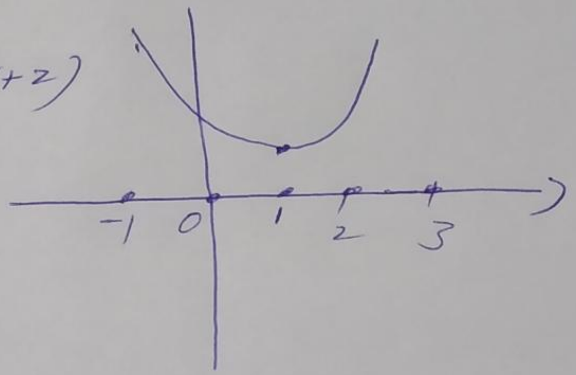
Gradient Descent

Given $f(x) = x^2 - 2x + 2 \overset{or}{=} (x-1)^2 + 1$

find $\hat{f}(x) = \min\limits_{x} (x^2 - 2x + 2)$



From calculus

By derivative

$f(x) = x^2 - 2x + 2$

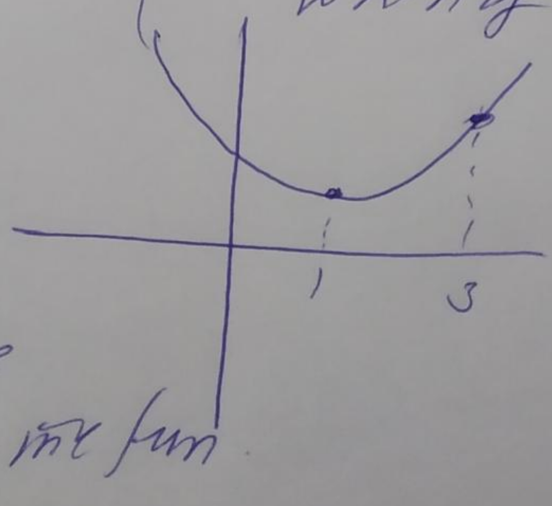
$f'(x) = 2x - 2 = 0 \Rightarrow x = 1$

now how to find by using Gradient Descent

$f(x) = \min\limits_{x} x^2 - 2x + 2$

we don't know the optimal x, so we pick a random no.

for ex if $x = 3$. (which is obviously wrong



step 1: we take the derivative of the fun

$f'(x) = 2x - 2$

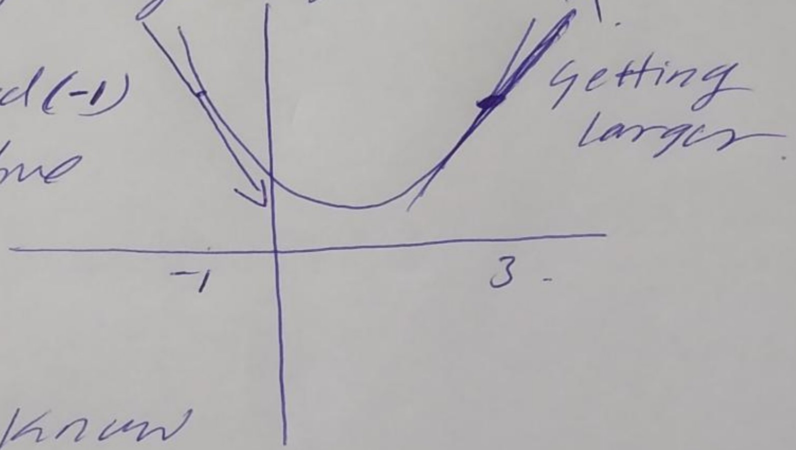
Step 2: we study the derivative at the point we guessed ($x=3$)

$$f'(3)=(2)(3)-2=4.$$

we know that the derivative at min should be zero.

Given that the derivative is +ve. we know that the value is getting larger

If we had guessed (-1) instead, the derivative would have been -4.



Then we would know that the fun is getting smaller and smaller

By studying the derivative of the current guess, we know if we are getting closer or further away from the minimum.

So, here is the equation

$$x_{i+1}=x_i-\alpha f'(x)$$

α → step length

(3)

$$x_{i+1} = x_i - \alpha f'(x)$$

x_{i+1} ↓ next guess. x_i ↓ initial guess. α → step length (say $\alpha = 0.2$).

