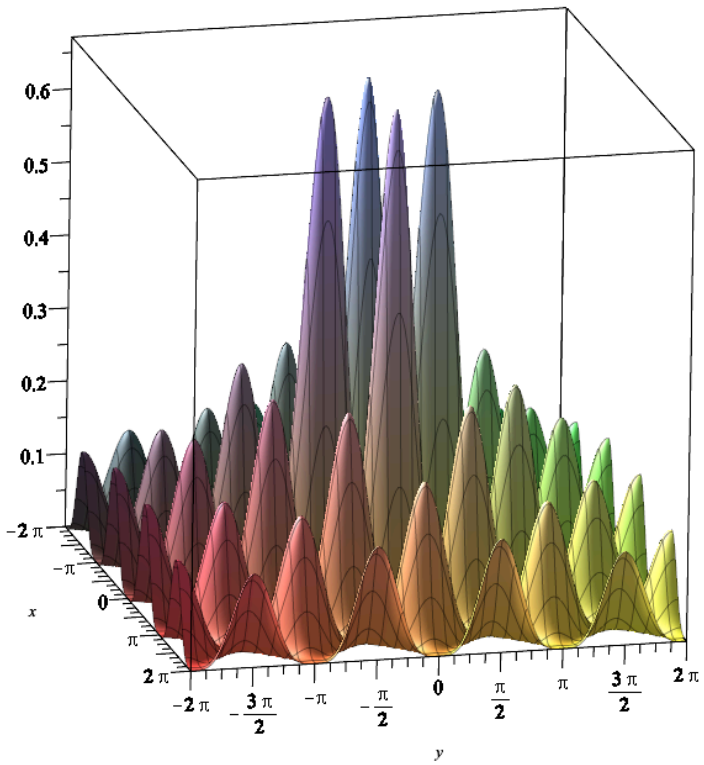
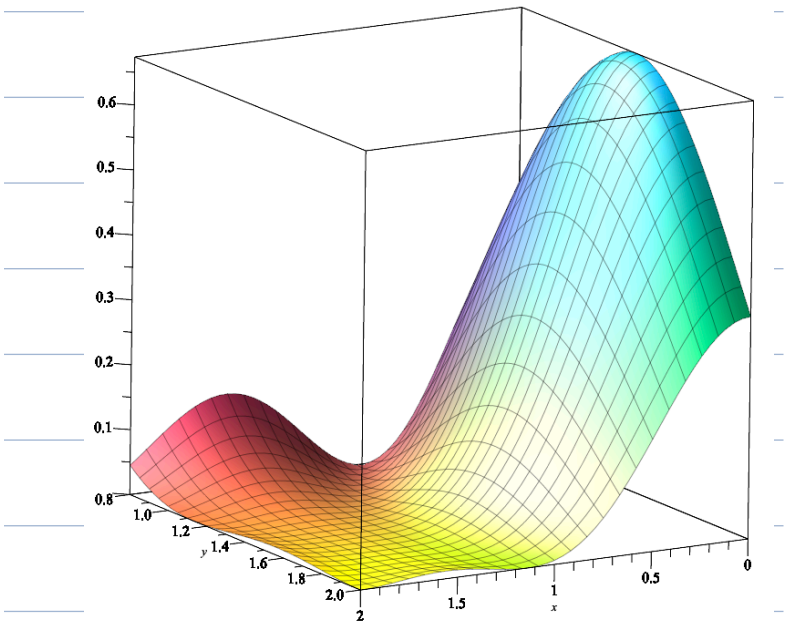

0.6
0.5
0.4
0.3
0.2
0.1
−2 π
−π
0
π
2 π
x
−2 π
−3 π/2
−π
−π/2
0
π/2
π
3 π/2
2 π
y


0.6
0.5
0.4
0.3
0.2
0.1
0.8
1.0
1.2
y 1.4
1.6
1.8
2.0
2
1.5
1
x
0.5
0

Monday, March 22 Lecture #24

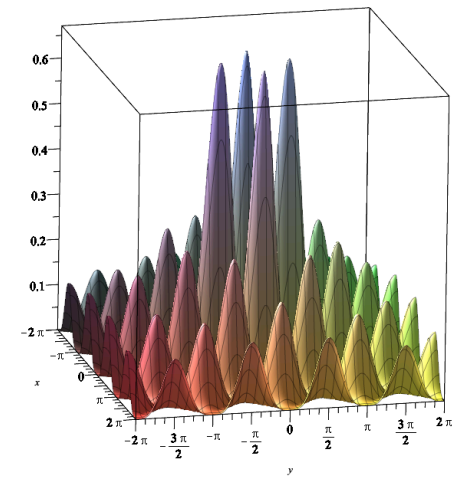
Announcements
* HW 4 due Wed.
* OH today 2:30 - 3:30
Tues 1:00 - 2:00
via Teams
* Midterm Grades

