describes a circle with center $(-3, -5)$ and radius L. The proportionality of base and height for the two smaller right triangles yields $x/5 = 3/y$, that is, the equation

$$x\,y = 15 \qquad (2)$$

of a rectangular hyperbola. Our problem simply asks for the circle (1) of minimum radius L that intersects the first-quadrant branch of the hyperbola (2). From a geometric viewpoint, it is apparent that this circle will be tangent to the hyperbola at their point of intersection (Figure 2).

C. Henry Edwards is a professor of mathematics at The University of Georgia. He is author of The Historical Development of the Calculus (*Springer-Verlag,* 1979) *and co-author of well-known calculus, linear algebra, and differential equations textbooks.*

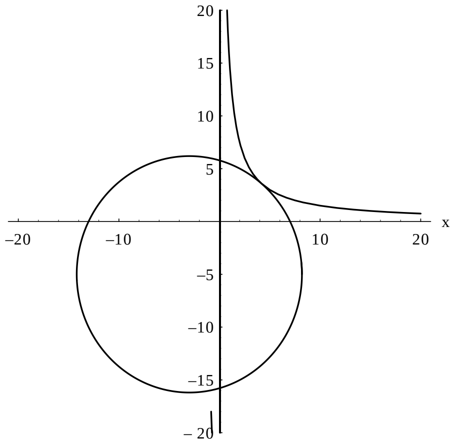


FIGURE 2. The circle (1) tangent to the hyperbola (2).

The elementary calculus approach to this problem would be to substitute $y = 15/x$ from (2) into the righthand-side of (1) to obtain the single-variable function

$$f(x) = (x + 3)^2 + (5 + 15/x)^2$$

that gives the square of the radius of the circle. The Lagrange multiplier approach sidesteps this elimination process by asking for the minimum value of the length-squared *objective function*

$$F(x, y) = (x + 3)^2 + (y + 5)^2$$

subject to the condition that x and y satisfy the *constraint equation*

$$G(x, y) = x\,y - 15 = 0.$$

The fact that the circle and hyperbola are contour curves of the functions F and G, respectively, means that their normal (gradient) vectors ∇F and ∇G must be collinear at the desired point of tangency. Assuming that ∇G is nonzero, it follows that

$$\nabla F = \mu \nabla G \qquad (3)$$

for some constant μ, the *Lagrange multiplier*. The two scalar components of this vector equation, together with the constraint equation, provide a system of three equations to solve for the three unknowns x, y, and μ.

The Moat Problem

The next problem resembles the ladder problem, but poses a greater computational and conceptual challenge. It leads to a constrained optimization example of higher dimension that would be hard to solve by elementary algebraic techniques.

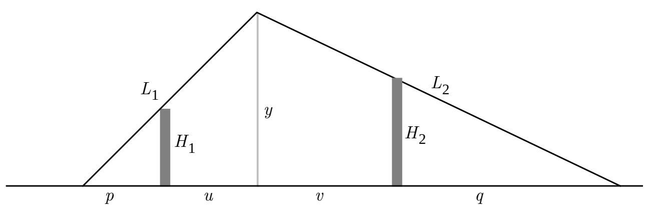


FIGURE 3. A moat and two ladders.

Consider two ladders of lengths L_1 and L_2 leaning across two walls of given heights H_1 and H_2 bordering an alligator-filled moat of given width W (Figure 3). Let's suppose that $H_1 = 10$, $H_2 = 15$, and $W = 50$. The lower ends of the ladders are placed at distances p and q from the walls, so that their upper ends meet at a point (x, y) above the water. The two ladders are supported in place by each other as well as by the two walls. Let u and v denote the horizontal distances of the point (x, y) from the two walls. Geometric intuition suggests the existence of a minimum possible value of the sum $L_1 + L_2$ of the lengths of the two ladders. Our problem is to find this minimal sum.

From Figure 3 we read off the constraint equations:

$$u + v - 50 = 0, \quad (p+u)^2 + y^2 - {L_1}^2 = 0,$$
$$(q+v)^2 + y^2 - {L_2}^2 = 0 \tag{4}$$
$$(y - 10)p - 10u = 0, \quad (y - 15)q - 15v = 0.$$

The first of these equations simply records the given width of the moat. The second and third equations are the Pythagorean relations for the two larger right triangles, while the last two equations follow from proportionality relations for the two pairs of similar triangles in Figure 3.

We see that our problem involves the seven variables p, q, y, u, v, L_1, and L_2, and we seek to minimize the value of the objective function

$$F(p, q, y, u, v, L_1, L_2) = L_1 + L_2$$

subject to the five constraints (4).

