

Lecture XVI: Finding the Minimum Using Newton's Method

Charles B. Moss

September 27, 2012

1 Univariate Newtons's Method

As a starting point consider the problem of finding the maximum of a quadratic utility function

$$U(x) = 5x^2 - 4x + 2 \quad (1)$$

which has a minimum determined by setting its first derivative equal to zero

$$\begin{aligned} \frac{\partial U(x)}{\partial x} &= 10x - 4 = 0 \\ x &= \frac{2}{5} \end{aligned} \quad (2)$$

In addition, straightforward transformations such as

$$U(x) = e^{5x^2 - 4x + 2} \quad (3)$$

offer little additional complexity

$$\begin{aligned} \frac{\partial U(x)}{\partial x} &= U(x) [10x - 4] = 0 \\ x &= \frac{2}{5} \end{aligned} \quad (4)$$

Thus, the functions presented in Equations 1 and 3 are not complex enough to be interesting functions to discuss algorithms. Another possibility is a rational function

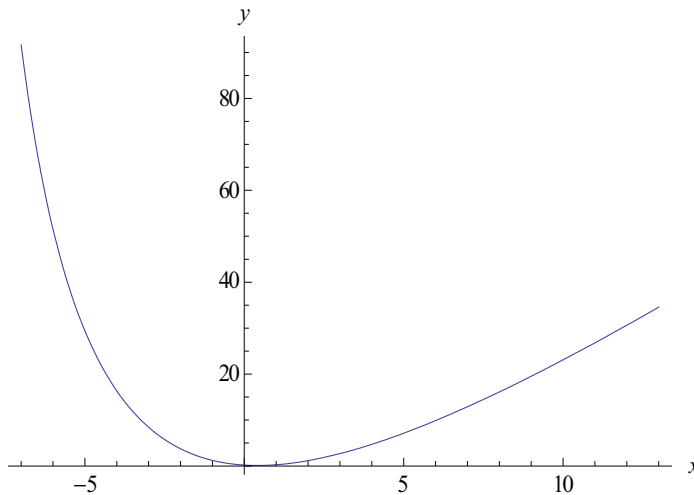


Figure 1: Graph of Rational Function

$$f(x) = \frac{5x^2 - 4x + 2}{x + 10} \quad (5)$$

which is depicted in Figure 1. However, these functions typically have bizarre discontinuities as depicted in a plot of a broader range for the same function in Equation 1. This plot shows the vertical asymptote at $x = -10$.

If we restrict our attention to the range of x 's such that x is greater than -10 the problem becomes more tractable. Taking the first-order condition

$$\begin{aligned} \frac{\partial f(x)}{\partial x} &= \frac{10x - 4}{10 - x} - \frac{5x^2 - 4x + 2}{(x + 10)^2} = 0 \\ x &= \frac{-100 + 2\sqrt{2710}}{10} \\ x &= 0.411532 \end{aligned} \quad (6)$$

Most of the numeric optimization procedures involve numerically finding the zero of the derivative. The gradient of the rational over the range $x \in (-10, \infty]$ is presented in Figure 3.

1.1 Method of Bisection

The method of bisection involves shrinking the interval. Specifically, suppose that the original interval was $[-8, 20]$. Evaluating the function at each point

