Michael Carreno
mcarreno@cs.princeton.edu
messed up by Ken Steiglitz
ken@cs.princeton.edu

Rootfinding and Optimization in Economics

Background: annuity interest rate, zero-finding

We'll start with a simple economic application of rootfinding, the problem of determining the interest rate of an annuity when other characteristics of the annuity are specified. Annuities involve compound interest and this is what makes the calculation of the interest rate nontrivial.[1]

Suppose you want to set up an individual retirement account (IRA). You must consider how many years you will deposit money into the account, the amount of each deposit, how many years you will draw from the account, and the amount that will be withdrawn each year (or some other chosen time period). We're interested here in the problem you face if you decide on all these factors, and then must determine what interest rate is necessary so that the last debit will bring the account down to a balance of precisely zero after the last withdrawal. For an example of this problem, suppose you want to deposit \$2200 a year for 30 years, and then withdraw \$30,000 a year for the next 10 years, when you retire. (We'll ignore taxes.) What interest rate do you need?

The following equation represents this situation [2, 3]:[2]

$$P_{in}[(1+r)^{N_{in}} - 1] - P_{out}[1 - (1+r)^{-N_{out}}] = 0 \tag{1}$$

where

- P_{in} = amount deposited for a set number of time periods
- N_{in} = number of deposits
- P_{out} = amount withdrawn from the account for a set number of time periods
- N_{out} = number of withdrawals
- r = interest rate received for each time period

We also assume that the withdrawals begin exactly one time period after the last deposit. In general, you might consider the problem where we specify all the parameters except one and ask what the remaining parameter should be. Actually, this is a straightforward problem

[1] Compound interest was once described by Albert Einstein as the "greatest mathematical discovery of all time" (quoted in [1]).

[2] If you like algebra, you might enjoy deriving this equation. Hint: get out your high school algebra book and use the formula for the sum of a finite geometric series. How is the formula changed if we wait an additional N_{wait} periods after the last deposit before withdrawing?

in all cases except when r is the unknown. In this case we need to use an iterative rootfinding method.[3]

Background: differential pricing in monopolies, optimization

An application of iterative optimization in economics arises in a perfect or nearly perfect monopolistic situation. In a *perfectly competitive* market, an individual producer is faced with a demand curve and market price that is unaffected by her production decisions. A producer's decision as to how much to produce is determined by input cost and factors of scale. But the price she charges and gets is determined by the market. If she charges above this price, her products will not be bought and she will soon be driven out of business. A producer is also unable to charge below the market price because in such a competitive market, the market price is the lowest one that leads to an acceptable profit. All the producers have optimized their operations as best they can to achieve a low price. The only way a new producer can sell at a lower price is to discover some new manufacturing or management technique which improves the production process.

Monopolists face a different process though: the price of their product in a market is determined by their output. For a single market the monopolist must simply determine the volume of output that maximizes revenues. In order to do this the monopolist considers the demand curve of the market: the amount that will be purchased as a function of price. From this curve the monopolist can determine how much revenue she will receive for a certain volume of output. If the monopolist is selling her goods in more than one market and can produce an unlimited volume of output, the situation remains the same. The volume of goods to be sent to each market is determined simply by the demand curve for that market.

In contrast, when a monopolist can sell in more then one market and, for one reason or another, can produce only a limited volume, her situation becomes more interesting. She is then faced with the problem of determining the optimal output for each market, in such a way as to maximize total revenues, but restricted by the constraint that her output be no more than a given, known volume. This is an example of a *constrained* optimization problem: the monopolist must maximize a known function representing revenues — subject to a known constraint of her total production volume.

Background: constraints, zero-finding, and optimization

Note that there are constraints in both the annuity rate problem above and the monopolist's pricing problem that we haven't mentioned — namely that interest rates and prices must always be nonnegative. In general, constrained problems are more difficult than unconstrained problems and it's not difficult to see why. Iterative algorithms go their own way, and if and when they bump into walls, we have to decide what to do from there. In practice it's common to begin by ignoring constraints and hoping for the best. If the unconstrained solution obeys the constraints, we're done. If not, it's back to the drawing board. For example, if you use an unconstrained root-finding method (like Newton's method) for the annuity rate problem and you get a positive interest rate, you're done. Just start your payments. If you get a negative rate, then you have to think more. It may be that there is no feasible solution to your problem. (Can this happen in this particular example?) Or it may be that you have made a mistake somewhere along the line.

