Object Optimization for AI and ML Date / /

Tutor

يوجد اسئلة بتيجي عادة عن الـ optimization

أول حاجة بندخل الـ data بجدول، استعمل الـ model ايش الي ML ← Classification, Regression ← ML

الـ data اللي بنجيبها بتكون عبارة عن samples فقط.

$x_1, x_2, \dots, x_n$ | y

n observations

العلاقة بين الـ I/P و الـ y → ML problem ← $\hat{y} = f(x_1, x_2, \dots, x_n)$

مربوطة بـ parameters

I/P: features, independent variable

y: target label, dependent variable

$$\hat{y} = \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

كل ما الـ parameters بتزيد الـ Dimension بتاع الـ model بيزيد

الـ ML ما هو إلا إن احاول أجيب الـ [illegible] ($\theta_0, \theta_1, \dots, \theta_n$ parameters)

الي شكل الـ model المعين لا يمكن يتمثل بالمعادلة linear زي معادلة Logistic regression

$$z = \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n \xrightarrow[\text{nonlinear}]{} \hat{y} = \frac{1}{1+e^{-z}} \leftarrow \text{sigmoid function}$$

ممكن تبقى الـ data كبيرة جداً والعلاقة بين الـ $\hat{y}$ والـ $x_1, x_2, \dots$ complex فالـ ML models العادية مش هتقدر تبني العلاقة وهتحصل overfitting فنستخدم Neural Network

أو الـ Deep N.N

$$z = b + w_1 x_1 + w_2 x_2 + \dots + w_n x_n$$

$$\hat{y} = g(z) = \frac{1}{1+e^{-z}} \rightarrow$$

حاول أجيب الـ parameters $b, w_1, w_2, \dots, w_n$ التي تبني أحسن model

دورنا نجيب $\hat{y} = f(x_1, x_2, x_3)$ التوصل للعلاقة

الـ learning ← إني أجيب العلاقة بين الـ features والـ $\hat{y}$ (الـ parameters)

→ How to find the parameters → في مرحلة الـ training عندنا طرق كتير نجيب

الـ parameters دي.

$x_1 \quad x_2 \quad y$ → actual target | $\hat{y}$ [illegible] function $\xrightarrow{}$ $h_\theta(x)$ hypothesis w, b

حل

x_1	x_2	y	$\hat{y}$
5	1	100	100
6	2	150	150
–	–	–	

$\boxed{h_\theta(x) = \theta_1 x_1 + \theta_2 x_2}$ → الـ model ده بعد ما أجيب الـ θ_1, θ_2 هيبقى

$5\theta_1 + \theta_2 = 100$

$6\theta_1 + 2\theta_2 = 150$

Can be solved Analytically

$$\begin{pmatrix} 5 & 1 \\ 6 & 2 \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} = \begin{pmatrix} 100 \\ 150 \end{pmatrix}$$

الـ model كويس جداً. ده بيسمى high variance

يعني حصلوا حاجة اسمها overfitting

[illegible] الـ model بتاعي

فبنروح لحاجة اسمها الـ Approximate solution

Object Date / /

$m > n$ / $m < n$ — الدالة معظمها بتضمن كذا

مزة الـ Approximation انو بيقلل الـ error في كلو.

$ML / DL \rightarrow$ model parameters — دائماً الـ solution بتكون approximate كاوز قيمة الـ

humble error — لتقليل الـ $|\bar{Y} - Y|$ أقل ما يمكن — approximate (حل) بتحسب الـ

$m > n \rightarrow$ — أقدر أحسبها الـ parameters عن طريق Algebraic solution عن طريقة

الـ least square method أقدر أحسب الـ θ التخليل الـ error أقل ما يمكن

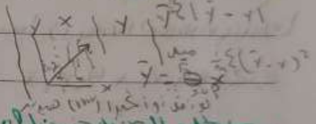
من خلال $\theta = (X^T X)^{-1} X^T Y \leftarrow X\theta = Y$ — عشان إن الـ X مش invertible فيحتاج الـ approximation — التحليل الـ error أقل ما يمكن

$m < n \rightarrow$ least norm method $\theta = X^T (X X^T)^{-1} Y$

في حالة الـ dimension بتاعت كبيرة — Normal Equation $\rightarrow$ لو

بيسبب الـ inverse — مشكلة في الـ expensive & very الـ linear بيقلوا مش مناسب للـ numerical solutions — مشروع

الـ numerical solution — عشان أ find الـ parameters لازم أستخدم



التقليل الـ error prediction أقل ما يمكن وصوصها

مفاهيم الـ optimization ومن خلال numerical methods

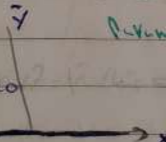
الخطوات :- المهمين كلها عشان أستخدم numerical techniques عشان نحل أي مشكلة

و optimization الـ problem — $\min ML(\theta) = 0$ — Assume θ — قيم مختلفة

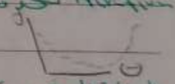
1) Initialization (Assume solution of parameters $\theta_1, \ldots, \theta_n$)

لازم من خلاله طريقة رياضية أقدر أعرف من خلالها أفترض الـ parameters بشكل عشوائي وأبدأ

$\theta = 0$ — مستخدمة أولى للوصول لـ best value للـ parameters



2) $h_\theta(x) = \bar{Y} = \theta X \rightarrow \bar{Y} = 0$



3) evaluation prediction — cost function — الدالة اللي بتحسب كذا اسمها الـ

loss function, error function — . وأنا عاوز أحسب الـ average error على الـ observations كلها

$J \left(\begin{array}{l} MAE \leftarrow \frac{1}{m}\sum |\bar{Y} - Y| \\ MSE \leftarrow \frac{1}{2m}\sum (\bar{Y} - Y)^2 \end{array} \right.$ لكل observation — عندنا نوعين مشهورين للـ error function

وأنا بحسب واحدة بين منهم وده هيكون بناءً على حاجة هنقولها قدام

$J(\theta)$ — ~~بيعتمد على الـ~~ نقدر نرسم العلاقة بين الـ θ والـ J — الاشتقاق الجزئي الأول