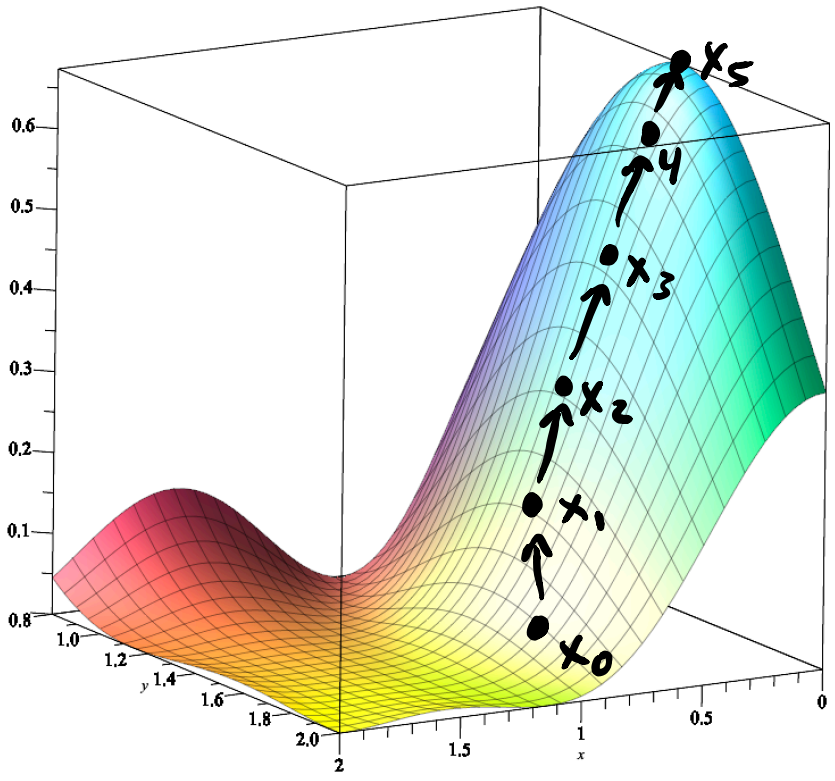
Intro to MHs

Gradient Ascent ("Descent")
Works on diff'bk functions

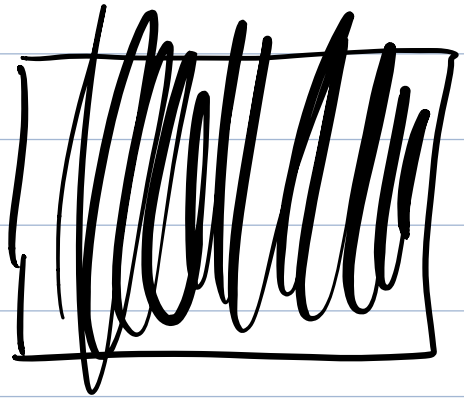
"gradient" → vector that points in the direction of steepest ascent

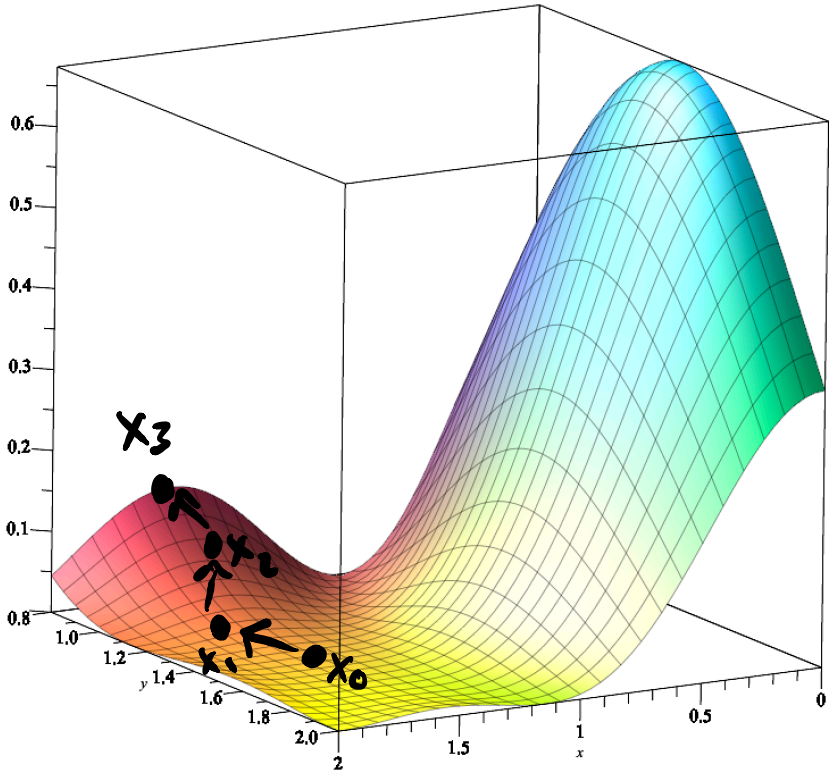
(1) start at a random pt
(2) compute the gradient
(3) move a little in that direction
(4) repeat





Where do we end up? The top of some hill. A local maximum.



