Fri, March 24, 2023
Lecture #26
MSSC 6000

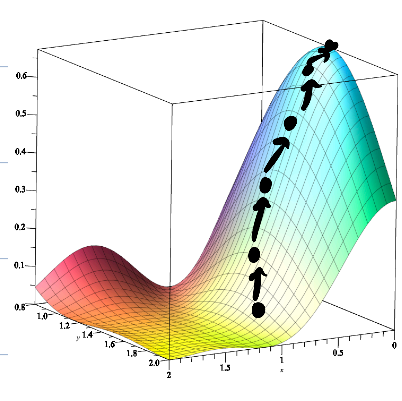
Announcements

* Normal Office Hours Monday
* OH Wed will be moved
* HW 4 assigned today, due Mon, Apr 3
  ↳ on D2L, two attachments

Topic 10 - Introduction to Metaheuristics

Gradient Ascent:

* start at a point
* compute gradient (vector that tells you "direction of steepest ascent")
* move a little in that direction
* repeat



you end up at the top of <u>some</u> hill and if you're lucky it's the tallest hill

Adapting to a discrete search space:

<u>Ex</u>: TSP

- search space: all tours on the set of cities

Each <u>tour</u> is a location on the landscape (a place in the mountains)

- need a definition for two tours to be "nearby"

<u>Ex</u>: cities = {1, 2, 3, 4, 5}

tour: 3 -> 5 -> 2 -> 1 -> 4 -> 3

what tours are "close"?

Up to you how to define this.

One possibility: the nearby tours

are the ones you get by swapping any two internal cities

3 -> 5 -> 2 -> 1 -> 4 -> 3

3 -> 1 -> 2 -> 5 -> 4 -> 3

* start at a random tour
* calculate the score of all nearby tours
* move to the best (cheapest) one
* repeat