If you use an unconstrained iterative algorithm in an optimization problem and get an infeasible solution, there is a third possibility, besides the possibilities of there being no

[3] Unless you can find the solution analytically. Do you think that's possible in this case?

solution at all, or an error. There may be a solution that is the best among all the ones that satisfy the constraints, but different from the one you've found. This case requires another level of sophistication in the development of efficient and robust algorithms, so-called *constrained optimization algorithms*.

In some cases an optimization problem can be reduced to a zero-finding problem. If you have some analytical expression for the function to be optimized (called the *objective* or *cost function*), you can look for zeros of its derivative. But again, you need to worry about any constraints that need to be respected by solutions.

Sometimes it's also the case that a constraint can be used to eliminate one of the variables in an optimization problem. This is (happily) the case in our monopolist's problem. (Always, though, you need to check the feasibility of your answer.) For example, if we formulate a problem with three unknowns, x, y, and z, and we add the constraint that $x + y + z = C$, a constant, then we can solve for one of the variables, say $z = C - x - y$, substitute for z in the objective function, and solve for the *two* unknowns x and y. (At the risk of being monotonous, this doesn't release you from the responsibility of checking the value of z you get to see if it's feasible.) In lucky situations like this when the result is feasible, you can at one stroke reduce the number of variables, a big win, and eliminate a constraint.

Your assignment, basic part

Part 1:

Take the example of the annuities given above and find the required interest rates for the three situations described by the data below:

Situation 1:	$P_{in} = \$2200$	$P_{out} = \$30,000$	$N_{in} = 30$	$N_{out} = 10$
Situation 2:	$P_{in} = \$2200$	$P_{out} = \$10,000$	$N_{in} = 40$	$N_{out} = 40$
Situation 3:	$P_{in} = \$2200$	$P_{out} = \$300,000$	$N_{in} = 40$	$N_{out} = 10$

Try at least bisection and Newton's Method. The mathematics behind these two algorithms is described in Numerical Recipes in C [4] and we'll certainly discuss them in class. Observe the rates at which these algorithms converge. Try different starting points and note the variation of running time. Does Newton's method ever fail to converge?

I can't and won't try to stop you from using the code in [4]. But what I want you to do, and what will be easier and more instructive, will be to look over their code and write your own. Don't try to make it bulletproof. And by all means, start with a very simple case and work up to more complicated examples. (I always start programming by having my program read some data and say hello. I always have a version that compiles and runs correctly before I go on.) This goes for all the assignments this term. As I mention in the information page for the course, I haven't installed the disk that comes with the book, and I've never used it. You may want to use it later in life, when you've coded some of the simpler algorithms yourself.

Another thing to consider in this problem and those that follow is the plausibility of your answers, or any other numbers you generate. Be sure to ask yourself continually whether the numbers you're getting make sense in the given situation.

Part 2:

Consider next the case when a monopolist is faced with two separate markets and a constraint on total production volume. As discussed above, if we use volumes in each market as the variables, we can reduce the problem to optimization in one variable. If, further, we know the derivative of the resulting objective function in analytical form, the problem becomes very similar to the annuity rate problem, finding the zero of a known function.

In this part of your assignment, a drug company has developed a treatment for people infected with the HIV virus. Due to a shortage of production equipment and the company's desire to monitor carefully the people given the drug, production has been limited to 10,000 treatments. These treatments can be administered only at the company's two laboratories, which are located in the U.S. and Frenetia.

The company must decide how to set volumes and prices in the two markets to maximize revenue, given the limitation on total output.

Shown below are functions for the the demand curves representing the U.S. and Frenetia, respectively:

$$D_1 = (1.1 \times 10^4)(e^{-1.8\times10^{-5} P_1}) \tag{2}$$

$$D_2 = (9.0 \times 10^3)(e^{-2.1\times10^{-5} P_2}) \tag{3}$$

The objective function is the total revenue, which is

$$f(D_1, D_2) = P_1 * D_1 + P_2 * D_2 \tag{4}$$

given that D_1 units are sold in the U.S. at a corresponding price of P_1, and similarly for Frenetia.

