MATH464 – Lecture 1 (01/10/2023)

This is Linear Optimization (CAPSTONE).
I'm Bala Krishnamoorthy, call me Bala.

Today:
* syllabus, logistics
* Optimization in calculus
* Toy example

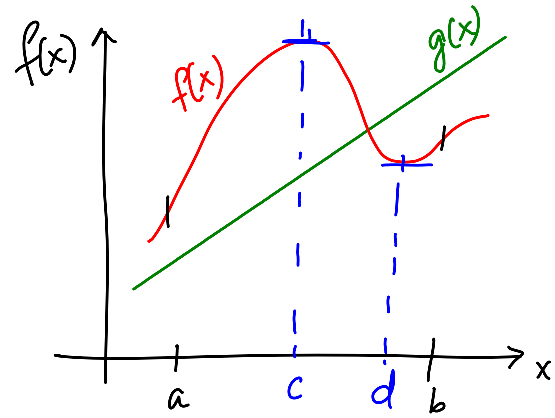
Linear Optimization (or linear programming, LP) is the most basic form of optimization. The word programming is used more commonly than optimization (along with "linear", "non linear", or "integer" etc.) Programming just means a set of instructions (and not computer programming).

LP is used in many areas. My own research includes theory and applications of integer linear programming, or IP in short, where variables are restricted to integers. Related areas I work on include combinatorial optimization, topology, computational biology, geometric measure theory, etc.

This course will place a lot of emphasis on the theory behind LP – so, there will be **many** proof-type exercises in the homework. We will also have a project that will involve the implementation and testing of (some of) the algorithms learned in class.

Optimization in Calculus

Find min/max of $f(x)$ for $a \le x \le b$



min: $f'(x)=0,\ f''(x)>0 \quad x=d$

max: $f'(x)=0,\ f''(x)<0 \quad x=c$

Need to also consider $f(x)$ at the endpoints $x=a$ and $x=b$.

Here, the minimum of $f(x)$ over $[a,b]$ is at $x=a$, and the maximum is at $x=c$.

Instead of $f(x)$, if we consider $g(x)$ which is linear, you need to check only the end points!

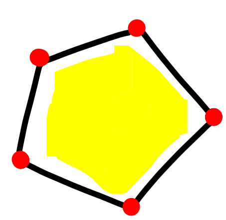
In this course, we will study generalizations of this easier case, i.e., linear function, to higher dimensions

$$\max/\min\ f(x_1,x_2,\dots,x_n) = c_1x_1+c_2x_2+\dots+c_nx_n = [c_1 \cdots c_n]\begin{bmatrix}x_1\\ \vdots\\ x_n\end{bmatrix} = \bar{c}^T\bar{x}$$

subject to linear constraints in $x_1,\dots,x_n$.

My notation:
$\bar{c},\bar{x},\bar{y}$: vectors
x,α,β: 1D variables
A,B: sets/matrices

Things are harder because of the high dimension, but still "easy" because of linearity.



Instead of the closed interval $[a,b]$, we get "polytopes" (generalizations of polygons in 2D).

It turns out we can get away with looking at the corner points, or vertices (just like the end points a,b of $[a,b]$) here!

An aside on optimization classes at WSU

364 - Intro to LP (no proofs)
464 - Linear Optimization (with proofs)
461/566 - Network optimization
564 - Nonlinear Optimization (convex)
565 - Nonconvex Optimization, Optimization in ML.
567 - Integer optimization

Recall from linear algebra (Math 220):

$A\bar{x} = \bar{b}$ → Notation: x, y, α, β : single variables
$\bar{x}, \bar{y}$: vectors (lower case letters with "bar")
A, B, L, U : matrices, sets

$[A \mid \bar{b}] \xrightarrow{\text{EROs}}$ RREF (reduced row echelon form)

elementary row operations

If RREF has a row $[0\ 0 \cdots 0 \mid *]$ where $*$ is nonzero, then the system is inconsistent.

Else, the system has
- unique solution
- infinitely many solutions.

We typically work in this setting

From among the infinitely many solutions, find a solution $\bar{x} = \bar{x}^*$ to $A\bar{x} = \bar{b}$ that optimizes $f(\bar{x}) = \bar{c}^T\bar{x}$.

A motivating example

We illustrate the whole pipeline on an example – start with a "word problem", formulate it as a linear program, and also solve it (on the computer).

"Math"!

Dude M. Major's Thursday problem:

Dude has 5 hrs to spend, has two options – party, or get tutored on math. He has \$48 to spend. Here are the costs involved:

Costs:
- tutoring – \$8/hr
- party – \$16/hr

Utility
- tutoring → 2/hr
- party → 3/hr

Dude estimates the utility of each activity in some units. It could be, e.g, in 100s of dollars.

Decide how long Dude will party, and how long he'll get tutored, so that his total utility is maximized.

<u>Decisions to make</u>: How many hours to party, how many hours to get tutored, so that the total utility is maximized?

Let x_1 = # hrs of tutoring and x_2 = # hrs of partying.

$$
\left.
\begin{array}{llll}
\max & z = 2x_1 + 3x_2 & & \text{(total utility)} \\
\text{s.t.} & x_1 + x_2 & \leq 5 & \text{(max time)} \\
 & 8x_1 + 16x_2 & \leq 48 & \text{(max money)} \\
 & x_1 \geq 0,\ x_2 \geq 0 & & \text{(nonnegativity)}
\end{array}
\right\} \text{Linear Program (LP)}
$$

maximize

subject to

The brief "descriptions", e.g., (total utility), are important to help us comprehend what the model is capturing in each constraint and the objective function. Similarly, the sign restrictions, i.e., non-negativity here, are also important.

We solve the problem using the software AMPL, which we will introduce in detail later on.

The "model" (Dude.txt)

```
var x {1..2} >= 0;

maximize TotalUtility: 2*x[1] + 3*x[2];

subject to MaxTime: sum {j in 1..2} x[j] <= 5;
subject to MaxCost: 8*x[1] + 16*x[2] <= 48;
```

Session from AMPL:

```
sw: ampl
ampl: option solver cplex;
ampl: model dude.txt;
ampl: expand MaxTime;
subject to MaxTime:
        x[1] + x[2] <= 5;

ampl: solve;
CPLEX 20.1.0.0: optimal solution;
objective 11
2 dual simplex iterations (1 in phase I)
ampl: display x;
x [*] :=
1  4
2  1
;
```

Dude will get tutoring for 4 hours and party for 1 hour, to get the maximum total utility of 11 units.

We will also learn how AMPL solved the problem, i.e., the mathematics behind the scenes.