The Lagrange Multiplier Approach

The Lagrange multiplier approach involves a natural multivariable generalization of Equation (3). (See, for example, [Edwards 1973, page 113].) Any local extremum of an "object function" $F(x_1, x_2, ..., x_n)$ of n independent variables subject to the $m < n$ constraints $G_1(x_1, x_2, ..., x_n) = 0, ..., G_m(x_1, x_2, ..., x_n) = 0$, must occur at a point $(x_1, ..., x_n)$ where the gradient vector ∇F of the function F is a linear combination of the gradient vectors of the constraint functions $G_1, ..., G_m$ (assuming that all these functions are continuously differentiable and that the gradient vectors of the constraint functions are linearly independent). That is, there exist Lagrange multipliers $\mu_1, \mu_2, ..., \mu_m$ such that the function F and the auxiliary function $H = \mu_1 G_1 + ... + \mu_m G_m$ have the same gradient vector,

$$\nabla F = \nabla H \tag{5}$$

(The gradient of a function of n independent variables is the vector consisting of its partial derivatives with respect to these variables.) If ∇G, the gradient of the vector-valued function $G = (G_1, G_2, ..., G_m)$, denotes the matrix with row vectors $\nabla G_1, \nabla G_2, ..., \nabla G_m$, and $\mu = (\mu_1, ..., \mu_m)$, then ∇H is the matrix product $\mu \nabla G$. Thus,

$$\nabla F = \mu\ \nabla G$$

in perfect analogy with the case $m = 1$ in (3). The m constraint equations, augmented with the n "multiplier equations" that are the scalar components of the vector equation in (5), then constitute a system of $m + n$ equations to solve for the $m + n$ unknowns $x_1, x_2, ..., x_n, \mu_1, \mu_2, ..., \mu_m$.

Here, it will be notationally more convenient to deal with the *expressions* representing functions than with the defined functions themselves. In the moat problem, we want to minimize the value of the functional expression:

```
In[1]:=  F = L1 + L2;
```

in terms of the independent variables:

```
In[2]:=  X = {p, q, y, u, v, L1, L2};
```

(though F depends explicitly only on L_1 and L_2) subject to the condition that each of the following constraint functions vanishes:

```
In[3]:=  G = { u + v - W,
               (p+u)^2 + y^2 - L1^2,
               (q+v)^2 + y^2 - L2^2,
               (y-H1) p - H1 u,
               (y-H2) q - H2 v  };
```

For the given values of H_1, H_2, and W:

```
In[4]:=  W = 50; H1 = 10; H2 = 15;
```

the constraint equations are:

```
In[5]:=  Thread[G == 0]

Out[5]= {-50 + u + v == 0, -L1^2 + (p + u)^2 + y^2 == 0,
          -L2^2 + (q + v)^2 + y^2 == 0, -10 u + p (-10 + y) == 0,
          -15 v + q (-15 + y) == 0}
```

Geometrically, the solution set of these equations is a two-dimensional surface in the seven-dimensional space of the variables.

The multipliers for our problem are given by:

```
In[6]:=  multipliers = Array[mu, Length[G]]
Out[6]=  {mu[1], mu[2], mu[3], mu[4], mu[5]}
```

and the auxiliary function H is defined by:

```
In[7]:=  H = multipliers . G;
```

We can now set up the system of constraint equations:

```
In[8]:=  constraintEquations = Thread[G == 0];
```

and the scalar components of Equation (5):

```
In[9]:=  multiplierEquations = Map[(D[F, #] == D[H, #])&, X];
In[10]:= LagrangeEqs =
           Join[constraintEquations, multiplierEquations]
Out[10]=
                                 2          2    2
         {-50 + u + v == 0, -L1  + (p + u)  + y  == 0,
              2          2    2
           -L2  + (q + v)  + y  == 0, -10 u + p (-10 + y) == 0,
           -15 v + q (-15 + y) == 0,
           0 == 2 (p + u) mu[2] + (-10 + y) mu[4],
           0 == 2 (q + v) mu[3] + (-15 + y) mu[5],
           0 == 2 y mu[2] + 2 y mu[3] + p mu[4] + q mu[5],
           0 == mu[1] + 2 (p + u) mu[2] - 10 mu[4],
           0 == mu[1] + 2 (q + v) mu[3] - 15 mu[5],
           1 == -2 L1 mu[2], 1 == -2 L2 mu[3]}
```

Thus, we get a nonlinear system of 12 equations in the $m + n = 12$ unknowns `{p, q, y, u, v, L1, L2, mu[1], mu[2], mu[3], mu[4], mu[5]}`.

