Five example optimization problems

There are three goals in starting the course with examples:

1) To suggest how optimization can come from real-world applications.
2) To allow you to create, for yourself, some of the basic theoretical and numerical ideas for how to solve such problems.
3) To provide examples on which to practice/learn MATLAB/OCTAVE[1] programming.

While the textbook[2] does not provide fleshed-out examples, all the experts have been exposed to many of them.

This document does *not* address how to solve these problems. That will be done in class, on homework, and in additional handouts.

In fact, Assignment #1 asks you to solve some of these examples. Regarding goal 2), your method may be "brute force" or inefficient . . . *that is just fine!* The textbook and the rest of the content of the course will make more sense if you see brute force approaches before elegant algorithms.

Note that each example has a "code name" like "`fit`." I will also use this name for my MATLAB/OCTAVE code (e.g. "`fit.m`") when I hand out solutions. Later Assignments will be largely based on textbook exercises, but I'll add my own problems as needed.

I. (`calcone`) Let

$$f(x) = \left(x^2 + \cos x\right)^2 - 10\sin(5x).$$

Compute

$$\min_{x\in[0,2]} f(x)$$

You may write the same problem as an inequality-constrained optimization in form (1.1), on page 3 of the textbook, namely as

$$\min_{x\in\mathbb{R}} f(x) \qquad \text{subject to} \qquad \begin{array}{c} x \geq 0 \\ 2 - x \geq 0 \end{array}$$

In this standard form, $\mathcal{E} = \emptyset$, $\mathcal{I} = \{1,2\}$, $c_1(x) = x$, and $c_2(x) = 2 - x$. However, I assert no particular benefit to this standardized form in this case.

You saw such problems in Calculus I. This one is hard to do by hand; it benefits from some programming and visualization using MATLAB. Of course one may plot $f(x)$ on the given interval; one gets close to the solution just from a picture.

---

[1]You may use other languages such as PYTHON, but I will only provide MATLAB/OCTAVE examples and solutions.

[2]J. Nocedal & S. Wright, *Numerical Optimization*, 2nd ed., 2006

II.   (fit)   Consider the following 11 data which are plotted below:

x	0.000	0.100	0.200	0.300	0.400	0.500	0.600	0.700	0.800	0.900	1.000
y	4.914	3.666	2.289	1.655	1.029	0.739	0.393	0.090	-0.197	-0.721	-0.971
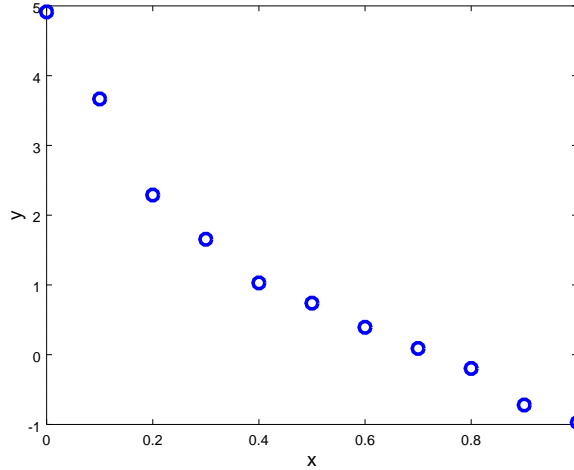
Suppose we believe that this data can be fit by a function of the form $g(x) = c_1 + c_2 x + c_3 e^{-5x}$. If the sense of "fit" is that the sum of the squares of the misfits should be as small as possible then we would solve

$$\min_{c \in \mathbb{R}^3} f(c)$$

where we define the objective function to be

$$f(c) = \frac{1}{2} \sum_{j=1}^{11} (g(x_j) - y_j)^2 = \frac{1}{2} \sum_{j=1}^{11} \left(c_1 + c_2 x_j + c_3 e^{-5x_j} - y_j \right)^2.$$

This problem is already in form (1.1) from the textbook, with $c = [c_1, c_2, c_3]$ an unknown vector of coefficients in $\mathbb{R}^3$, and with no constraints ($\mathcal{E} = \emptyset, \mathcal{I} = \emptyset$). Here x_j, y_j are the data, with $x_j = 0.1(j - 1)$ regularly-spaced. Note we are *not* finding x_j or y_j values in the minimization process; these values merely determine the objective function. Also note that the overall factor of $1/2$ in defining the function $f(c)$ is just a convenience for differentiating.[3]

III. (salmon)   Ed and Thomas caught 21 salmon, and all will be filleted. Of these, x_1 will be eaten fresh, which requires 2 time units per fish. Then x_2 will be vacuum-packed and frozen (3 time units per fish) and another x_3 will be smoked, vacuum-packed, and frozen (4 time units per fish). Thus the total amount of processing time is $2x_1 + 3x_2 + 4x_3$. However, at most 2 fish can be eaten fresh before they go bad, and at most 10 fish can be smoked in the time allowed.