Given our fx, we guessed $x_0 = 3$.

$$x_{i+1} = x_i - 0.2 f'(x)$$

$\underline{i = 0}$

$$x_1 = x_0 - 0.2 f'(3)$$

$$x_1 = 3 - 0.2(4)$$

$f'(x) = 2x - 2$

$f'(3) = 4$

$$x_1 = 2.2$$

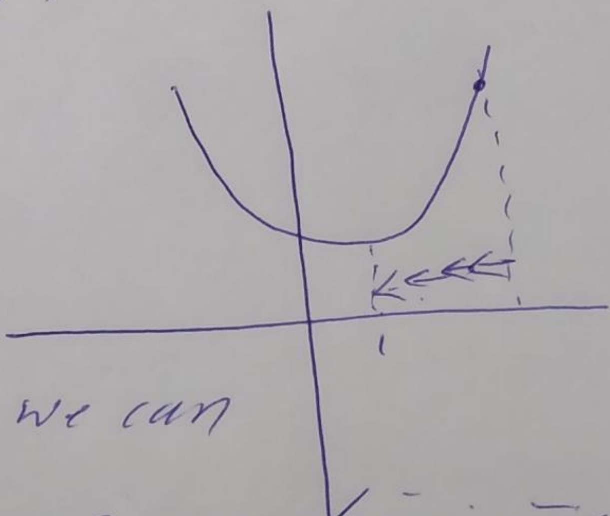
repeat this process again

$f'(2.2) =$

$$x_2 = x_1 - 0.2 f'(x_1)$$

$$x_2 = 1.44$$

III^{ly} $x_3 = 1.432$



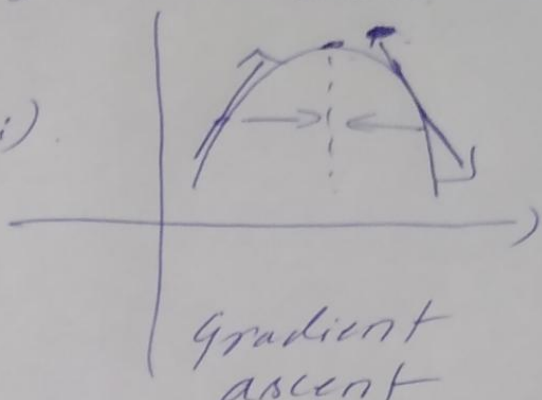
seeing this pattern, we can guess that if we keep repeating this process, we can find min^m pt of the soln.

This process is a pain to calculate by han[d] but if we write a small program, it is really easy to do...

If ~~we~~ we are looking for a $\underline{\underline{maxm}}$ value.

The Eq. is

$$x_{i+1} = x_i + \alpha f'(x_i).$$



Gradient ascent (Maxm).

Summarise

Gradient Descent

$$x_{i+1} = x_i - \alpha f'(x_i).$$

- we make an initial guess x_0.
- Derivative at our guess $f'(x_0)$.
- From this, we get x_1
- with x_1, we get x_2 the same way.
- Eventually at x_n, there is basically no change.

But the calculus approach seems easier.

Ex: $f(x) = x^2 - 2x + 2 \Rightarrow f'(x) = 2x - 2 = 0$
$\Rightarrow \underline{\underline{x = 1}}$.

Why we have to use gradient descend?

Well, that was a very simple ex., but in real life situation (complex Eqs), it is not easy.

<u>For Example</u> (complex fun).

$f(\cancel{x}) = e^{-\frac{x^T A x}{2\sigma^2}}$ where $\cancel{x} \in \mathbb{R}^n$ vector.

$A \in \mathbb{R}^{n \times n}$ matrix.

$f'(x) = -e^{-\frac{x^T A x}{2\sigma^2}}\left[(A+A^T)x\right]$

How can u set this to zero and solve for x ? (really messy)

we can find the derivative but solving for x is really <u>hard</u>.

$$\boxed{\text{Here definitely Gradient descend is useful.}}$$

<u>How to solve.?</u>

let $A = \begin{bmatrix} 2 & 1 \\ 0 & 1 \end{bmatrix}$, $\sigma = 1$.