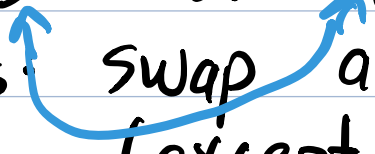


Discrete spaces (finite)
- no gradient — what do we do?

[pretend you're in the mountains]
GA:
* look in small radius
how? * find the point in your radius that is highest
* go there and repeat

Ex: TSP
- search space: all tours on the graph (these are the places on the mountain you could be standing)
- need a definition of "nearby" / "small radius"

cities 1, 2, 3, 4, 5
tour 3 → 5 → 2 → 1 → 4 → 3
nearby tours: swap any two cities (except the first)

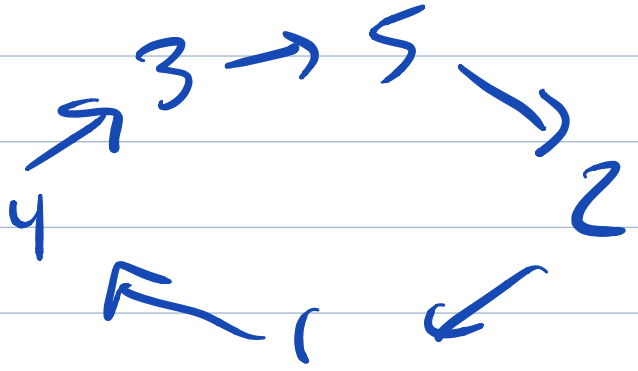


3 → 1 → 2 → 5 → 4 → 3

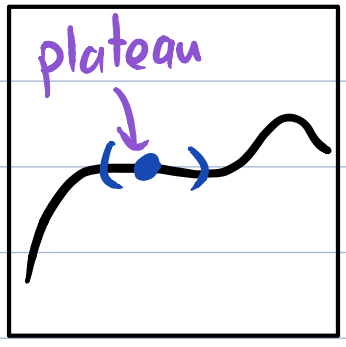
* start at a random tour
↱ * calculate the score of all nearby

tours

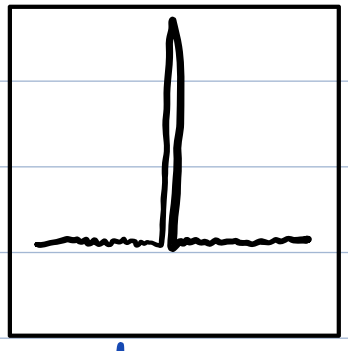
* move to cheapest one
* repeat



MHs are all about exploring the search space in clever ways in the hopes of finding a good solution.

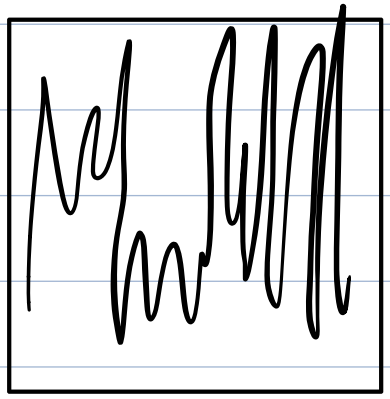


don't know where to go



needle-in-a-haystack

leads you the wrong way



noisy

small steps can make big differences

Topic 12 - Hill Climbing

Design MHs that mimic gradient ascent and work for:

* discrete spaces
* continuous spaces where we can't compute a gradient

Problem Setup:

* Search space S full of candidates / solutions
* Scoring function: score(x) for any $x \in S$ ("fitness" / "quality")
* A way to generate either:
  - all the candidates near some candidate - the "neighborhood" nbhd(x)

  OR
  - a random candidate nearby some candidate, tweak(x).

might not make sense for continuous problems

Two running examples:

(1) TSP — discrete

score = length of tour

nbhd = all ways of swapping 2 cities

tweak = swap 2 random cities

nbhd is huge

n cities $\Rightarrow \binom{n}{2}$ $O(n^2)$

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

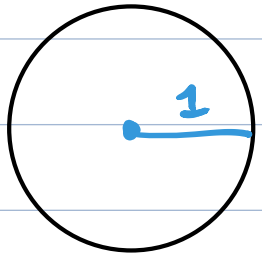
search space = all (x,y) points in some specified domain

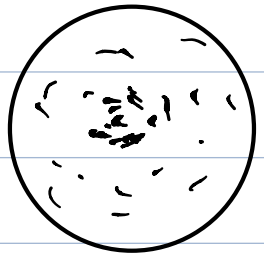
score = value of f at the point

nbhd = all points within some fixed distance δ of x

tweak = one random point in the nbhd.

picking random points in a circle





MH #1: Random Search

```
best = random element of S
while True:
    x = random element of S
    if score(x) > score(best):
        best = x
```

Stopping Conditions:

* best score does not improve for N iterations
* preset number of iterations
* you get impatient

This is a bad MH usually. It doesn't use any information of old solutions to guide future choices.