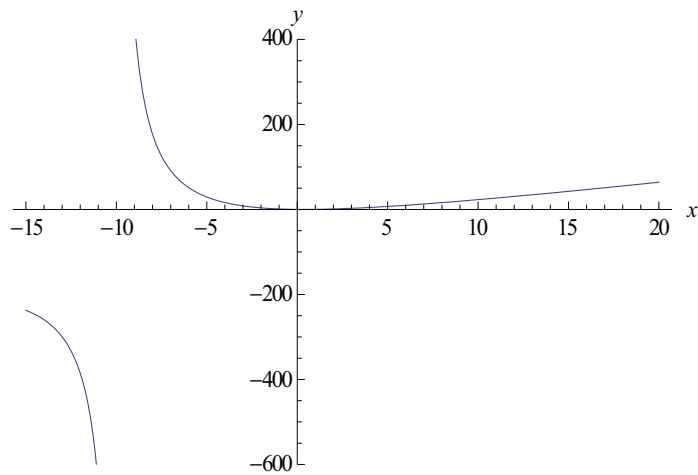


Figure 2: Extended Graph of Rational Function

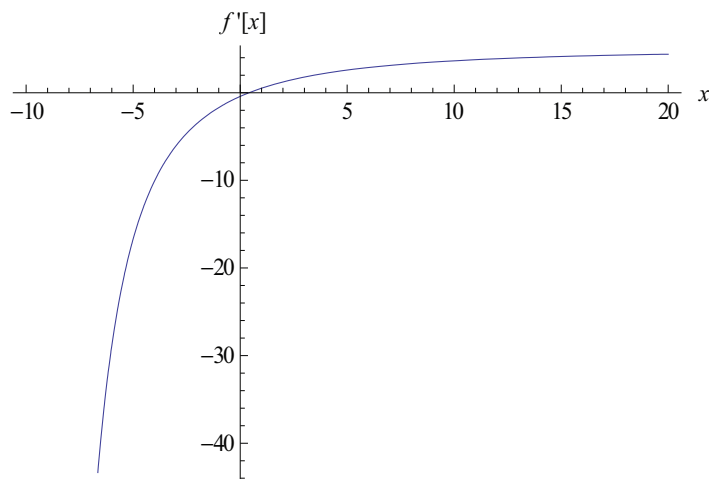


Figure 3: Derivative of Rational Function

$f[-8] = 16.111$ and $f[20] = 64.067$. The midpoint of the interval is $x = 6$. At $x = 6$, the value of the derivative is $f[6] = 2.88281$. Therefore, we throw away to higher bisection yielding an interval of $[-8, 6]$ (we throw away the bisection with the same sign on the derivative). Next, taking the midpoint of the new interval is -1 with a derivative value of -1.69136. At this iteration, we throw out the lower bisection yielding an interval of $[-1, 6]$. Table 1 presents a complete sequence of bisections.

1.2 Newton's Method

The theoretical foundation of Newton's method involves inscribing triangles inside the function. Mathematically

$$\begin{aligned}\Delta y &\approx \frac{\partial f(x)}{\partial x} \Delta x \\ \Delta x &= \frac{\Delta y}{\frac{\partial f(x)}{\partial x}} \\ x_{t+1} &= x_t - \frac{f(x)}{\frac{\partial f(x)}{\partial x}}\end{aligned}\tag{7}$$

This Newton step is depicted graphically in Figure 4. Table 2 presents the steps for the Newton search algorithm for the derivative of the rational function presented in Equation 6.

2 Finding the Multivariate Maximum

The basic difference between univariate and multivariate optimization is the number of equations which we want to solve simultaneously for zero. In the multivariate case we want to solve

$$\nabla_x f(x) = 0\tag{8}$$

where x is an n element vector, so we want to solve for n equations equal to zero. Appealing again to the second order Taylor series expansion

