Classical Search + Hill Climbing

Note Title

Classical Binary Search:

BS(x, key, lo, hi)

```
if (lo > hi)
   return -1

m ← (lo+hi)/2

if x_m == key
   return m
if x_m > key
   return BS(x, k, lo, m-1)
if x_m < key
   return BS(x, k, m+1, hi)
```

given $X_0, X_1, \ldots X_{n-1}$

$X_0 \leq X_1 \leq \cdots \leq X_{n-1}$

Find K such that $X_K = key$

or return -1 if no K exists

Knowledge base: Array $X_0 \ldots X_{n-1}$, key, state = [lo, hi]

Initial state: [0, n-1]

Goal: lo == hi

$mid = \frac{lo+hi}{2}$

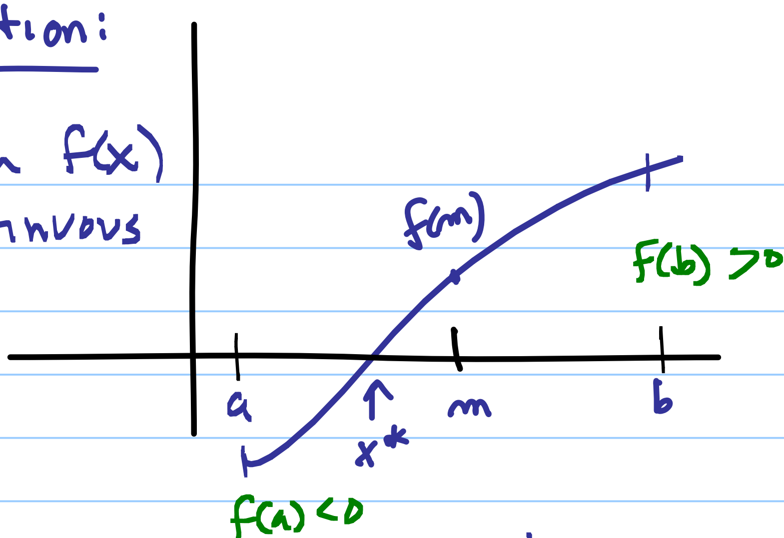
Rules:

[lo, hi] → [lo, mid-1] Precondition: $X_{mid} > Key$	[lo, hi] → [mid, mid] Precondition: $X_{mid} == Key$
[lo, hi] → [mid+1, hi] Precondition: $X_{mid} < Key$	[lo, hi] → [-1, -1] Precondition: lo > hi

Hill-Climbing Function: $g(lo, hi) = lo - hi \quad \{ < 0 \text{ if } lo < hi, \; = 0 \text{ if } lo == hi \}$

Bisection:

given $f(x)$ continuous



find x^* such that $f(x^*) = 0$

$\boxed{\text{Classical Method}}$:

find $m = (a+b)/2$
if $|f(m)| < \epsilon$
    return m
else
    if $f(a)f(m) > 0$ // i.e., f(a) & f(m) have same sign
        a = m
    else
        b = m

Initial State: $[a,b]$

an interval for which f(a) and f(b) have opposite signs – i.e., $f(a)f(b) < 0$

Goal condition $b - a < \epsilon$

where ϵ is some small "tolerance" that says a & b are sufficiently close

Rules: $\left(m = \frac{a+b}{2}\right)$

- $[a,b] \to [a,m]$

  precondition:

  $f(b)f(m) > 0$ (same sign)

- $[a,b] \to [m,b]$

  precondition:

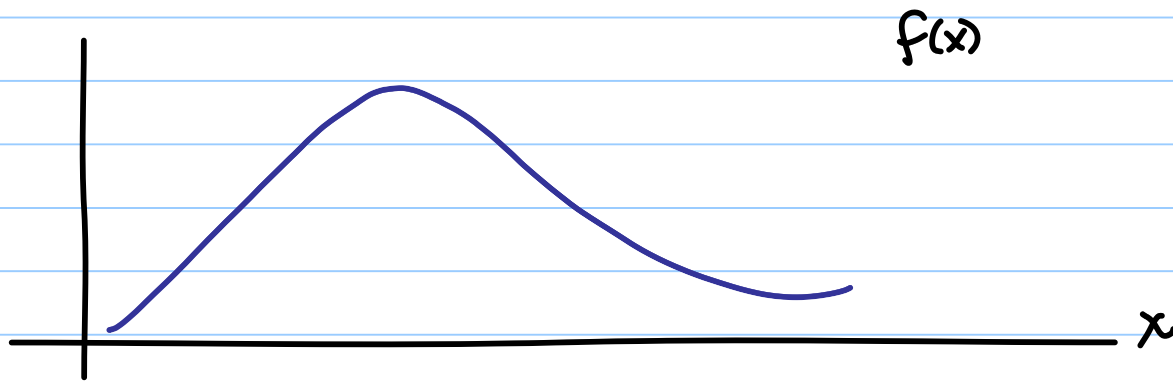
  $f(a)f(m) > 0$

Hill Climbing Function

$g(\text{state}) = g(a,b) = a - b$ improves as $a \to b$

Newton's Method

* find maximal value of a function



[$f(x) \geq 0, \ \forall x$]

Note: single maximum.

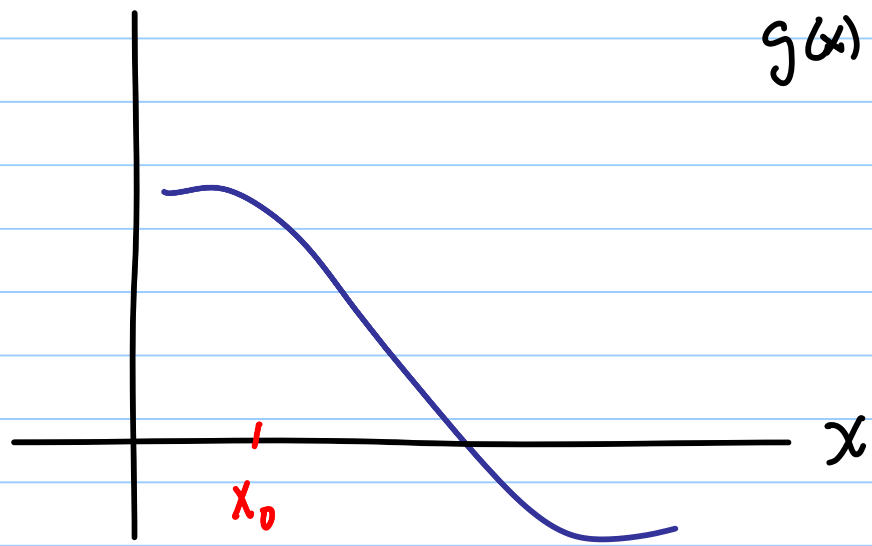
[Global vs. Local Optima

[Objective function

[Examples

One approach: if $f(x)$ is differentiable, solve for $f'(x) = 0$

Find a point where $g(x)=0$:

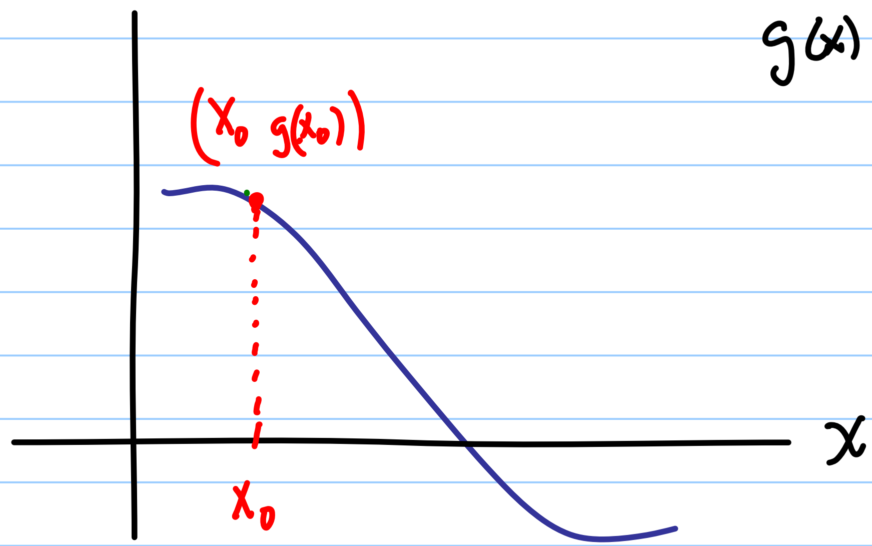


[Newton's Method]

start with an initial guess, x_0

update by moving in direction of slope at x_0 : $g'(x_0)$

Find a point where $g(x)=0$:

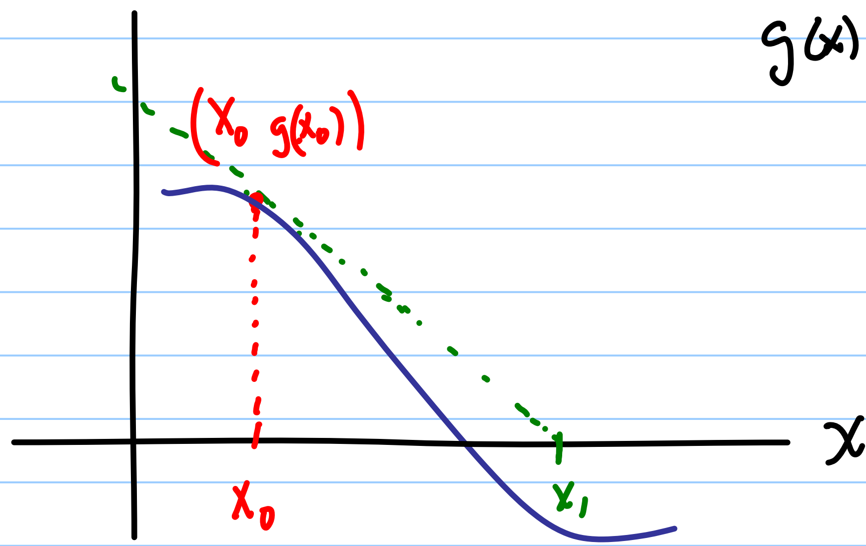


[Newton's Method]

start with an initial guess, x_0

update by moving in direction of slope at x_0 : $g'(x_0)$

Find a point where $g(x) = 0$:



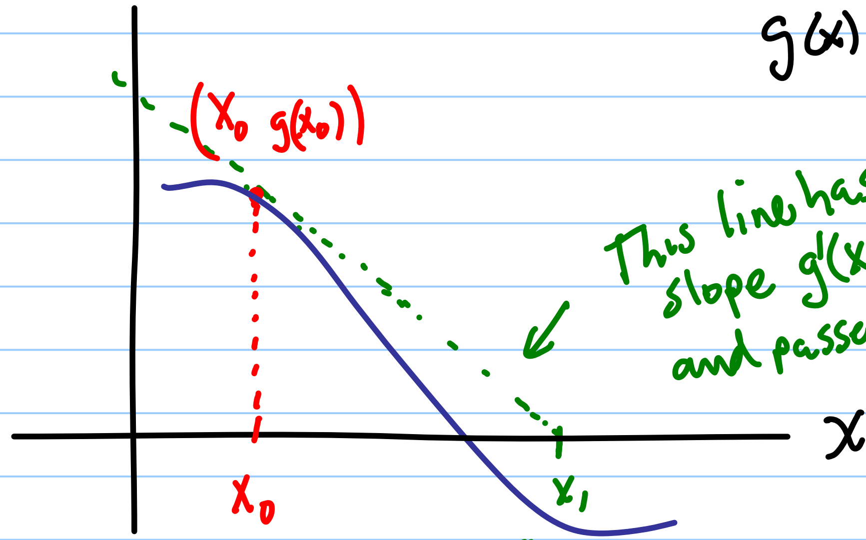
[Newton's Method]

start with an initial guess, x_0
update by moving in direction of slope at x_0 : $g'(x_0)$

Find a point where $g(x)=0$:

[Newton's Method]