لكل parameter — $\frac{dJ}{d\theta} > 0$ — $\frac{dJ}{d\theta} < 0$

4) Direction of change and Amount of change — حاجة تعرفني أزود الـ θ ولا أقللها عشان أوصل للحل المثالي

وكمان أزودها لحد فين. — gradient descent

5) update θ :- $\theta_{t+1} = \theta_t + \text{Amount} \rightarrow$ update equation

6) لنقل من الخطوة رقم 2 تاني لحد ما نوصل لأحسن حل iteration

← لازم يبقى في حاجة تقولي إني وصلت عند الـ θ_{opt}

Object   Date   /   /

في ال Numerical بنروح للـ iterations علشان نوصل لحد ما نوصل لل ... اللي أنسب حاجة هو ال iterations

لازم احاول اوصل لل $\min_{error}$ في أقل عدد من ال iterations (أسرع وقت، أقصر طريق)

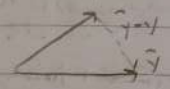
واستفيد من اني بستخدم ال Numerical دة.

اشهر ال cost function الموجودة في ال Regression هما (MSE, MAE)

$y \quad \hat{y} \quad |\hat{y}-y| \quad (\hat{y}-y)^2$

e^2

$|\hat{y}-y| = |1+1+1+1 \dots|$ → L1 norm



$MAE = \frac{L_1 norm}{m}$

$L_2 norm = \left(\sqrt{e_1^2+e_2^2+\dots+e_m^2}\right)^2$

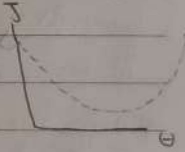
احنا هنا بنتعامل مع ال data فيها ...

$MSE = \frac{L_2 norm^2}{m} = \frac{S.E}{m}$

$J(\theta) \quad |\hat{y}|$

بنستخدم ال MSE أحيانًا وليه بنستخدم ال MAE أحيانًا

$MSE = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}^{(i)}-y^{(i)})^2 = \frac{1}{m}\sum_{i=1}^{m}(\theta^T x^{(i)} - y^{(i)})^2$



$MAE = \frac{1}{m}\sum_{i=1}^{m}|\theta x^{(i)} - y^{(i)}|$

بيتأثر أكتر لو الداتا فيها outlier

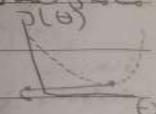
عادةً ال MSE فيها مميزات كتير هنتكلم عنها قدام.

لازم نعرف ايه هو ال Gradient $\boxed{\frac{df}{dx}}$ → Direction of update

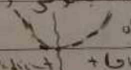
1st derivative → Amount of change (sensitivity of change)

مدى حساسية ال J بتغير ال θ

$J(\theta) = \boxed{}$ دي ال ...



$\frac{dJ(\theta)}{d\theta} = 0 \Rightarrow \nabla = 0$ دة معناه انك وصلت لل $\theta_{optimum}$ ←



Direction of update in the opposite direction of gradient

How to stop. دة كدا هو يعني

$\frac{dJ}{d\theta}$ → −Gradient   +Gradient $\frac{dJ}{d\theta}$

optimum

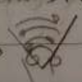
$\theta_{t+1} = \theta_t - (\text{Amount})$

$\theta_{t+1} = \theta_t \pm \left(\frac{dJ}{d\theta}\right)_t$ Gradient ±

MSE function مميزاتها

كل ما ال update ال Gradient بيقل، فحجم ال update كل ما بيقرب من ال optimum بيقل

ال MAE ممكن يحصل ... → وعند ال min النقطة غير قابلة للتفاضل



Object Date / /

diverge يحصل

$\theta_{t+1} = \theta_t - \left(\frac{\partial J}{\partial \theta}\right)_t$ كلما (بقيمة الـ) update يمر الـ

فالاغلب سيكون اضم مطشر كدة عملنا overshoot بيحصل لما

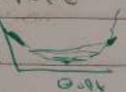
يكون update كل مشكلة وفيه θ_{opt} الـ عديا قيم الـ update لما للـ θ

diverge ويحصل او ما يتنقل الى min ويبعد عن الـ initialize كثير من الاقلية، وكان يحصل

$\theta_{t+1} = \theta_t - \alpha \left(\frac{\partial J}{\partial \theta}\right)_t \leftarrow$ $0 < \alpha < 1$ بيكون كبيرة $\frac{\partial J}{\partial \theta}$ بيستبدل

بيه في التحكم بيكون hyperparameter

training الـ معدل learning rate

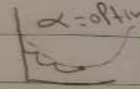


الـ α بيكون قيمة صغيرة update الـ

الكثير Iteration الـ في المشكل يقرب للـ min المرادي

$\alpha = 1$ or very large numb < 1 → overshoot and diverge

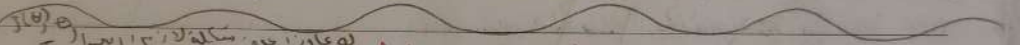
Iteration كثير ولكن كتير → overshoot and converge لو خلينا



$\alpha = $ optimum

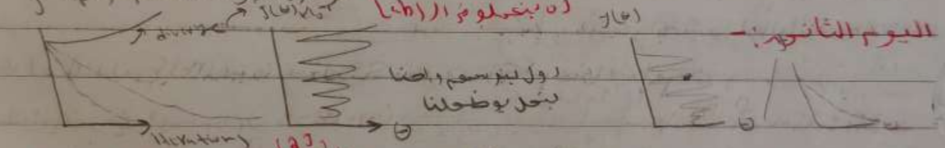
بيختار الـ α بقيمة معينة بحيث الـ update يمشي بشكل افضل

ولو قللنا الـ α اكتر من كدة هاخد عدد Iteration كتير

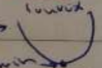


$J(\theta)$, θ

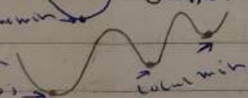
اليوم الثاني :-



$\theta_{t+1} = \theta_t - \alpha \nabla\!\left(\frac{\partial J}{\partial \theta}\right)_t (\theta_t) \rightarrow$ update Equation named (Gradient Decent)



وده Guarantee find local min ← Global min = local min



→ non convex function.