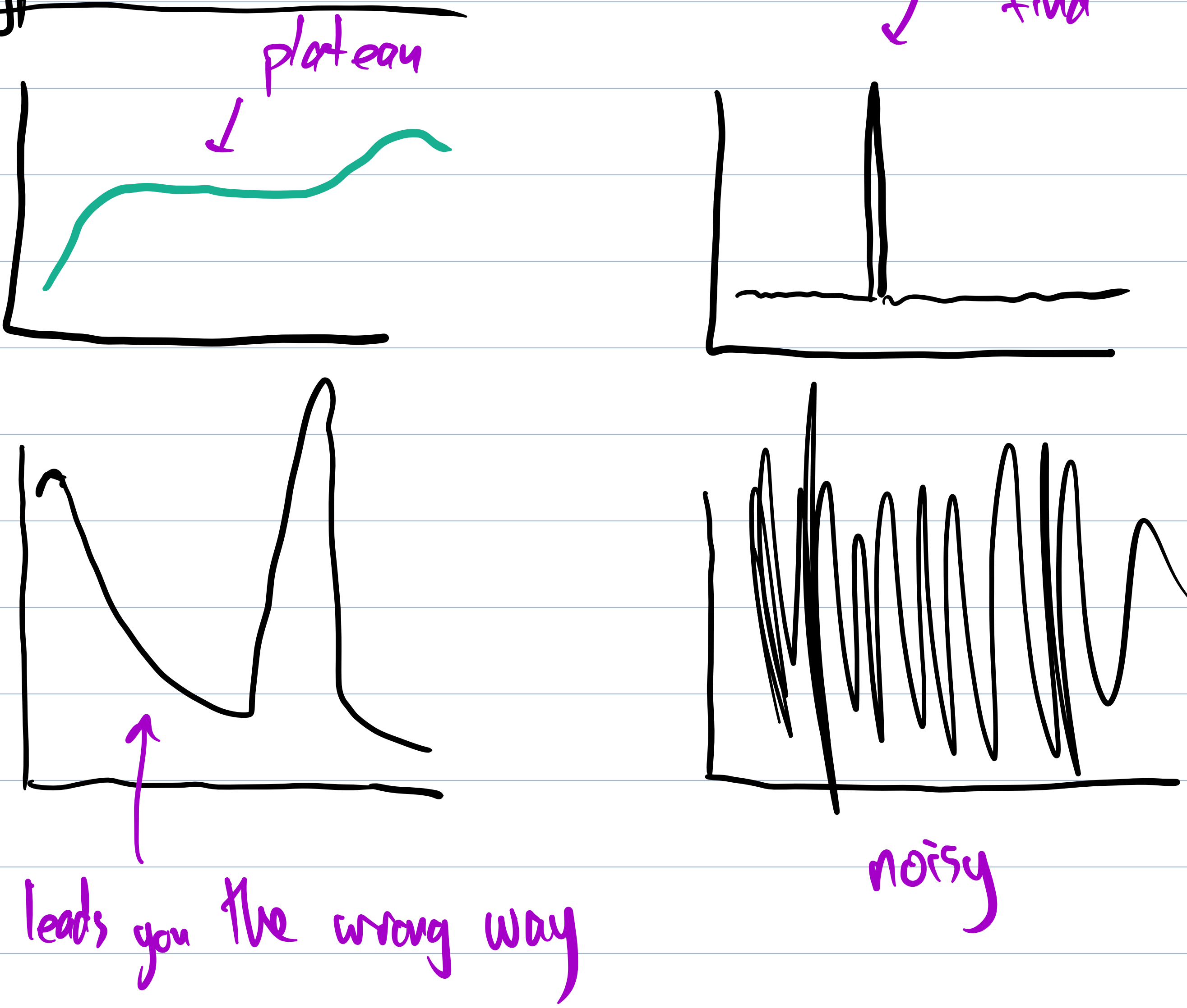
This will find a local optimum

↳ max or min

↳ best in it's area, but maybe not best overall

Metaheuristics are all about smart ways to explore the search space.

Typical Obstacles:



Topic 11 - Hill-Climbing

Goal: Develop a MH with Gradient Ascent as our inspiration. We want to find a global optima for either continuous or discrete search spaces.

Problem Setup:

* Search space S full of candidates
* Scoring function: score(x), $x \in S$
  (also called "fitness" or "quality")
* A way to generate either:
  - all the candidates "nearby" a candidate (the "neighborhood of x", nbhd(x))
    (doesn't make sense for continuous problems)
  - a single random candidate near a candidate
    (a "tweak", tweak(x))

"nearby" is up to you to define, and definitions can completely change how a MH behaves.

Two running examples:

(1) TSP

* discrete
* score = cost of tour, want to minimize
* nbhd(x):

Suppose $x = C_1 \to C_2 \to C_3 \to \ldots \to C_n \to C_1$

Define the nbhd of x to be all ways of picking two internal cities and swapping them.

How big is nbhd? $\binom{n-1}{2} = \frac{(n-1)(n-2)}{2} \approx \frac{n^2}{2}$

Pretty big nbhd.

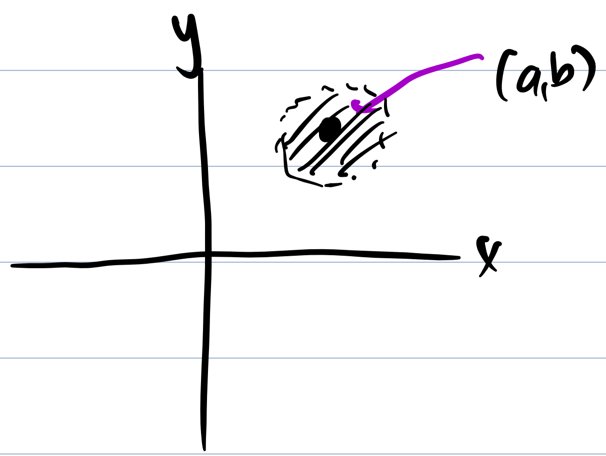
* tweak(x): one random thing in the nbhd

(2) optimizing a continuous function in two variables $f(x,y)$

not discrete

* continuous

* Search space
  all (x,y) points maybe within some interval
* score of a point = the value of f at that point
* nbhd((x,y))
  (what points are "near" a point in 2D, 3D, ..., 20D, space?)
  all points with a distance < δ from x for some small # δ.



* tweak((x,y)) =
  a random point in the neighborhood, like before.

MH #1: Random Search

```
best = random element of S
while True:                    (run until you're bored)
    x = random element of S
    if score(x) > score(best):
        best = x
```

Possible stopping conditions:

* best score does not improve for N consecutive tries
* a preset # of attempts
* run until you get bored

This is not a good MH! It does not use any of the old information to guide future choices.

Inspired by Gradient Ascent:

MH #2: Steepest Ascent Hill Climbing (for discrete only)

```
x = random element of S
while True:
    N = nbhd(x)
    s = element of N with the highest
        score
    if score(s) > score(x):
        x = s
    else:
        # we're at the top of a hill
        quit
```

If continuous, N is probably an infinite set, so we can't compute the score of everything in N.