To apply `FindRoot` to solve this problem numerically, we need a plausible initial guess for the values of the variables. In a geometric problem such as the moat problem, we need only look at a figure to gauge the approximate relative values of the geometric variables. For instance, in Figure 3, the approximate values:

```
In[11]:= X0 = {15, 15, 25, 25, 25, 60, 60};
In[12]:= initialValues = Thread[X -> X0]
Out[12]= {p -> 15, q -> 15, y -> 25, u -> 25, v -> 25, L1 -> 60,
           L2 -> 60}
```

look fairly reasonable. We must work a bit harder to find initial estimates for the multipliers. If `X0` were actually a solution, the multipliers would satisfy the following equations:

```
In[13]:= multiplierEquations /. initialValues // TableForm
Out[13]//TableForm=
      0 == 80 mu[2] + 15 mu[4]
      0 == 80 mu[3] + 10 mu[5]
      0 == 50 mu[2] + 50 mu[3] + 15 mu[4] + 15 mu[5]
      0 == mu[1] + 80 mu[2] - 10 mu[4]
      0 == mu[1] + 80 mu[3] - 15 mu[5]
      1 == -120 mu[2]
      1 == -120 mu[3]
```

This is an over-determined linear system of n equations in $m < n$ unknowns, so we "solve" by the method of least squares, thereby obtaining a "solution" that minimizes the sum of the squares of the discrepancies in the n equations. The $n \times m$ coefficient matrix is the transpose of the matrix ∇G at the given initial values:

```
In[14]:= AT = Outer[D[#1, #2]&, G, X] /. initialValues;
In[15]:= (A = Transpose[AT]) // MatrixForm
Out[15]//MatrixForm=
      0     80     0      15     0
      0     0      80     0      10
      0     50     50     15     15
      1     80     0      -10    0
      1     0      80     0      -15
      0     -120   0      0      0
      0     0      -120   0      0
```

The vector of constants is

```
In[16]:= b = Map[D[F, #]&, X] /. initialValues
Out[16]= {0, 0, 0, 0, 0, 1, 1}
```

The hypothesis that the vectors ∇G_1, ∇G_2, ... , ∇G_m are linearly independent implies that the $m \times m$ matrix $\boldsymbol{A}^T\boldsymbol{A}$ is nonsingular, so we can solve the least squares system $\boldsymbol{A}^T\boldsymbol{A}\ \boldsymbol{x} = \boldsymbol{A}^T\ \boldsymbol{b}$ to estimate the multipliers (see, for example, [Edwards and Penney 1988, page 226]):

```
In[17]:= mu0 = LinearSolve[AT . A, AT . b] // N
Out[17]= {1.00303, -0.00816004, -0.00723893, 0.0330785, 0.0295572}
```

Now we can assemble our initial guesses into a list of the form needed by `FindRoot`.

```
In[18]:= XPairs = Transpose[{X, X0}];
In[19]:= muPairs = Transpose[{multipliers, mu0}];
In[20]:= allPairs = Join[XPairs, muPairs]
Out[20]= {{p, 15}, {q, 15}, {y, 25}, {u, 25}, {v, 25}, {L1, 60},
           {L2, 60}, {mu[1], 1.00303}, {mu[2], -0.00816004},
           {mu[3], -0.00723893}, {mu[4], 0.0330785},
           {mu[5], 0.0295572}}
```

Finally, we have our complete system of equations and initial guesses:

```
In[21]:= LagrangeData = Join[{LagrangeEqs}, XPairs, muPairs];
```

as needed to proceed with a numerical solution:

```
In[22]:= OptimalValues = Apply[FindRoot, LagrangeData];
```

The minimum value obtained for F, the sum of the lengths of the two ladders, is given by:

```
In[23]:= F /. OptimalValues
Out[23]= 102.715
```

The optimal values for the variables are:

```
In[24]:= Take[OptimalValues, Length[X]]
Out[24]= {p -> 17.2361, q -> 13.118, y -> 30.9275, u -> 36.0708,
           v -> 13.9292, L1 -> 61.629, L2 -> 41.086}
```

The functions LagrangeEquations and LagrangeSolve assemble the steps carried out above. Given an objective function F, a list X of independent variables, a list G of constraints, and a symbol (such as mu) for the multiplier, LagrangeEquations returns a list of the constraint and multiplier equations:

```
In[25]:= LagrangeEquations[F_, X_ ,G_, mu_Symbol:mu] :=
         Module[{H = Array[mu, Length[G]] . G},
           Join[ Thread[G == 0],
             Map[(D[F, #] == D[H, #])&, X ] ] ]
```

The function LagrangeSolve computes the optimal values of the objective function and the independent variables, given F, X, G, and an initial guess $X0$ for the list X:

```
In[26]:= LagrangeSolve[F_, X_, X0_, G_] :=
         Module[{
           AT = Outer[D[#1, #2]&, G, X] /. Thread[X -> X0],
           b = Map[D[F, #]&, X] /. Thread[X -> X0],
           mu, mu0, optimalValues},
           mu0 = N[LinearSolve[AT . Transpose[AT], AT . b]];
           optimalValues =
             Apply[FindRoot,
               Join[
                 {LagrangeEquations[F, X, G, mu]},
                 Transpose[{X, X0}],
                 Transpose[{Array[mu, Length[G]], mu0}]]];
           {F /. optimalValues,
               Take[optimalValues, Length[X]]} ]
```

