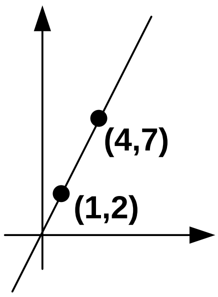
Introduction

在一個多維空間中，一個點就代表一筆資料
e.g. 一個人可由他的身高、體重、年齡…來記錄，一個人就有一組數據，可在一個(身高、體重、年齡、…)的多維空間中以一點表示。

以一個簡單的二維空間來做開場白，有兩組資料(1,2),(4,7)，y 為 target，x 為 input data，我們可以找出一條最能代表這兩點資料趨勢的直線，要代表這條直線，我們需要參數來代表這條直線，此處我們將 input data 集合稱為 A，target 集合稱為$\vec{b}$，我們想要求的直線參數為$\vec{x}$



令$y = ax + b$

$7 = 4a + b$

$2 = a + b$

$$\Rightarrow \underset{A}{\begin{bmatrix} 4 & 1 \\ 1 & 1 \end{bmatrix}} \underset{\vec{x}}{\begin{bmatrix} a \\ b \end{bmatrix}} = \underset{\vec{b}}{\begin{bmatrix} 7 \\ 2 \end{bmatrix}}$$

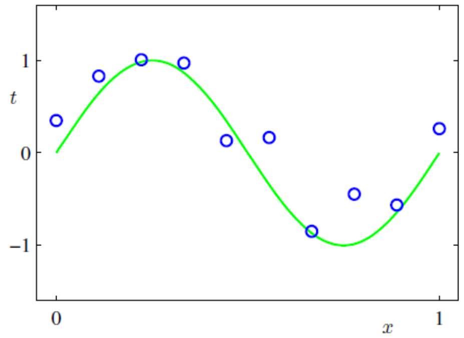
若要解此問題，可使用 Gauss Jorden elimation

$$\left[\begin{array}{cc|cc} 4 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{array}\right] \rightarrow \left[\begin{array}{cc|cc} 1 & 0 & \frac{1}{3} & \frac{-1}{3} \\ 0 & 1 & \frac{-1}{3} & \frac{4}{3} \end{array}\right]$$

$$\Rightarrow A^{-1} = \begin{bmatrix} \frac{1}{3} & \frac{-1}{3} \\ \frac{-1}{3} & \frac{4}{3} \end{bmatrix}$$

但這種方法太花時間 O(n^3)，可使用 LL 或是 LU 分解來做比較快(HW)

以下的討論都是以 $f(x) = \sin(2\pi x)$, $0 \le x \le 1$來討論



在現實生活中，我們得到的資料通常不可能是完美的，通常都會有很多的原因導致誤差，e.g. 得到的資訊不夠多、判斷誤差、量測誤差...，就像上面那張圖一樣，我們觀測一個函數，得到九個數據，但我們並不知道這個函數是甚麼。如果沒有誤差，我們得到的資料會剛好在 sin 函數上，要推得正確的函數應該是不難的是，但是由於誤差的存在，我們沒辦法直接得到正確的函數。

假設我們有 N 筆數據，x 為我們已知的參數，t 為我們希望能藉由已知參數模擬出來的參數，舉例來說，若我們想租房，我們有一筆預算，但我們想知道到底這樣的價錢能租到多少坪數的房間，我們就會想知道一間房間坪數多寡和租屋價錢的關係，此時 x 就是房間坪數，t 是租屋價錢。但是這種關係不可能完美套用在每筆租屋訊息中，我們得到的資訊不夠多(可能還會受屋齡、樓層、屋主個性等等影響)，所以會有誤差存在，但總體而言，還是應該會有一個趨勢存在，也就是我們有興趣的。

整理一下參數

$$\mathbf{x} \equiv [x_1 \;\; x_2 \;\; \ldots \;\; x_N]^T$$
$$\mathbf{t} = [t_1 \;\; t_2 \ldots \;\; t_N]^T$$

粗體代表向量，正常粗細代表純量

其中一種預測方法就是使用多項式(polynomial basis function)來試圖解釋輸入輸出的關係

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

對於 **w** 來說，此方程式是線性的，故又稱為 linear method

假設我們使用 quadratic form 表示 y，有兩組資料

$$\begin{bmatrix} x_0^{\,2} & x_0 & 1 \\ x_1^{\,2} & x_1 & 1 \end{bmatrix} \begin{bmatrix} c \\ a \\ b \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \end{bmatrix}$$

(note: 這個形式請深深烙印在腦中，以後會一直用到)

我們可以將此做衍伸

$[\phi_0\ \phi_1 \cdots][\vdots] = \begin{bmatrix} y_0 \\ y_1 \end{bmatrix}$

$\Rightarrow \phi\vec{x} = \vec{b}$

ϕ 其實是我們能自行定義的，在這只是舉個例，其實 basis function 不一定要使用 polynomial，如果你想，你可以創造任意的 basis function