Table 1: Method of Bisection

Iteration	Lower		Upper		Bisection		Replace
	Value	Derivative	Value	Derivative	Value	Derivative	
0	-8.0000	-130.5000	20.0000	4.3978	6.0000	2.8828	Upper
1	-8.0000	-130.5000	6.0000	2.8828	-1.0000	-1.6914	Lower
2	-1.0000	-1.6914	6.0000	2.8828	2.5000	1.5312	Upper
3	-1.0000	-1.6914	2.5000	1.5312	0.7500	0.3099	Upper
4	-1.0000	-1.6914	0.7500	0.3099	-0.1250	-0.5581	Lower
5	-0.1250	-0.5581	0.7500	0.3099	0.3125	-0.0965	Lower
6	0.3125	-0.0965	0.7500	0.3099	0.5313	0.1130	Upper
7	0.3125	-0.0965	0.5313	0.1130	0.4219	0.0099	Upper
8	0.3125	-0.0965	0.4219	0.0099	0.3672	-0.0429	Lower
9	0.3672	-0.0429	0.4219	0.0099	0.3945	-0.0164	Lower
10	0.3945	-0.0164	0.4219	0.0099	0.4082	-0.0032	Lower
11	0.4082	-0.0032	0.4219	0.0099	0.4150	0.0034	Upper
12	0.4082	-0.0032	0.4150	0.0034	0.4116	0.0001	Upper
13	0.4082	-0.0032	0.4116	0.0001	0.4099	-0.0016	Lower
14	0.4099	-0.0016	0.4116	0.0001	0.4108	-0.0007	Lower
15	0.4108	-0.0007	0.4116	0.0001	0.4112	-0.0003	Lower

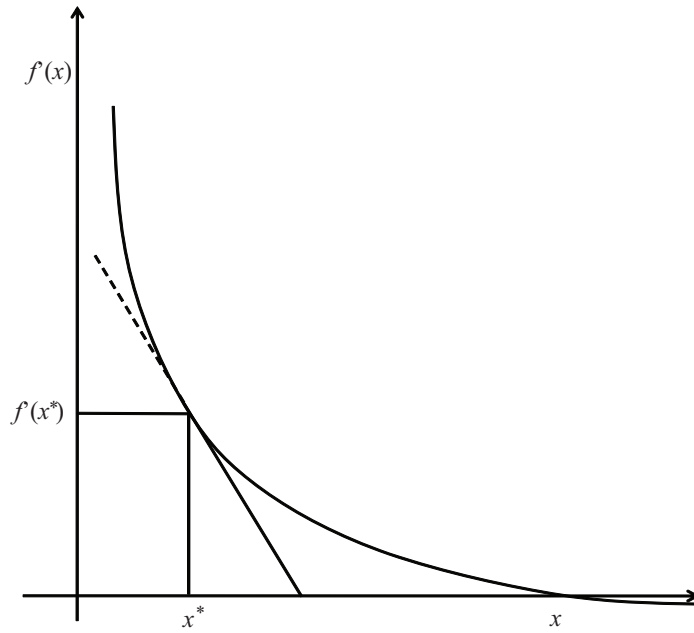


Figure 4: Newton Step

Table 2: Newton Search for Rational Function

Search Point	Function Value	Derivative	Step
-8.0000	-130.5000	135.5000	0.9631
-7.0369	-56.7315	41.6668	1.3616
-5.6754	-23.9799	13.4022	1.7893
-3.8861	-9.4998	4.7432	2.0028
-1.8833	-3.2269	2.0272	1.5919
-0.2914	-0.7503	1.1846	0.6334
0.3419	-0.0675	0.9800	0.0689
0.4108	-0.0007	0.9607	0.0007
0.4115	0.0000	0.9605	0.0000

$$\begin{aligned}\nabla_x f(x) &= \nabla_x f(x^*) + \nabla_{xx}^2 f(x^*)(x - x^*) = 0 \\ \Rightarrow (x - x^*) &= \left(\nabla_{xx}^2 f(x^*)\right)^{-1} \nabla_x f(x)\end{aligned}\tag{9}$$

In this case, we take a linear approximation of the gradient function (i.e., the $n \times 1$ vector of derivatives $\nabla_x f(x)$). This approximation following the first-order Taylor approximation is a function of the original gradient function at the point of approximation ($\nabla_x f(x^*)$) and the derivative (matrix) of the gradient (i.e., the hessian matrix of the objective function $\nabla_{xx}^2 f(x^*)$). Substituting x_{t+1} for x and x_t for x^* we are left with an estimate of the value of x for the next iteration such that the value of the derivative is equal to zero

$$x_{t+1} = x_t - \left(\nabla_{xx}^2 f(x_t)\right)^{-1} \nabla_x f(x_t)\tag{10}$$