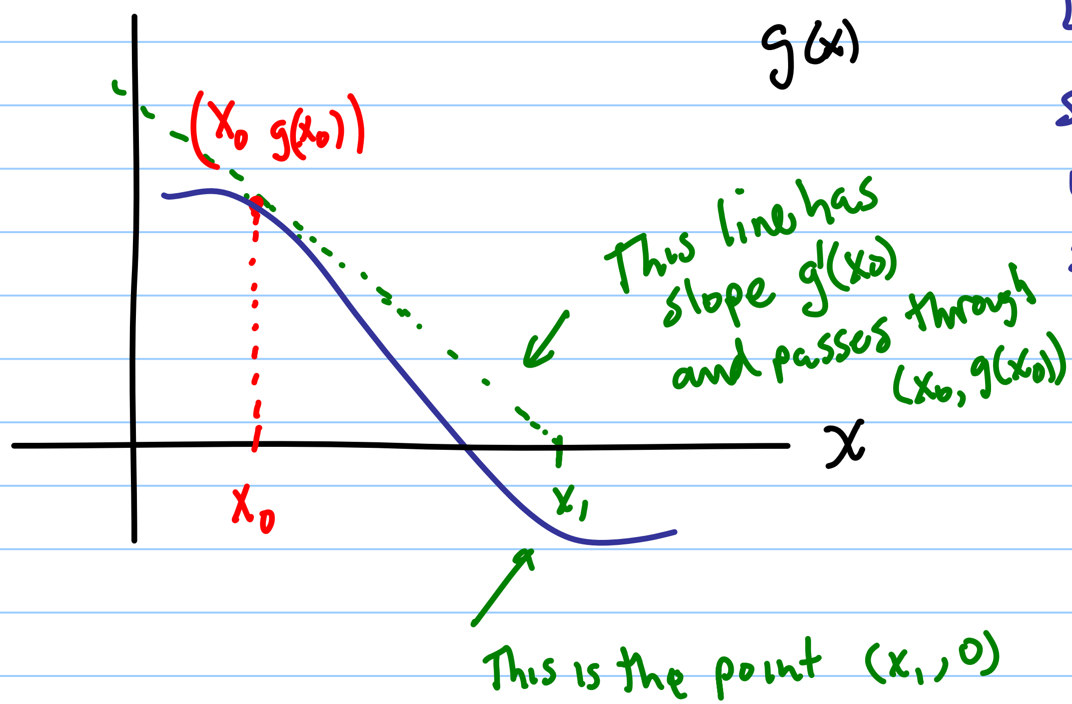
start with an initial guess, x_0,
update by moving in direction of
slope at x_0 : $g'(x_0)$



This line has slope $g'(x_0)$ and passes through $(x_0, g(x_0))$

This is the point $(x_1, 0)$

Find a point where $g(x)=0$:



[Newton's Method]

start with an initial guess, x_0

update by moving in direction of slope at x_0 : $g'(x_0)$

Using point-slope formula —

$$m = \frac{(y - y_0)}{(x - x_0)}$$

$$g'(x_0) = \frac{y - g(x_0)}{x - x_0}$$

$$= \frac{0 - g(x_0)}{x_1 - x_0}$$

$$(x_1 - x_0)\, g'(x_0) = -g(x_0)$$

$$\boxed{x_1 = x_0 - \frac{g(x_0)}{g'(x_0)}}$$

Example:

Solve $x^4 = 16$

$g(x) = x^4 - 16$

$g'(x) = 4x^3$

$$\frac{g(x)}{g'(x)} = \frac{x^4-16}{4x^3}$$

$$x_1 = x - \left(\frac{x^4-16}{4x^3}\right)$$

$$= \frac{3x^4+16}{4x^3}$$

guess $x_0 = 1$

$$x_1 = \frac{3x_0^4+16}{4x_0^3} = \frac{19}{4} = 4\tfrac{3}{4}$$

i	x[i]	x[i+1]	abs. err	rel.err
0	1	4.75	2.75	1.375
1	4.75	3.599823	1.599823	0.799912
2	3.599823	2.785614	0.785614	0.392807
3	2.785614	2.274264	0.274264	0.137132
4	2.274264	2.045744	0.045744	0.022872
5	2.045744	2.001512	0.001512	0.000756
6	2.001512	2.000002	1.71E-06	8.56E-07
7	2.000002	2	2.2E-12	1.1E-12
8	2	2	0	0
9	2	2	0	0
10	2	2	0	0

Newton's Method as Hill-Climbing

initial state $= x_0$

rules $= x \to \left[x - \frac{g(x)}{g'(x)}\right]$ (Precond: $g'(x) \neq 0$)