LSE

最後，假設我們已得到多組 **w**，要評估哪個是最好的方法，通常用 least square error(LSE)，也就是每個實際數據和預測數據相減，平方後再全部一起加總

$E(\mathbf{w}) = \min_{\mathbf{w}} \{y(x_n, \mathbf{w}) - t_n\}^2$ (課本)　$\min \sum (f(x_i) - y_i)^2 = \min \| A\vec{x} - \vec{b} \|^2$ (上課)

$$\min \| A\vec{x} - \vec{b} \|^2 = (A\vec{x} - \vec{b})^T (A\vec{x} - \vec{b}) = \begin{bmatrix} (ax_0 - b - y_0)^2 \\ (ax_1 - b - y_1)^2 \\ \dots \end{bmatrix}$$

$\min \| A\vec{x} - \vec{b} \|^2 = (A\vec{x} - \vec{b})^T (A\vec{x} - \vec{b}) = (\vec{x}^T A^T - \vec{b}^T)(A\vec{x} - \vec{b}) = \vec{x}^T A^T A\vec{x} - \vec{b}^T A\vec{x} - \vec{x}^T A^T \vec{b} + \vec{b}^T \vec{b}$

$\because \vec{b}^T A\vec{x}$和$\vec{x}^T A^T \vec{b}$皆為純量，取轉置後值不變

$\vec{b}^T A\vec{x} = (\vec{b}^T A\vec{x})^{\mathrm{T}} = \vec{x}^T A^T \vec{b}$

$\therefore \min \| A\vec{x} - \vec{b} \|^2 = \vec{x}^T A^T A\vec{x} - \vec{b}^T A\vec{x} - \vec{x}^T A^T \vec{b} + \vec{b}^T \vec{b} = \vec{x}^T A^T A\vec{x} - 2\vec{x}^T A^T \vec{b} + \vec{b}^T \vec{b}$

令$f = = \vec{x}^T A^T A\vec{x} - 2\vec{x}^T A^T \vec{b} + \vec{b}^T \vec{b}$

要找 x 使得 f 最小的話，我們將 f 微分=0，解 x

推$\dfrac{d(\vec{x}^T A^T A\vec{x})}{d\vec{x}}$

$$令A^{\mathrm{T}}A=G, G為對稱矩陣= \begin{bmatrix} g_{11} & g_{12} & g_{13} & \cdots & g_{1n} \\ g_{21} & g_{22} & g_{23} & \cdots & g_{2n} \\ g_{31} & g_{32} & g_{33} & \cdots & g_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ g_{n1} & g_{n2} & g_{n3} & \cdots & g_{nn} \end{bmatrix}$$

$$\frac{d(\vec{x}^T G\vec{x})}{d\vec{x}} = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} \\ \dots \\ \frac{\partial}{\partial x_n} \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_n \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} & g_{13} & \dots & g_{1n} \\ g_{21} & g_{22} & g_{23} & \dots & g_{2n} \\ g_{31} & g_{32} & g_{33} & \dots & g_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ g_{n1} & g_{n2} & g_{n3} & \dots & g_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_n \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} \\ \dots \\ \frac{\partial}{\partial x_n} \end{bmatrix} \begin{pmatrix} x_1(g_{11}x_1 + g_{12}x_2 + \dots + g_{1n}x_n) + \\ x_2(g_{21}x_1 + g_{22}x_2 + \dots + g_{2n}x_n) + \\ x_3(g_{31}x_1 + g_{32}x_2 + \dots + g_{3n}x_n) + \\ \dots \\ x_n(g_{n1}x_1 + g_{n2}x_2 + \dots + g_{nn}x_n) \end{pmatrix}$$

$$= \begin{bmatrix} (g_{11}x_1 + g_{12}x_2 + \dots + g_{1n}x_n) + (g_{11}x_1 + g_{21}x_2 + \dots + g_{n1}x_n) \\ (g_{21}x_1 + g_{22}x_2 + \dots + g_{2n}x_n) + (g_{12}x_1 + g_{22}x_2 + \dots + g_{n2}x_n) \\ (g_{31}x_1 + g_{32}x_2 + \dots + g_{3n}x_n) + (g_{13}x_1 + g_{23}x_2 + \dots + g_{n3}x_n) \\ \dots \\ (g_{n1}x_1 + g_{n2}x_2 + \dots + g_{nn}x_n) + (g_{1n}x_1 + g_{2n}x_2 + \dots + g_{nn}x_n) \end{bmatrix} = \begin{bmatrix} (g_{11}x_1 + g_{12}x_2 + \dots + g_{1n}x_n) \\ (g_{21}x_1 + g_{22}x_2 + \dots + g_{2n}x_n) \\ (g_{31}x_1 + g_{32}x_2 + \dots + g_{3n}x_n) \\ \dots \\ (g_{n1}x_1 + g_{n2}x_2 + \dots + g_{nn}x_n) \end{bmatrix} + \begin{bmatrix} (g_{11}x_1 + g_{21}x_2 + \dots + g_{n1}x_n) \\ (g_{12}x_1 + g_{22}x_2 + \dots + g_{n2}x_n) \\ (g_{13}x_1 + g_{23}x_2 + \dots + g_{n3}x_n) \\ \dots \\ (g_{1n}x_1 + g_{2n}x_2 + \dots + g_{nn}x_n) \end{bmatrix}$$