معظم الـ Models في الـ Loss function بتعها convex لما فالـ Gradient بيلاقي حلو → ML

الـ update بيتغير على حسب نوع الـ Algorithm المستخدم معادلة $\theta_{t+1} = \theta_t - \alpha \nabla J$ function

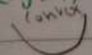
في معظم الـ ML الـ fn $\leftarrow J(\theta) = \frac{1}{m} \sum_{i}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

لو كان الـ Model ابي لوكان linear حتى $\frac{1}{1+e^{-\theta x}}$ ← دوال توضيحية فيستخدم cost function

MSE في دالة

Logistic function (Nonlinear) بيستعملوا الـ الـ Logistic function) الدالة اللي مبتدأ دوال الحد

بتاع الـ α بتتحول الى convex

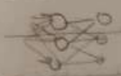


بدو.

اي كان الـ Model المستخدم في الـ ML بيستخدم معادلة نوع الـ cost function في الاخر بنبغي

الحلومة بين الـ J الـ convex فالـ Gradient decent تلاقي حلو

Object Date / /



المشكلة في الـ D.L → $X_{input}^{[1]} \to Z \to A$

$A^{[1]} = f(x\,w^{[1]} + b) \quad A^{[2]} = f(A^{[1]} w^{[2]} + b^{[2]})$

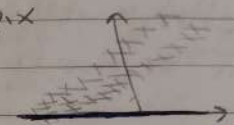
$\hat{y} = h_{w,b}(x) \longrightarrow$ very complex nonlinear

آ الـ loss function هتستخدمها هتبقى non-convex فـ الـ Gradient descent مبيجيش كويس.

المهم عاوز اجيب الـ parameters اللي يجيبلي الـ error الـ min . مش فارق معايا شكلها ولا اتجاهها.

x	x	$\hat{y}$

model has 2 parameters $h_\theta(x) = \theta_0 + \theta_1 x$ (bias)

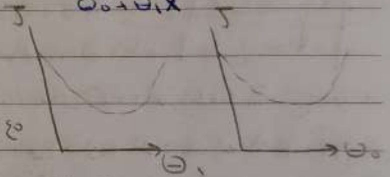


1) assume $\theta_0 = 0,\ \theta_1 = 0 \quad \alpha = r$

2) $h_\theta(x) = 0$

3) evaluate error $J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$ — $\theta_0 + \theta_1 x$

4) $\left(\frac{dJ}{d\theta_0}\right)_t \ , \ \left(\frac{dJ}{d\theta_1}\right)_t$



$\nabla = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \end{bmatrix}$ الوقف لما $\|\nabla\| = 0$

5) $\theta_{0_{t+1}} = \theta_{0_t} - \alpha \left(\frac{dJ}{d\theta_0}\right)_t$

$\theta_{1_{t+1}} = \theta_{1_t} - \alpha \left(\frac{dJ}{d\theta_1}\right)_t$ مع بعض

كل واحد ماشي مع نفسه ممكن واحد يوصل للـ min و واحد للسه هيفضل يتحرك اكتر

بعد ما نعمل iteration لحد ما نوصل عند الـ min ونجيب θ_0 و θ_1

لحد امتى stop ؟ يعني لما θ_0 نوصل للصفر ولا θ_1 ولا الاتنين مع بعض.

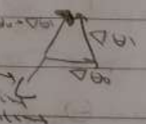
$\|\nabla\| = \text{slope}$

تذكير: ~~Gradient Vector~~

~~$J(\theta_0, \theta_1) = \theta_0^2 + \theta_1^2$~~

multivariable

Gradient $= \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \end{bmatrix}$ $G = \nabla\theta_0 + \nabla\theta_1$



$\nabla\theta_1$ $\nabla\theta_0$ Gradient Direction

لو عندي $X_1, X_2 \ldots X_n$ $\quad \nabla\theta = \begin{bmatrix} \partial\theta_0 \\ \vdots \\ \partial\theta_n \end{bmatrix}_{(n+1)}$

نقدر نقول الـ loss function بتاعتنا إزاي ← من الـ dimension بنرسم كل واحدة لواحدها θ_0, θ_1

لحد كدة علشان لو عندي n من الـ dimension بنرسم مش هنعرف نرسم كلهم في الـ θ (n+1) الـ

يجب $\left(\frac{\partial J}{\partial \theta_0} \text{ و } \frac{\partial J}{\partial \theta_1}\right)$ إعادة تغيير الـ Alpha

In L. Regression → $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y} - y)^2$

عم عشوائي؟! →

مش عارف إيه مكان الـ θ_{opt}

$\frac{\partial J}{\partial \theta_0} = \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})$ مش هيتغير

مكانها فين ← θ_{opt}

$\frac{\partial J}{\partial \theta_1}$ ✓ in Implementation $\nabla \theta = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \end{bmatrix}$

$\theta_{0,t+1} = \theta_{0,t} - \alpha \left(\frac{dJ}{d\theta_0}\right)_t$ [illegible] لو موقفش بعد [illegible]

$\theta_{1,t+1} = \theta_{1,t} - \alpha \left(\frac{dJ}{d\theta_1}\right)_t$

single variable LR

Multi variable LR

x_1	x_2	x_3	x_n	y

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$

θ as vector $= \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$, $\underset{\text{as vector}}{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

[illegible] الـ x_0 وأدخلها ب 1 وحطيته على الـ feature الـ dimension بتاع θ زي الـ X

$h_\theta(x) = \theta^T x$

Implementation → (mathematical) $h_\theta(x) = [\theta_0 \ \theta_1 \ \theta_2] \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_n \end{bmatrix}$ bias intercept

→ data

$\underset{m \cdot (n+1)}{X} \quad \underset{(n+1) \cdot 1}{\begin{bmatrix} \theta_0 \\ \vdots \\ \theta_n \end{bmatrix}} = \underset{m \times 1}{\hat{y} = h_\theta(x)}$

x_1	x_2	... x_n

1) initialization $\theta_{n+1} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $\alpha = ✓$

2) $h_\theta(x) = X\theta = \hat{y}$

3) $J(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{n} (h_\theta(x)^i - y^{(i)})^2$

4) $\nabla \theta = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \vdots \end{bmatrix}$ vector

$\frac{\partial J}{\partial \theta_0} = \frac{1}{m} \sum$ ___ $x_0^{(i)}$

$\frac{\partial J}{\partial \theta_1} = \frac{1}{m}$ ___ $x_1^{(i)}$

___ $x_2^{(i)}$

In different scale

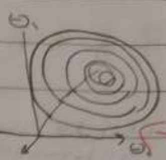
لو الـ [illegible] و training set

عامل زاي ذا .