Determine the optimal number of treatments the company should offer in each market. Also find the corresponding prices and the resulting revenue. Again, try at least bisection and Newton's method. As above, compare the rate at which these algorithms converge, and investigate the sensitivity to starting point.

Notice that I couched the problem in terms of the unknowns D_1 and D_2, but you can just as well use the corresponding prices P_1 and P_2 as the unknowns. For this to work, it is important that there be only one price for a given demand in each market, which is the case in this example because of the monotonically decreasing shape of the demand curves. (Is it reasonable to assume that the demand curves are always uniquely invertible?)

There is a real practical advantage to using demands as unknowns, though. The constraint then takes the form

$$D_1 + D_2 = 10000 \tag{5}$$

which simplifies the algebra when you eliminate one of the variables. I'll let you finish the job of reducing the problem to one of minimizing a function of one variable.

After you solve the case above, experiment with different constants A_1, A_2, B_1, and B_2. Can you think of cases where you can predict your answer, and therefore test your program? (These are often a big help in debugging.) Can you predict which way the optimal volumes and prices move when you vary the relative demands in the two markets? (This kind of experimentation is a valuable way to gain confidence in your programs.)

Part 3:

The problem for the monopolist gets harder when there are more than two markets. For three markets and a constraint on total volume, for example, her problem becomes one of minimizing a function of two variables instead of one. That might seem like it's only twice as hard in some sense, but it can be, depending on how nasty the shapes of the demand curves are, a lot harder than that.

We'll consider the problem of pricing theater seats. A theater operator is free to charge different prices to different segments of the population, and in that sense has a kind of monopoly because no one else is selling seats to her particular show. Suppose we decide on three categories: townies, alumni, and students, and let the respective demand curves be

$$D_1 = 750e^{-0.55P_1} \tag{6}$$

$$D_2 = 1000e^{-0.65P_2} \tag{7}$$

$$D_3 = 1100e^{-1.05P_3} \tag{8}$$

Suppose also that there are altogether 1100 seats in the theater. How many seats should be offered in each category at what prices, and what is the total resulting revenue expected?

Reduce the problem to one with two unknown variables. To solve it, start with the fixed-step gradient algorithm. That is, make each step proportional to the gradient. If $\mathbf{x_i}$ is the n-dimensional variable at step i, and we are trying to *maximize* $f(\mathbf{x})$, this means the iteration step is

$$\mathbf{x}_{i+1} = \mathbf{x}_i + s\nabla f(\mathbf{x_i}) \tag{9}$$

where s is a fixed step size. This is literally a "hill-climbing" algorithm, going uphill locally a distance that is a fixed multiple of the derivative in the that direction.

Implement the fixed-step algorithm and try it on the monopolist's three-market (two variable) example. Experiment with different values of s. What happens when s is too large? Too small?

The next level of sophistication in gradient algorithms is the "optimum-step-size" gradient searches for the maximum of $f(\mathbf{x})$ in the gradient direction. That is, at each step in the iteration, the right-hand side of (9) is maximized by searching of values of s. Implement the optimum step-size gradient algorithm by using bisection search for s and try it on the monopolist example with three markets. (You should be able to lift the code you wrote for the first part of this assignment. But you need to figure out an effective way to bracket the answer along the gradient direction before you begin chopping down the range.) How does its performance compare with the fixed-step method?

Finally, implement Newton-Raphson on the three-market monopolists problem, and compare with the fixed- and optimum-step-size methods. Which is the fastest and which is the most reliable for this problem?

If you have time, it's interesting to plot the trajectories followed by the different algorithms in the x-y plane.

Extra Credit

1. Try the following data for the theater owner's problem:

$$D_1 = 400e^{-2.1P_1} \tag{10}$$

$$D_2 = 700e^{-0.65P_2} \tag{11}$$

$$D_3 = 500e^{-0.9P_3} \tag{12}$$

$$\text{Seats} = 1100 \tag{13}$$

What goes wrong? What can you do about it?