goal $= |x_{new} - x_{old}| < \epsilon$

Note: hillclimbing implies a function to be maximized

here, we are minimizing

ERROR $= |x_{new} - x_{old}|$

irrevocable: old values of state discarded

Principal

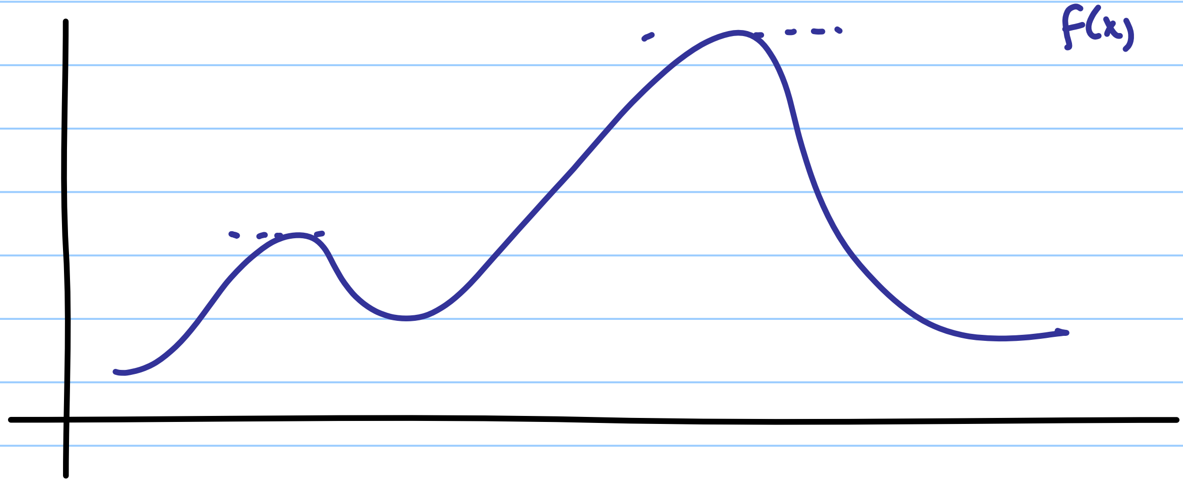
Maximizing $h(x)$

$\equiv$

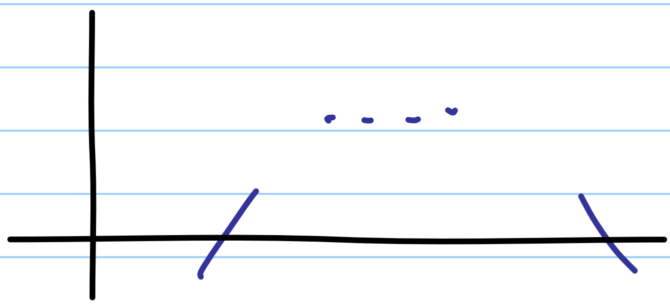
minimizing $-h(x)$

Note: This process finds a local maximum

Suppose



solve for $f'(x) = 0$



crosses axis more than once

The point of convergence may not be the global optimum!

Common occurrences of hill-climbing algorithms

- Machine Learning

  Find best set of weights $w_0, \dots, w_{k-1}$

  for a function $f(x) = w_0 f_0(x) + \dots + w_{k-1} f_{k-1}(x)$

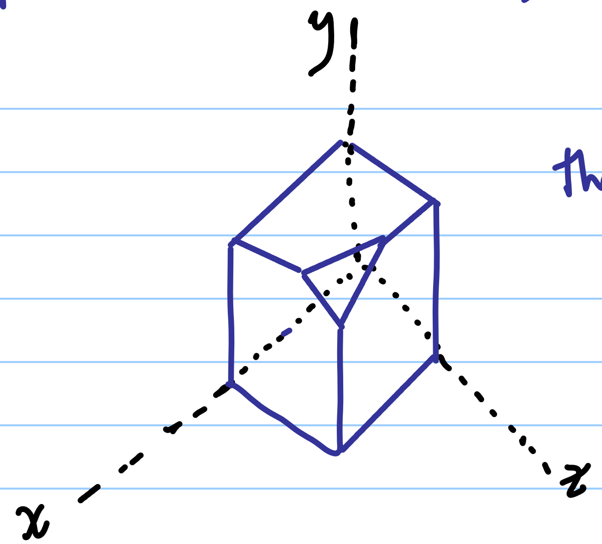
  - Start with initial set of weights $\underline{w} = (w_0, \dots, w_{k-1})$
  - Experiment (e.g. play game using this heuristic)
  - Adjust as needed $\underline{w}' = (w_0', \dots, w_{k-1}')$

- Training Neural Network:

  initial set of weights — $\underline{w}$

  compute new set $\underline{w}'$

- Simplex Method for solving Linear Programming problems.



The interior of the polyhedron is the set of all feasible solutions to a set of inequalities

$$a_{11}x_1 + \cdots + a_{1n}x_n \leq b_1$$
$$\vdots$$
$$a_{m1}x_1 + \cdots + a_{mn}x_n \leq b_m$$
$$x_1, \ldots x_n \geq 0$$

Find feasible sol'n with maximal value of objective function

$$z = c_1x_1 + \cdots + c_nx_n$$

Simplex Method:

- start at a vertex (which is a feasible solution

[in this case, $x_1=0, x_2=0, x_3=0$ is feasible]

- look at adjacent vertices [slide to one of them] — most likely, the one w/ highest value of z