5) $\theta_{t+1} = \theta_t - \alpha \frac{dJ}{d\theta}$

$[\ ] = [\ ] - \alpha [\ ]$

Object Date / /

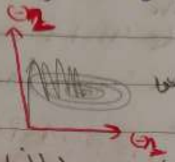


السلواردة بيعتمد على إن ... الـ ... contours في اتجاه Θ_2 و ... contours

في Θ_1 الاتنين قريبين من بعض لإن ... الـ contours ...

متكونش قريبة من بعض و بيكون إن الـ range ... Θ_1 و Θ_2

x_1 (1 → 5) | x_2 (1000 → 2000) | y



مع بعض قريبين من بعض.

لما بيكون عندي الـ feature ... range ... scale

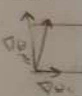
كبير الـ Θ بتاعتها تكون ...

مثلاً: $\Theta_1 x_1 + \Theta_2 x_2 = 100$ الحفاظ عليها.

$$\frac{dJ}{d\theta} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$

فالـ feature الصغير $\frac{dJ}{d\theta} \downarrow$ تقليل ... $\nabla\Theta$ small ... الـ feature

$\nabla\Theta_1$ large



الـ contours ... contours واحد ... كبيرة في اتجاه Θ_2

... Θ ... range قريب من بعض و الـ gradient vector يكون قريب

من بعض فالمشكلة حصلت لما الـ scale بتاع الـ features كان مختلف عن بعض

... features scaling (in the same scale) ...

عندنا ٣ طرق :- على حسب الداتا و الـ problem. لو ... images → Min Max

لو Standardization ... لو outliers ...

→ Min Max Normalization → x_1 | x_2 $\quad \hat{x} = \frac{x - \min(x)}{\max(x) - \min(x)} \quad 0 < \text{range} < 1$

يحافظ على نفس شكل التوزيع الكان عليه.

→ Mean Normalization (Standardization) $\quad \hat{x} = \frac{x - \bar{x}}{\sigma} \quad \mu = 0,\ \sigma = 1$

التوزيع بيتحول إلى Normal Distribution

بشكل الطريقة دي ... لو الداتا فيها outliers

Robust Scaling → $\hat{x} = \frac{x - Q_1}{Q_3 - Q_1}$  ... outliers

+ ... الـ Standardization التوزيع ... بيتحول إلى Normal dist

مع المشكلة ... إن ... Gradient Descent و ... الـ model parameters

... R.L.

... الـ Gradient descent ...

Object   Date / /

1 update = 1 iteration

$J = \frac{1}{2m}\sum$

$\frac{dJ}{d\theta} = \frac{1}{m}\sum$

$\theta_{t+1} = \theta_t - \alpha \left(\frac{dJ}{d\theta}\right)_t$ لما بعمل single update

لما بعمل one update على الداتا كلها summation
all data = batch
update

Batch Gradient Descent ← يعني كمية الداتا ← Batch ← all data ← ML . في استخدمها كلها

ال iteration اللي بتسمى Epoch (لما نستخدم كل الداتا في iter واحدة)

Epoch = use all data in training

1 iteration = 1 Epoch (ال 1 Epoch ممكن يبقى جواه ~~...~~ كذا iteration هنشوفه.)

iter

طب مشكلة ال Batch Gradient Descent اي هي؟؟

1) استخدام كل الداتا علشان نعمل one update مشكلة لإن الداتا اللي بستخدمها ممكن تكون large data

ودة بتسبب computational overhead . لما ... cannot perform كتير.
traffic

2) ال update بيكون very smooth لحد ما نوصل لل minimum فممكن يبطء

حاجة بقا تحل مشاكل ال batch size و تدي عدد اكتر من ال update مع نفس الداتا

ونساعد على ... (Stochastic Gradient Descent)

نعمل update بتضريب observation واحدة طب ازاي؟ (SGD)

1) $\theta_0, \theta_1 = 0$ $\alpha = ?$

x	y	$\hat{y}$

2) $h_\theta(x^{(1)}) = \theta_0 + \theta_1 x^{(1)} = \hat{y}^{(1)}$

3) $J(\theta_0, \theta_1) = \frac{1}{2}(\hat{y}^{(1)} - y^{(1)})^2$

4) $\frac{dJ}{d\theta_0} = (\hat{y}^{(1)} - y^{(1)})$, $\frac{dJ}{d\theta_1} = (\hat{y}^{(1)} - y^{(1)}) \cdot x^{(1)}$

5) $\theta_{0,1_{t+1}} = \theta_{0,1_t} - \alpha \frac{dJ}{d\theta_{0,1}}$ علشان نمشي عكس اتجاه ال Gradient

تاني iteration استخدم تاني observation و ... observation

على عدد الداتا كلها عملت m updates → observations ... θ updates → Batch

1 Epoch $\equiv$ m iterations → ... لحد ما وصلت لل minimum

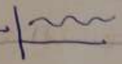
لا فهيبقى كدا ال Epoch ... بعدي على الداتا كلها

... ال update ... لل minimum و ...

بعد ال check ... ال Epoch ... iterations وكل ... ال θ

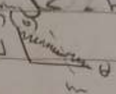
... ال iteration ... ال Epoch ...