$\hat{f}(x) = \max_{x} e^{-\frac{x^T A x}{2\sigma^2}}$

$f'(x) = e^{-\frac{x^T A x}{2\sigma^2}}\left[-(A+A^T)x\right]$

⑥

$$f(x) = [x_1 \; x_2]\begin{bmatrix} 2 & 1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

$$f(x) = [x_1, \; x_2]\begin{bmatrix} 2x_1 + x_2 \\ 0x_1 + x_2 \end{bmatrix} = 2x_1^2 + x_1 x_2 + x_2^2.$$

$$\frac{\partial f}{\partial x_1} = 4x_1 + x_2, \quad \frac{\partial f}{\partial x_2} = x_1 + 2x_2.$$

$$\begin{matrix} 4x_1 + x_2 \\ x_1 + 2x_2 \end{matrix} \Rightarrow \begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$= \left[\begin{bmatrix} 2 & 1 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 2 & 0 \\ 1 & 1 \end{bmatrix}\right]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$(A + A^T)x.$$

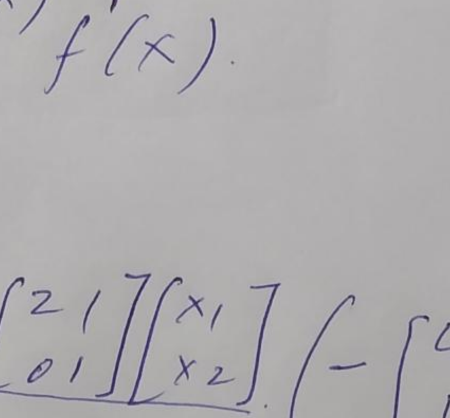
$$\frac{d}{dx}\left\{e^{f(x)}\right\} \Rightarrow e^{f(x)} f'(x).$$

now, therefore.

$$f'(x) = e^{-\frac{[x_1, x_2]\begin{bmatrix} 2 & 1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}{2}}\left(-\begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right).$$

$$x_{i+1} = x_i + \alpha f'(x_i) \qquad \text{say} \Rightarrow \alpha = 0.2$$

$$\text{at } x_{50} \cong \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \qquad x_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$



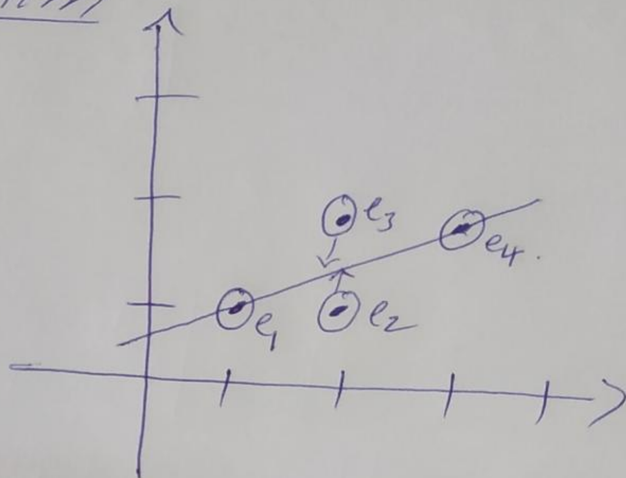
Gaussian model.

(7) by Chieh Wu.
(North Eastern Uni).

How to use gradient descend for a more realistic problem

Given data

X	Y
1	1
2	1
2	2
3	2



our goal is to find the best fit line.

let $y_{line} = \alpha X + \beta$

we want to find α and β.

First, we define error

(The error is the difference b/w the data and the line).

error

$e_1 = y_{1,line} - Y_1$

$e_2 = y_{2,line} - Y_2$

$\vdots$

In this case, the errors are:

$e_1 = 0$

$e_2 = 0.5$

$e_3 = -0.5$

$e_4 = 0$.

The total error.

$e_1 + e_2 + e_3 + e_4 = 0$.

$0 + 0.5 - 0.5 + 0 = 0$.

$e_1 + e_2 + e_3 + e_4 = 0$

(Does not make any sense, the error is obviously not zero, but e_2 and ~~e_4~~ e_3 cancelled each other).

What we normally do, is to find the error squared, so the errors can't cancel out.

Therefore: $e_1^2 + e_2^2 + e_3^2 + e_4^2 = e_{total}$

$0 + 0.25 + 0.25 + 0 = \underline{\underline{0.5}}$

So, the eq. for error is

$e_1 = y_{1,line} - \underline{y}_1$ (given data) $\Rightarrow e_1 = [\alpha x_1 + \beta] - \underline{y}_1$

$e_2 = y_{2,line} - \underline{y}_2 \Rightarrow e_2 = [\alpha x_2 + \beta] - \underline{y}_2$

$e_3 = y_{3,line} - \underline{y}_3 \Rightarrow e_3 = [\alpha x_3 + \beta] - \underline{y}_3$

$e_4 = y_{4,line} - \underline{y}_4 \Rightarrow e_4 = [\alpha x_4 + \beta] - \underline{y}_4$.