2. You used iterative optimization techniques in this assignment. But were they necessary? Try to solve the problems analytically. Can you conclude that analytical solutions are impossible? Unlikely? You might want to use Mathematica or Maple [5, 6, 7, 8]. Can you find other demand functions which lead to optimization problems that can be solved analytically?

3. Can you find demand functions for either one- or two-variable problems like the ones in this assignment that lead to objective functions with more than one local minimum? I suggest you restrict the demand functions to be monotonically decreasing, so there's only one price that corresponds to each demand.

4. In the early days of work on nonlinear optimization algorithms, some bizarre functions were devised to test algorithms. Rosenbrock's banana-shaped valley [9] is the most famous. The problem is to minimize the following function of two variables:

$$f(x, y) = 100(y - x^2)^2 + (1 - x)^2 \tag{14}$$

If you're used to thinking in terms of maximization, just multiply by -1.

- Why can this function called a banana-shaped valley?
- Try your Newton-Raphson algorithm on this for various starting points. Does it work reliably? Efficiently?
- For a more advanced project, experiment with other methods, possibly some of your own invention.

Fletcher and Powell [10] proposed the following even trickier three-dimensional test function:

$$f(x, y, z) = 100\left[(z - 10\theta)^2 + (r - 1)^2\right] + z^2 \tag{15}$$

where

$$x = r\cos(2\pi\theta) \tag{16}$$

$$y = r\sin(2\pi\theta) \tag{17}$$

That is, r and θ are polar coordinates in the x–y plane. Warning: be careful to choose the right quadrant if you use the arctan function to get θ. This is a frequent source of headaches.

- Why can this be called a helical valley?
- Repeat the experiments you did for the curved banana valley.

Aoki's book [11] uses a fair amount of math (vector algebra and calculus) and is a bit dated, but is a clear and compact exposition of basic nonlinear optimization algorithms. He also gives other test functions, and discusses comparison of methods.

Scales [12] is a bit less mathematical, a bit more up-to-date, and has some illuminating figures of examples as well as good pseudocode. If you enjoy numerical pathologies, Scales includes, besides the Rosenbrock and Fletcher-Powell functions mentioned above, two more wicked beasts: a four-dimensional function (called "Wood's function" without further identification) that has a "near saddle point and a variety of possible paths to the minimum," and finally a singular function of four variables due to M. J. D. Powell.

Dennis and Schnabel [13] (referenced in *Numerical Recipes* [4]) is more mathematical than Aoki, and doesn't deal with constrained minimization problems.

5. See the projects page for an application to option pricing.

Acknowledgement

Thanks to Professor Tim Van Zandt of the Economics Department for suggesting the monopolist example.

References

[1] B. G. Malkiel. *A Random Walk Down Wall Street*. W. W. Norton and Company, New York, 6th edition, 1996.

[2] K. Atkinson. *Elementary Numerical Analysis*. John Wiley, New York, 1985.

[3] C. Stickney and R. Weil. *Financial Acounting*. Dryden Press, New York, 8th edition, 1997.

[4] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: the Art of Scientific Computing*. Cambridge University Press, Cambridge, U.K., 2nd edition, 1992.

[5] J. Robertson. *Engineering Mathematics with Maple*. McGraw-Hill, New York, 1996.

[6] J. Robertson. *Engineering Mathematics with Mathematica*. McGraw-Hill, New York, 1995.

[7] R. Nicolaides and N. Walkington. *Maple: A Comprehensive Introduction*. Cambridge University Press, Cambridge, U.K., 1996.

[8] W. Shaw and J. Tigg. *Applied Mathematica*. Addison Wesley, Menlo Park, Ca., 1994.

[9] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *Computer J.*, 3, 1960.

[10] R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *Computer J.*, 6:163–168, 1963.

[11] M. Aoki. *Introduction to Optimization Techniques*. Macmillan, New York, 1971.

[12] L. E. Scales. *Introduction to Nonlinear Optimization*. Springer-Verlag, New York, 1985.

[13] Jr. J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, N.J., 1983.