Object ↗ Date / /

مشكلتها إن (SGD)  لو مشلدين convergence  noise

الـ gradient كانت بعيدا  طب ما دا بحسب observation x observation

فالموضوع فيه noise على observation لو الـ gradient بياخد في الاعتبار

فمحتاج الـ f(x) function  لحد ما توصل بعد عدد من الـ iterations ونرجع

علشان كدة بعمل check على كل epoch بعد الـ iteration علشان الـ noise

بطلع من المظبوطة بحسب الـ observation إنما بعد كل sm

الـ noise behavior لو ميزة global point

mini batch GD ← batch → All data; SGD → one observation

batch 2 → 32 observation بيقسم الـ traditional

x | y | ŷ

5 mini batch ← batch ← كل mini batch ← فصلها لـ 5 point

1) initialize

$h_\theta(x) = \theta_0 + \theta_1 x$

$J(\theta) = \frac{1}{2b} \sum_{i=1}^{b} (h_\theta(x)^{(i)} - y^{(i)})^2$

$\frac{dJ}{d\theta_0, \theta_1}$

$\theta_{t+1} = \theta_t - \alpha \, d\theta_t$ ← بيستخدم الـ θ الجديدة من أول mini batch

الـ batch الأولى الدور 1 epoch → 5 minibatch

كل ابتداء و الـ batch تاني

مهم إنك تعمل random shuffle للداتا علشان بيحصل للـ observation ~~shuffle~~ ~~mini batch~~

تقلل الـ effect بتاع الـ data collection بتاعها بيدخلها في إحصائيات

الـ behavior بيبقى طبيعي. بنعمل الـ shuffle علشان لو الداتا جاية بترتيب معين

أول 100 صفة والـ 100 التانية كدة فبيعمل إن كسر الـ

بنعمل كده ونعيده في بداية كل epoch.

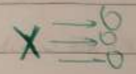
Object Date / /

BGD → batch كويس لو الداتا ال size بتاعها مش كبير

D.L → الأشهر ال mini batch , لما الداتا كبيرة و كمان ال training time

D.L → Advanced optimization for D.L

عدد ال Parameters كبير جدا عندنا

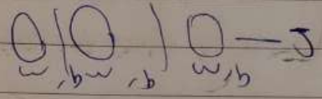


training Deep Neural Network some of training is easy

ML كان ... ازاي طيب ؟؟ فكرتها كلها بعد حساب ال ... جاوز احسبه ال cost

بين ال x وال y بتاعه بان cost function اي كان شكلها اي بصورة بحسب $\frac{dJ}{dw}$, $\frac{dJ}{db}$

كل ال w , b الموجودة في ال N-N



بـ chain rule و بحسل Back Propagation for Gradient

عبارة عن شوية حاجات مضروبة في بعضها $\frac{dJ}{dw} = \frac{dJ}{dA} \cdot G \cdot G$

حاصل ضرب جملة ثوابت

بعد كده $w_{t+1} = w_t - \alpha \frac{dJ}{dw}$

ال مشكلة ← Vanishing Gradient بعد ما بيطلع بضرب ال G في α لو الثوابت

دي كانت اقل من الواحد و كمان ال α اقل من الواحد و دة هيوصلني الى ان ال Gradient صغير

كده ال weight مش هتتحرك وكده هيحصل ... خلا ولو كبرت ال α

دي هيحصل ... Exploding G لان ال $\frac{dJ}{A}$ لو كانت اكبر من الواحد $\frac{dJ}{dw}$ ...

ولما يحصل ... دة الحل وال ... كبير هيحصل diverge

in Vanishing لما بيمشي في نفس الاتجاه , Gain Moment في حالة ال Vanishing (G)

بستخدم تكنيك جديد Momentum based GD ← ازاد سرعة

علشان بدل ما اخد وقت علشان امشي في الجهة ال ... بتاعها قليل لا هاخد عدد iteration

iteration قليل بس في نفس الوقت ... يتحرك ... مش ...

لو بدأ ... Confident الأول اني ماشي في نفس اتجاه ال Gradient طب لو الاتجاه اتغير

يبقى ال Confident بيقل وال momentum بيقل

ال Gradient

كل ما بعمل iteration ولا ... ماشي في نفس الاتجاه ال Confident ... و Gain momentum

Performance الهيكل ال training ... كويس ...

الـ $\nabla\theta_t$ نفس اتجاه الـ Gradient علشان انا جايبه بيزودني في الأول وبعدين قدام
اتجاهه هيبقى بيزود في حاجة تانية

Object momentum → قوة دافعة كبيرة جدا Date / /

$$\theta_{t+1} = \theta_t - \boxed{\alpha \nabla\theta_t}$$ → طب الحل هنا هيكون في الـ momentum بنزود الجزء ده

ولكن الزمان اللي بنبتدي فيه الـ iteration اللي بتكون ماشية في نفس الاتجاه بيتم اضافة الـ $\nabla\theta$ و γV

لو في نفس الاتجاه يزود الجزء ده ←

$$\theta_{t+1} = \theta_t - \{\alpha \nabla\theta_t + \gamma V_{t-1}\}$$

كل ما بيمشي في اتجاه الـ Gradient القيمة بتزيد

$$V_t = \gamma V_{t-1} + \alpha \nabla\theta_t \longrightarrow \theta_{t+1} = \theta_t - V_t$$

momentum term

hyperparameter < 1

1) $\theta_0 = \theta_{1} = 0 \rightarrow V_0 = 0$

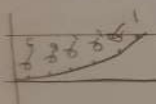
$V_1 = \gamma V_0 + \alpha \nabla\theta_1$ ↳ mean the first iteration هنا

$$\theta_{t+1} = \theta_t - V_t$$

iteration 2 → $\theta_2 = \theta_1 - V_1 \rightarrow \boxed{\theta_2 = \theta_1 - \alpha \nabla\theta_1}$ → first iteration first stage

$$V_2 = \gamma[\alpha \nabla\theta_1] + \alpha \nabla\theta_2$$

<1 جزء من الـ Gradient θ_1 ... كله



$$V_3 = \gamma[\gamma\alpha \nabla\theta_1 + \alpha \nabla\theta_2] + \overset{\text{كاملة}}{\alpha \nabla\theta_3}$$

exponential weighted sum → $V_3 = \gamma^2 \alpha \nabla\theta_1 + \gamma\alpha \nabla\theta_2 + \underbrace{\alpha \nabla\theta_3}_{\text{current}}$

وهكذا ... أقل ... أقل

كل ما بيعدي بيقل الـ Gradient الـ effect بتاعه بيقل

كل نقطة بتاخد جزء من اللي قبلها كتير ... طب ما كل ده علشان أقرب للـ Vanishing

ملحوظة ← لو الـ cost function معقدة فـ Vanishing و استخدم طريقة الفانت وأخذت

قيمة γ كبيرة هيحصل diverge. لو مفيش Vanishing و استخدمت قيمة معقولة

هيعمل update أسرع (بيزود الـ update فيقلل الـ iteration) ولكن استخدمها مع حتة

الـ Vanishing.