For example, with F, X, $X0$, and G defined as above for the moat problem we get:

```
In[27]:= LagrangeSolve[F, X, X0, G]
Out[27]= {102.715, {p -> 17.2361, q -> 13.118, y -> 30.9275,
           u -> 36.0708, v -> 13.9292, L1 -> 61.629, L2 -> 41.086}}
```

For the original ladder problem with:

```
In[28]:= F = (x + 3)^2 + (y + 5)^2;
         X =  {x, y};
         X0 = {5, 3};
         G  = {x y - 15};
```

we get:

```
In[32]:= LagrangeSolve[F, X, X0, G]
Out[32]= {125.308, {x -> 4.21716, y -> 3.55689}}
```

Thus, the shortest ladder that will suffice in Figure 1 has length

```
In[33]:= Sqrt[F /. Last[%]]
Out[33]= 11.1941
```

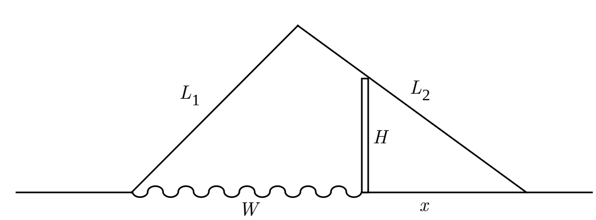
Problems

There should always be homework! The following five problems range from the original simple ladder situation to a more complicated moat situation.

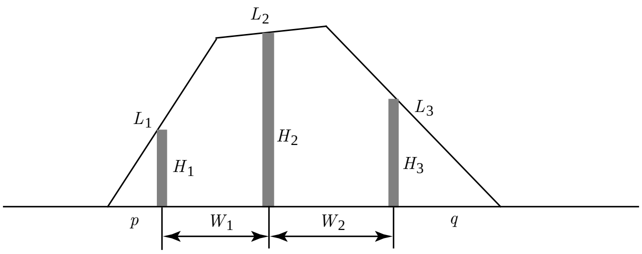
Problem 1. Use LagrangeMultiplier to solve the ladder problem again, except now using objective function $F = L$ and two constraints corresponding to Equations (1) and (2).

Problem 2. Observe that, if $H_1 = H_2$, then our moat problem simply amounts to a ladder problem and its mirror image in an imaginary wall bisecting the moat. Confirm that with $W = 50$ and $H_1 = H_2 = 15$ (say), the solution of this symmetric moat problem agrees with the solution of the corresponding ladder problem.

Problem 3. The figure below shows a moat of width $W = 50$ having a wall of height $H = 15$ feet on only one side. Given $L_1 = 40$, find the minimum possible length L_2 of the second ladder.



Problem 4. In the original moat problem, suppose that the length $L_1 = 40$ of the first ladder is given in advance. Then find the minimal length L_2 of the second ladder such that the two ladders will jointly suffice to span the moat in the indicated manner.



Problem 5. The figure above illustrates a two-moat problem. Two adjoining moats of given widths W_1 and W_2 are bounded by walls of given heights H_1, H_2, and H_3. This pair of moats-with-walls is to be spanned by three ladders as indicated in the figure. With an appropriate choice of the five given dimensions, use Lagrange multipliers—perhaps with 11 independent variables and eight constraints, so that a system of 19 equations must ultimately be solved—to find the minimum possible sum $L_1 + L_2 + L_3$ of their lengths.

A Programming Problem. The Lagrange multiplier condition expressed in Equation (5) is merely a first-derivative *necessary* condition for constrained local extrema. If the initial guess is not sufficiently close to a desired extremum, the sequence of Newton's method iterates produced by `FindRoot` may either diverge or converge to a critical point other than the one anticipated. Implement the second-derivative *sufficient* conditions (in terms of bordered Hessians) given in [Spring 1985], which may be used to distinguish between constrained maxima/minima and saddle points.

References

Edwards, C. H. 1973. *Advanced Calculus of Several Variables*. Academic Press, N.Y. (1994 reprint forthcoming from Dover Publications, N.Y.)

Edwards, C. H. and David E. Penney 1988. *Elementary Linear Algebra*. Prentice Hall, Englewood Cliffs, New Jersey.

Spring, D. 1985. On the second derivative test for constrained local extrema. *American Mathematical Monthly* 92:631–643.

C. Henry Edwards
Department of Mathematics, University of Georgia,
Athens, GA 30602
`hedwards@joe.math.uga.edu`

The electronic supplement contains the notebook `Ladders` and the package `LagrangeMultiplier.m`.