---

[3]This is a hint about the standard algorithms for finding the minimum.

Finding x_1, x_2, x_3 to minimize the total processing time means solving this constrained problem with $f(x) = 2x_1 + 3x_2 + 4x_3$:

$$\min_{x \in \mathbb{R}^3} f(x) \qquad \text{subject to} \qquad \begin{array}{c} x_1 + x_2 + x_3 = 21 \\ 0 \le x_1 \le 2 \\ 0 \le x_2 \\ 0 \le x_3 \le 10 \end{array}$$

This problem is not in standard form (1.1), but an easy exercise on A #1 asks you to put all of the inequality constraints into the form "$c_i(x) \ge 0$" to make it so.

IV.   (tsp)   Jill sells amazing devices which make your brain absorb math more easily. To sell these devices she plans to visit cities A, F, J, N, W starting and ending at city S. Some pairs of cities have connecting flights and some do not; the one-way costs of the various flights are shown below in a *graph* with costs ("weights") on each connection ("edge"). It is clear to Jill that she should visit each city exactly once, except for returning to S at the end. Note that a possible itinerary is expressible as a seven-letter string like "SANFWJS."
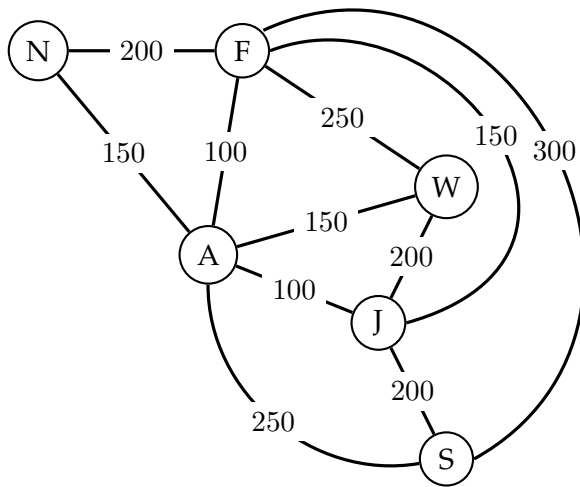
This is an example of the famous *traveling salesperson problem*. If x denotes a feasible seven-letter string (itinerary) then we may define the objective function $f(x)$ to be the cost of that itinerary; thus $f(x)$ is defined using the edge weights. Finding a feasible itinerary is generally nontrivial—it corresponds to finding a *Hamiltonian cycle*—in fact one may add in all remaining edges with large weights so that any itinerary x is feasible with well-defined cost $f(x)$.

The problem *could* be written

$$\min_{x \text{ is feasible itinerary}} f(x)$$

but this does *not* mean it is in standard form (1.1) because there is no sensible way to treat the possible inputs as "$x \in \mathbb{R}^n$." This is a *discrete optimization* problem, not a continuous problem as covered in form (1.1) and the textbook generally.

V.   (beam)   A new tent design requires bending a pole to go from one corner of the tent to another while also having certain heights at particular locations. Note that the corners of the tent, and the ends of the pole, are on the ground. The pole is modeled as a *beam* in standard engineering language. We want to know the shape of the pole so as to determine (1) its (arc) length and (2) how fabric can be cut to reach the pole and yet be taut below it.

To be specific, the length of the tent along the ground is π length units. We seek a function $y = h(x)$, defined on the interval $x \in [0, \pi]$, for the height of the pole above the ground. This function must be at least twice-differentiable and satisfy $h(0) = 0$ and $h(\pi) = 0$. The theory of beams says that actual bent shape will be the minimizing solution h of

$$\min_h I[h] \qquad \text{where} \qquad I[h] = \frac{1}{2} \int_0^\pi |h''(x)|^2 \, dx.$$

Here I is called the objective *functional*. The input h into I is itself a function from an (infinite-dimensional) vector space of functions.

The following particular inequalities, shown in the figure below, give locations where the heights are constrained:

$$0.9 \leq h(1) \leq 1.1$$
$$1.2 \leq h(2) \leq 1.4$$
$$0.4 \leq h(3) \leq 0.6$$

The figure does not show the shape of the tent pole. That is, it does not show the *solution* of the minimization problem. That solution, after all, has minimal concavity, i.e. size of second derivative, subject to the constraints. The figure does, however, show a *feasible* $h(x)$ which has the required differentiability, satisfies $h(0) = h(\pi) = 0$, and satisfies the inequality constraints.