في مشكلة في الـ Momentum مشكلته لما بيوصل عند الـ min بسبب اعتماده

على الـ history بشكل كبير فهيزود الـ θ_{min} لحد ما يقرب للصفر ولكن الباقي هيفضل ...

الصحيح الـ Gradient عكس اتجاهه لو موجب Buma $V_t = \{\gamma V_{t-1}\}_{\text{negative}} + \{\alpha \nabla\theta_t\}_{\text{positive}}$

فالـ V_t هيفضل لحد ما يوصل للصفر

الـ ... كل Gradient

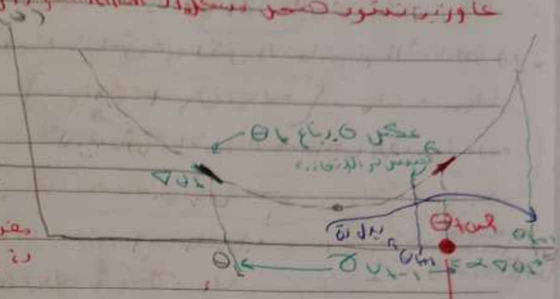
الـ momentum كبيرة بيبعد عن الـ minimum لحد ما يوصل للصفر

Object Date / /

Nesterov Accelerated Gradient ⇐ عاوزين نشوف نحل مشكلة ال oscillations و زاي

$V_t = \gamma V_{t-1} + \alpha \nabla \theta_t$

$\theta_{t+1} = \theta_t - V_t$



عند النقطة دي ه update بأسلوب تاني ⟩

1) $\theta_{temp} = \theta_t - \gamma V_{t-1}$ مفروض دي زي ما فيه ده إتحسب قبل كده

2) $\theta_{t+1} = \theta_{temp} - \alpha \nabla \theta_{temp}$

move allerate | أشوف ال Gradient عند ما عامل إزاي وده الهدف بيتظبط لأن دا ال

3) $V_t = \gamma V_{t-1} + \alpha \nabla \theta_{temp}$

(ابطأ من ال momentum)

الشكل ال Normal equation ←

θ_t update → θ_t NAG update

V_t momentum → V_{NAG}

حسنين كتا ال NAG ميتعلق ال oscillations الحاصل في ال momentum

نفهم الاحسن عن حسيب ال calc نفسها الهيستوغرام

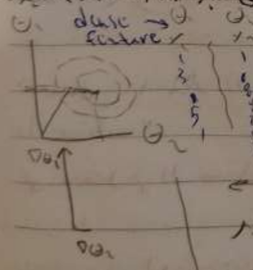
اليوم الرابع (Part 1) فاكر المشكلة دي الكتابة حلها

ال sparse feature هي مش المشكلة دي تظهر ومفروض نحلها بال dense feature

المشكلة تكون ال features دي حاجة بسها

sparse feature ← أغلب قيم الداتا بتكون أصفار ←

ونادرًا لما تظهر رقم



دي حاجات لا نستخدمها ان احنا نشيلها ←

$\nabla\theta_1 = \frac{1}{m}\sum_{i=1}^{m} h_\theta(x^{(i)}) - y^{(i)} \cdot x_1^{(i)}$

$\nabla\theta_2 = \frac{1}{m}\sum_{i=1}^{m} h_\theta(x) - y^{(i)} \cdot x_2^{(i)}$ عدد ال iterations كتير

لأن θ_2 أغلب الداتا بأصفار

مشكلها إن ال θ_1 هتحط weight أكبر لل update بتاع ال θ_1 ووصل ال update

فهي ال component كبير و ال θ_2 صغير ⟸ $\nabla\theta_1$ Gradient $\nabla\theta_2$ فال update هتكون طويل

و ده تتعملوا من خلال ال α.

Object Date / /

Adagrad (Adaptive Gradient)

$\theta_{t+1} = \theta_t - \boxed{\alpha} \nabla \theta_t$ ← effective learning rate

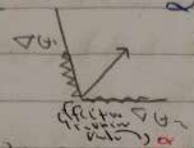
Adaptive Gradient:-

الـ Gradient بقا adaptive هيقاس بضرب في

و دة adaptive بيفاعل حسب الـ θ

الـ Gradient هي adaptive في الـ adaptive بياخد

و دة على حسب الـ α



$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix}_{t+1} = \begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix}_{t} - \begin{bmatrix}\frac{\alpha}{\square}\\ \frac{\alpha}{\square}\end{bmatrix} * \begin{bmatrix}\frac{dJ}{d\theta_0}\\ \frac{dJ}{d\theta_1}\end{bmatrix}$$

learning rate vector

وهيكون بقا في الـ Dense feature ← $\frac{\alpha}{\square \to \text{large}}$ علشان نقلل

في الـ Sparse feature ← $\frac{\alpha}{\square \to \text{small}}$ علشان نزود

الجزء اللي تحت دة مهيتش نفس فيه دة بيمثل مثال في الـ [illegible] اللي بيجي ازاي

$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{v_t}+\epsilon} \nabla \theta_t$ دة غير دة في الـ momentum دة حاجة تانية

$\epsilon = 10^{-8}$ → avoid Division by 0

$v_t = v_{t-1} + (\nabla \theta_t)^2$.

المشكلة المشاكل بتحصل !!!

$v_t = v_{t-1} + (\nabla \theta_t)^2$

في الـ Dense لو كانت قيم الـ ∇ كبيرة و انت عامل تربيع ليها فيها الـ

v_t كبير جدا !! و دة لو حصل هيحصل لو هيكون الـ $\frac{\alpha}{\sqrt{v_t}+\epsilon}$ صغير جدا !!