- if optimal (there is a quick way of checking this!) stop

(stop when no adjacent vertex has a higher value of z than current vertex)

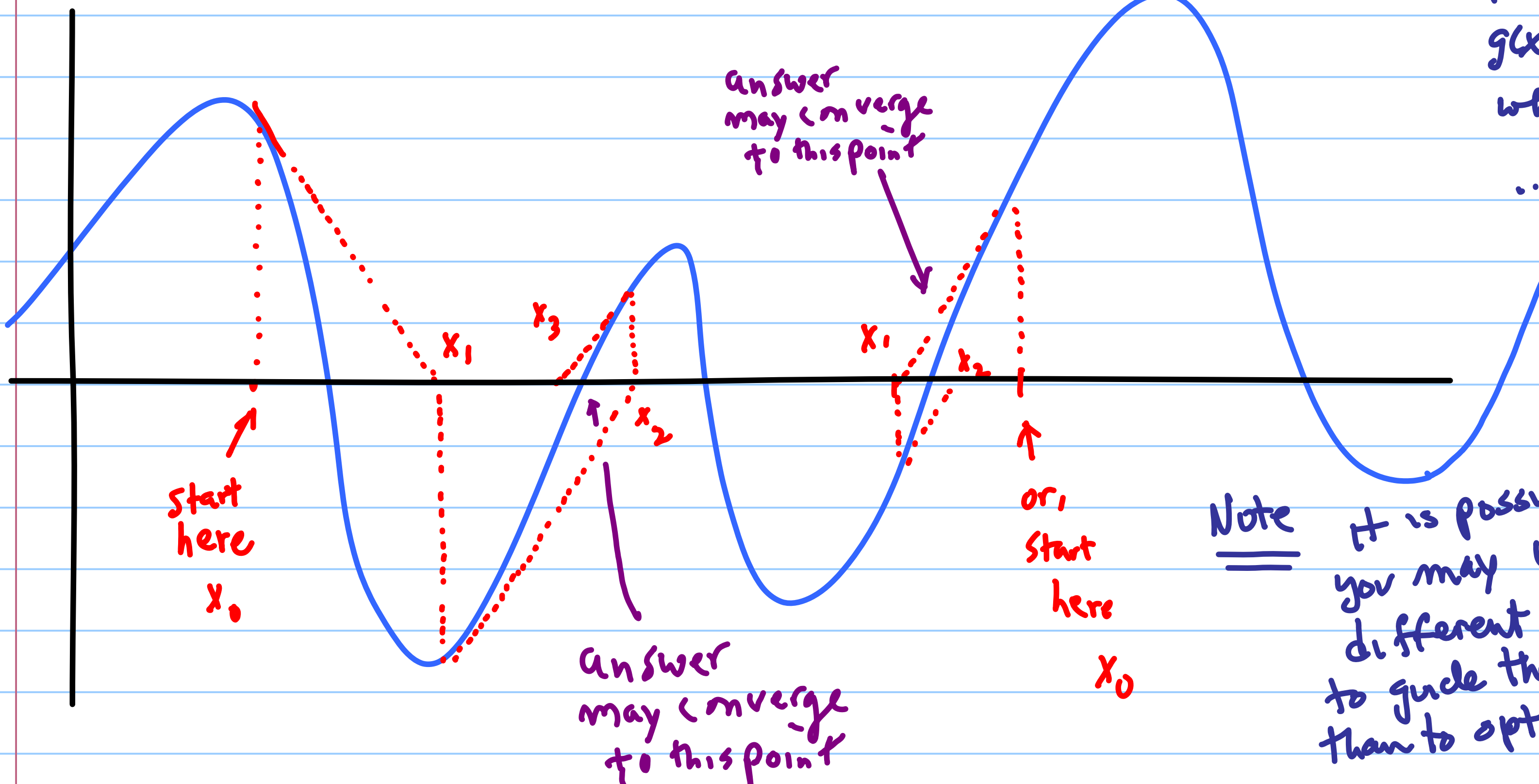
All of these are guided by some <u>objective function</u> — a function to be maximized —

if multiple optima exist, can get <u>stuck</u> at local optimum

by using <u>obj f'n</u> to drive the algorithm, cannot re-visit a previous point! [IMPORTANT - since we aren't maintaining a record of previous points visited]

Handling multiple optima

Try several initial guesses and run several times. Choose maximum



to maximize $g(x)$, find where $g'(x)=0$ ...

Note: it is possible that you may use a different function to guide the iteration than to optimize –