The sum of total error

$$e_{total} = \sum_{i=1}^{4} e_i^2 = \sum_{i=1}^{4} \left[(\alpha x_i + \beta) - \underline{y}_i\right]^2$$

In line fitting, we want to find the best line, so that the error b/w the line and the data is as small as possible.

now, we can set this problem as an optimization gradient descent problem

$$\min_{\alpha,\beta} \sum_{i=1}^{4} \left[(\alpha x_i+\beta)-Y_i\right]^2 \longrightarrow \underline{\text{cost fun.}}$$

(error)

we can recall, gradient descent

$$\begin{bmatrix}\alpha_{next}\\ \beta_{next}\end{bmatrix}=\begin{bmatrix}\alpha_{previous}\\ \beta_{previous}\end{bmatrix}-(0.2)\begin{bmatrix}\frac{\partial f}{\partial \alpha}(\alpha_{previous})\\ \frac{\partial f}{\partial \beta}(\beta_{previous})\end{bmatrix}$$

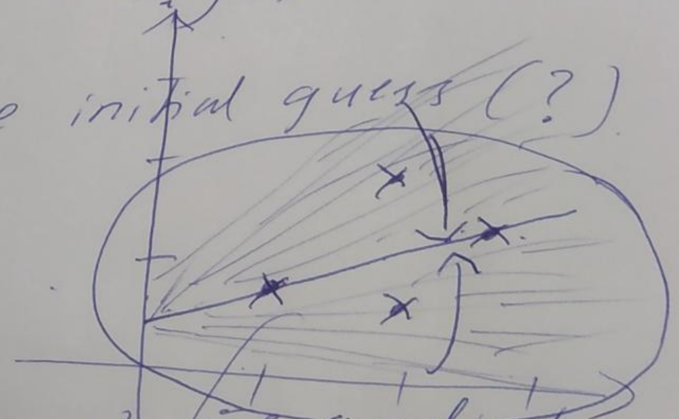
ie $f(\alpha,\beta)=\sum_{i=1}^{4}\left[(\alpha x_i+\beta)-Y_i\right]^2$

$$\frac{\partial f}{\partial \alpha}=\sum_{i=1}^{4} 2(\alpha x_i+\beta-Y_i)\,x_i$$

$$\frac{\partial f}{\partial \beta}=\sum_{i=1}^{4} 2(\alpha x_i+\beta-Y_i)(1)$$

we can set some initial guess (?)

$$\begin{bmatrix}\alpha_0\\ \beta_0\end{bmatrix}=\begin{bmatrix}1\\ 1\end{bmatrix}$$



now, if we run through the gradient descent method, the soln is $\alpha=0.49$
$\beta=0.5$
(Good Estimation).

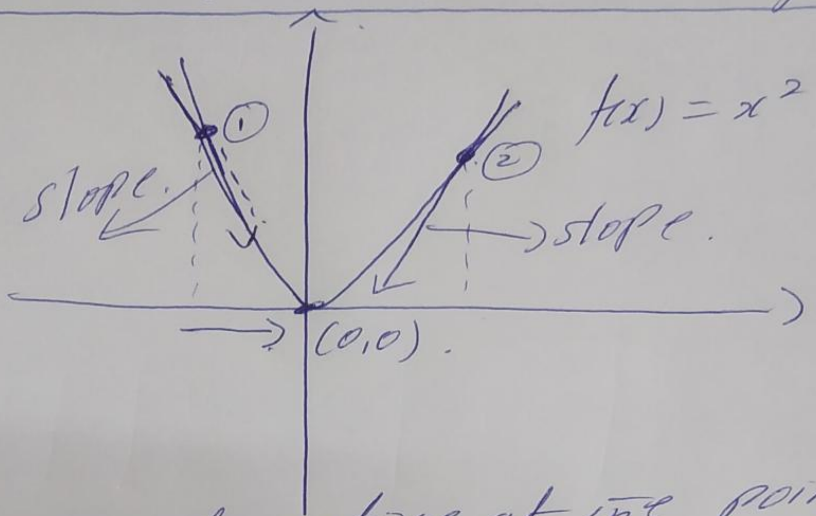
real soln is

$\alpha=\beta=0.5$

These lines are the steps the algorithm took to get the final soln..

How to choose the initial guess(?)

EX:



find the derivative at the point 1.

if $f'(1) \rightarrow$ decreases as $x \rightarrow$ increases.

if $f'(2) > 0$ as $x \rightarrow$ increases.
↓
we don't need this.

our objective is to minimize.

Formula:

newvalue = old value − slope × learning rate.

slope ↓ (−ve) increases.

(+ve) decreases.