و دة بنسميه aggressive decay ← learning rate

← هيقال تربيع حاجة كويسة

و دة بيعمل حاجتين الإتنين ١) هنلاقي حصل stall في اتجاه الـ Dense عند نقطة معينة

٢) بنجي بعد الـ [illegible] قريب منو و بيحصل الغي لـ (١) و الاتنين كويسين. ولو حل !!

نكمل. ان انا كل الـ [illegible] parameters [illegible]

$v_t = v_{t-1} + (\nabla \theta_t)^2$

بحساب الـ v_t تتغير weighted الـ history و القيمة الحالية بحساب الـ v_t بيبقى

احسن بكتير

Object RMS Prop Date / /

اليوم الرابع (Part 2) $V_t = \beta V_{t-1} + (1-\beta)(\nabla\theta_t)^2$ / $\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{V_t + \varepsilon}} \cdot \nabla\theta_t$

hyperparameter < 1 $V_0 = 0$

$V_1 = \beta V_0 + (1-\beta)(\nabla\theta_1)^2$ at iteration 1

$V_1 = (1-\beta)(\nabla\theta_1)^2$

$V_2 = \beta V_1 + (1-\beta)(\nabla\theta_2)^2$

EWMA

$\rightarrow V_2 = \beta(1-\beta)(\nabla\theta_1)^2 + (1-\beta)(\nabla\theta_2)^2$

ال accumulation لل V بقاله ميزين ال EWMA للـ gradients

← حل ال Adagrad وال RMS Prop يقدروا يتعاملوا مع ال Gradient Vanishing

ولا لا !!!! وهل ال momentum وال NAG يقدروا يتعاملوا مع ال sparse feature

الاجابة لا ال momentum يعملوا إشارة ال $\nabla\theta_t$ على عكس ال Adagrad وال RMS

يعملوا القسمة. طب اشتغال ال DL سكة تقابل المشكلتين ال sparse feature

مع ال Vanishing ضمح ال Combine الاتنين مع بعض ← Adam

Adaptive moment → momentum + RMS Prop.

بالتحديد ال α ليكون موجود في ال update ← من ال momentum

another version from momentum → $m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla\theta_t$

ADAM RMS Prop → $V_t = \beta_2 V_{t-1} + (1-\beta_2)(\nabla\theta_t)^2$

يقدر يتعامل مع المشكلتين الفاتو $\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{V_t + \varepsilon}} m_t$

Real Application $\beta_1 = 0.9$ $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$, $\alpha \rightarrow 0.0001 \rightarrow 0.001$

(ADAm + mini batch) → الأشهر في DNN

ليه استخدمنا ال momentum ومستخدمناش ال NAG ← ال RMS فحسي

على ال V_t بيحل المشكلة الـ momentum اللي سوية مشكلتو ناع ال oscillation بيشتغل

ولكن موجود بين من دوشر اول اى مكان لوحدة بوفي تكملة NAG → momentum

وبيسمو Adam = NAdam

في بحوث عشان ال m_t وال V_t في البدايات بيوحشوا $\hat{V}_t$, $\hat{m}_t$

الاول دي بيعمل عشان كده ← $\hat{V}_t = \frac{V_t}{1-(\beta_2)^t}$, $\hat{m}_t = \frac{m_t}{1-(\beta_1)^t}$ ال iteration

ال update equation بسبب ال bias

ليه لأن هذا القيمة الحقيقية المفروض تكون محسوبة لأن في نقط مدخلها بداية بسبب قيمة ال β

Object Date / /

كنت بستخدم الـ 1st order optimization (الـ optimization المعتمد على $\frac{dJ}{d\theta}$)

الـ method المعتمد على $\frac{d^2J}{d\theta^2}$ ← 2nd order optimization →

وإي الحكمة وبنستخدم إزاي وبنستخدمش إزاي وعيوبها واستخدمها إمتى!!!

اليوم الرابع (Part 3) الـ Gradient Descent جت منين أصلا!!!

جت من الـ First order Approximation للـ function باستخدام الـ taylor series

Second order optimization ← بنعمل approximation لـ curve فبروح للـ min على طول

اليوم الخامس (Part 4) لو عندي أي function أقدر أعمل approximation عند نقطة واحدة فيها الـ first order ... لـ approximation بعد الـ taylor series بعد الـ ... objective على حسب الـ الحاجات وزوم الـ approximation

فكرة الـ Gradient descent إني أ objective الـ iteration الـ next function تكون أقل من القيمة اللي أنا فيها الـ step اللي خدتها في إتجاه الـ min مفروض أنضمن إن الـ next iteration هي الـ min ، لو $f(x_{k+1}) < f(x_k)$ بالتالي لازم أختار

$$f(x_{k+1}) = f(x_k) + f'(x_k)(x_{k+1} - x_k)$$

لو ده سالب بالتالي $f'(x_k)$ يكون سالب

$\theta_{t+1} - \theta_t$

negative ← $\nabla\theta \cdot \Delta\theta$ لو عكس بعض

$\Delta\theta$ في الإتجاه المعاكس لـ Gradient $\Delta\theta = -\nabla\theta$

$$\theta_{t+1} - \theta_t = -\nabla\theta_t \longrightarrow \theta_{t+1} = \theta_t - \alpha\nabla\theta_t$$

لم بتوصل للـ min بتاع أي function كده بس لازم الـ الـ بعض

مشكلة الـ G.D إنها بتاخد steps صغيرة help من الـ ML بتاع حلوة

فى الـ ML ← ← line كـ approximation بدل شكلها بالمعادلة بتاعتها

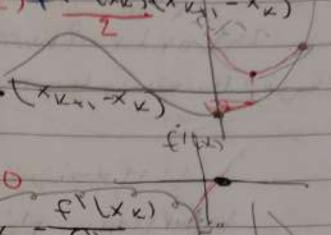
النقطة دي ولو بين الـ curve والـ objective نروح لـ min بتاع الـ save الـ جديد

simple convex fun ممكن نروح بخطوة واحدة و ممكن ومن عيوبه ج learning فالـ theoritical value

Object Date / /

$$f(x_{k+1}) = f(x_k) + f'(x_k)(x_{k+1} - x_k) + \frac{f''(x_k)}{2}(x_{k+1} - x_k)^2$$

objective → x_{k+1} ⟹ $f'(x_{k+1}) = 0$ شرط



$$f'(x_{k+1}) = 0 + f'(x_k) + \frac{2}{2} f''(x_k)\cdot(x_{k+1} - x_k)$$

$$f'(x_k) + f''(x_k)\cdot(x_{k+1} - x_k) = 0$$

$$x_{k+1} - x_k = \frac{-f'(x_k)}{f''(x_k)} \longrightarrow x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

احنا كدا بنحل المعادلة بطريقة التفاضل دي وكدا Newton Raphson

كدا بنحلها Newton Raphson .