2.1 A Simple Example

As a first problem, consider a Cobb-Douglas utility function with a budget constraint imposed

$$\begin{aligned}\max_x & x_1^{0.2} x_2^{0.3} x_3^{0.4} x_4^{0.1} \\ \text{s.t. } & x_1 + 2x_2 + 3x_3 + x_4 = 100\end{aligned}\tag{11}$$

By substituting the constraint for Equation 11 into its objective function the unconstrained maximization problem becomes

$$\max_x (100 - 2x_2 - 3x_3 - x_4)^{0.2} x_2^{0.3} x_3^{0.4} x_4^{0.1}\tag{12}$$

Starting with the point $x \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$ (note that we now only have three variables because the value of x_1 is determined by the constraint $x_1 = 100 - 2x_2 - 3x_3 - x_4$)

$$\nabla_x f(x) = \begin{pmatrix} 0.7337 & 0.9766 & 0.2428 \end{pmatrix}$$

$$\nabla_{xx}^2 f(x) \begin{pmatrix} -0.5275 & 0.2885 & 0.0717 \\ 0.2885 & -0.6085 & 0.0954 \\ 0.0717 & 0.0954 & -0.2244 \end{pmatrix}$$

$$x_{t+1} = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} - \nabla_x f(x_t) \left(\nabla_{xx}^2 f(x_t) \right)^{-1} = \begin{pmatrix} 5.3559 & 5.3479 & 5.3226 \end{pmatrix} \quad (13)$$

Need to follow through with the numbers (i.e., the point that the new solution is better than the old solution and that iteratively applying the procedure will converge to an optimum)

2.2 Prototype Algorithm

The idea that the rules defined by the Newton-Raphson approach above applied multiple times yields a value the x_t vector which is arbitrarily close to the optimal point defines an Algorithm. More specifically, an algorithm is a set of rules that can be mechanically applied to yield an approximation of the optimal value of the function (either maximum or minimum). Following the example of Gill, Murray and Wright we will define a prototype algorithm for solving the unconstrained optimization algorithm [U] based on the Newton Raphson process. In this definition we assume that we start with a current estimate of x^* (x_t) and attempt to find a point that minimizes the objective function $f(x)$

- U1 *Test for convergence* If the current point (x_t) satisfies the condition for convergence the algorithm terminates with x_t . In the case of the unconstrained problem or convergence condition is $|\nabla_x f(x_t)| = 0$. If x_t satisfies this condition $f(x_t)$ is a local minimum.
- U2 *Compute a search direction* If the current point is not optimum then what direct do we want to move to attempt to find an optimum. In an univariate application the search direction is either to the right (a higher value of x) or to the left (a lower value of x) on the number line. In the multivariate case we can proceed in an infinite number of directions (i.e., in the two variable case an infinite number of x_1/x_2).

The direction of the search is a $n \times 1$ projection vector p_t which defines the direction of the search. In the Newton-Raphson approach $\nabla_x f(x_t) (\nabla_{xx}^2 f(x_t))^{-1}$ defines the direction of the search.

U3 *Compute a step length* Given that we know the direction of the search the next step is to compute how far to step a_t . In Equation 13 our implicit step length was 1. However, if the function is concave so that the derivative is decreasing at a decreasing rate we know that the zero implies a larger step than the linear approximation. The question is how much farther could we go? An appropriately selected step length reduces the number of iterations required for minimization. However, a step length that is too large could potentially increase the number of iterations required. At this stage, it is sufficient to state that the step length is chosen to reduce the value of the objective function $f(x_t + a_t p_t) < f(x_t)$.

U4 *Update the estimate of the minimum* Set $x_{t+1} = x_t + a_t p_t$ and go back to step U1

Given the steps to the prototype algorithm, I want to develop a sample problem that we can compare the various algorithms against. Notebook Algorithm3.ma includes the numeric problem from Equation 11. The desktop CPU solves this sequence (not optimally specified in Mathematica) in 8 iterations taking 0.374 CPU seconds.