في مشكلة في الموضوع ده !!!! يعني $f'(x_{k+1}) = 0$ → min / max / saddle

Newton ← بيروح للـ saddle point فيروح للأقرب لو كان الأقرب max مش الحل

فبنستخدمه مع الـ convex function بس وكده تمام زي الفل

فبنستخدمه في الـ ML الـ convex و بنستخدمها في الـ D.L (second order optimization)

الـ Newton في الـ formulation الخاصة بالـ (Quasi Newton) شبه

$$\theta_{t+1} = \theta_t - \frac{J'(\theta_t)}{J''(\theta_t)}$$

$$\begin{bmatrix}\theta_0\\ \theta_1\\ \theta_2\\ \theta_3\\ \theta_n\end{bmatrix} = \begin{bmatrix}\theta_0\\ \theta_1\\ \theta_2\\ \theta_3\\ \theta_n\end{bmatrix} - \nabla f(x_k) \cdot H^{-1}(x_k)$$

$\nabla f(x_k) = \begin{pmatrix}\frac{\partial J}{\partial \theta_0}\\ \frac{\partial J}{\partial \theta_1}\\ \vdots\end{pmatrix}$, $n\times n$

المشكلة ١) $H \to +ve$ ... indefinite saddle

٢) كبيرة (nonparametric method) بسبب مشكلة كبيرة وهي

لو زودت features كتير وليكن الـ ∇ هيبقى ... $O(n)$ $O(n^2)$

هتوصل الـ $H_{n\times n}$ والـ inverse كتير → $O(n^3)$ computation expensive كبيرة جداً.

اليوم الخامس (Part 2) ومن عيوب الـ Newton method إن الـ convex fn يحصل الـ ... على مثال (1)

تتغير أسرع لإن الـ Newton method الـ time complexity بتاعها كبير جداً وده بيشكل عيب وده

بسبب الـ حاجات اللي بنحسبها كلها بتدخل في حسابات كتير H^{-1}, H, g $O(n^3), O(n)$

→ Quasi Newton method $x_{k+1} = x_k - \boxed{H^{-1}(x_k)} \cdot \nabla(x_k)$ ← Newton

$B^{-1} - \boxed{B}$ ← H ل Approximation هنشيلها ونحط الـ

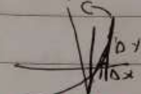
← لما بنعمل approximation هنتعامل لـ H نجيب الـ B وبعدين ناخد الـ inverse

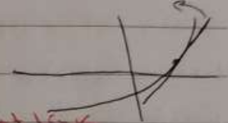
أحنا هنعمل الـ approximation من الـ H^{-1} ← للوصول فبنستخدم B^{-1} عشان time complexity عالية

وده عن طريق اللي بيستفيد من الـ Previous information وده بيسبط حساباتنا Secant Method

$\frac{\Delta y}{\Delta x}$ ← Secant Line لاني من الـ tangent في Derivative . كل ما قلل الـ Δx كل

الـ derivative approximation بيكون أحسن



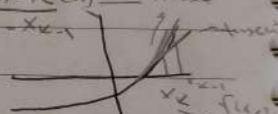


$\{$ approximation Derivation secant line
فكرة الـ secant method
$\{$ derivative tangent line

$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$ approximation كل f' من الحاجات السابقة الحسيتها لحسبنا

$f'(x_k) \leftarrow \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$ — secant

من محتاج الـ tangent هنعمل على الـ secant والنقاط السابقة



تحديد

$x_{k+1} = x_k - f(x_k) \cdot \boxed{\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}}$

$f''(x_k) = \frac{f'(x_k) - f'(x_{k-1})}{x_k - x_{k-1}}$

مثال (2) f.D هنعتبر محتاج الـ derivative secant فحاجة بقا تعمل approximation

$f'(x_k) \cdot (x_k - x_{k-1}) = f'(x_k) - f'(x_{k-1})$

multi variable (2) ← secant condition for n dimension $B(x_k)(x_k - x_{k-1}) = \nabla f(x_k) - \nabla f(x_{k-1})$ vector ← Secant condition

Approximation for H^{-1}

$B_k \, \Delta x_k = y_k \longrightarrow \Delta x_k = B_k^{-1} y_k$

$B_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\Theta_0 = \{ \}$, $\Theta_{-1} = \{ \}$

B $\begin{bmatrix} x_1 & x_2 & x_3 \\ x_2 & x_4 & x_5 \\ x_3 & x_5 & x_6 \end{bmatrix} \begin{bmatrix} ? \\ ? \\ ? \end{bmatrix}$

$n(n+1)$

وعدد المعادلات n^2 تقريباً unknown عدد الـ ... بيكون الـ unknown

عدد الـ $(n^2-(n-1))$ اقل من عدد الـ unknown

محتاج أضيف condition علشان تبقى متساوية بحيث

يعوض ان المعادلات 6 equation و 6 unknown

$\frac{n(n+1)}{2} = \frac{3 \cdot 4}{2} = 6$ known

Object ....

Date / /

ضروري عليه ... positive

Quazi Method → $x_{k+1} = x_k - B_k^{-1} \nabla f(x_k)$

كل نوع بيختلف عن التاني في كيفية تحقيق Property / Constraint

$Bx = y$ →

نوع الـ Quazi المستخدمة في التحديث نوع الـ Constraint [illegible]

BFGS → بيحط مجموعة من الـ Constraints علشان يحل المعادلات