$$= G\vec{x} + G^T\vec{x} = 2G\vec{x} \ \ (\because G \text{ is symmetric})$$

推$\frac{d(\vec{x}^T A^T \vec{b})}{d\vec{x}}$是一樣的方法，$\frac{d(\vec{x}^T A^T \vec{b})}{d\vec{x}}$=$A^T\vec{b}$

$$\frac{d(\vec{x}^T A^T \vec{b})}{d\vec{x}} = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} \\ \dots \\ \frac{\partial}{\partial x_n} \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_n \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \dots \\ b_n \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} \\ \dots \\ \frac{\partial}{\partial x_n} \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_n \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \dots \\ h_n \end{bmatrix}$$

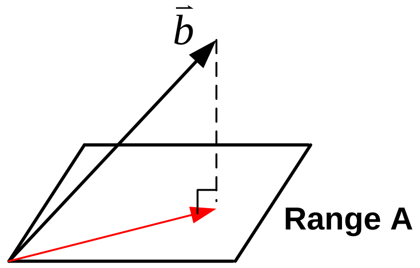
$$= \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \dots \\ h_n \end{bmatrix} = A^T\vec{b}$$

$$f' = 2A^T A\vec{x} - 2A^T\vec{b} = 0$$

$$\Rightarrow A^T A\vec{x} = A^T\vec{b}$$

$$\vec{x} = (A^T A)^{-1} A^T \vec{b}$$

也可以用幾何方式解釋：



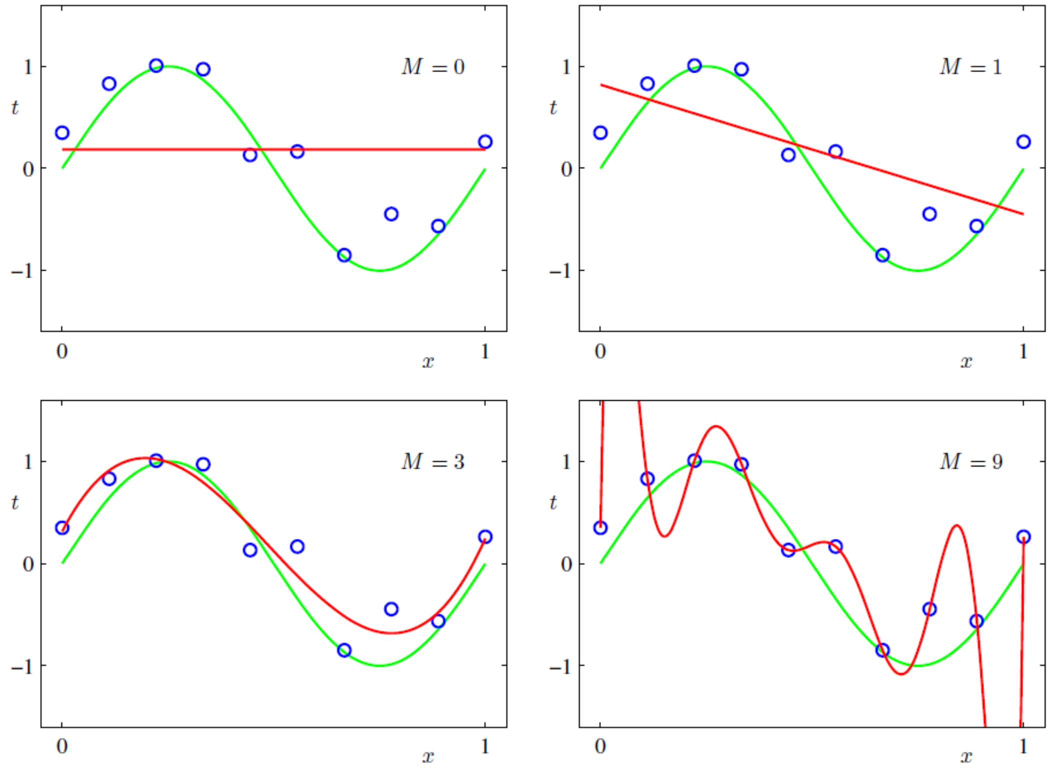
$$< \vec{b}-A\vec{x}, A >= A^T(\vec{b}-A\vec{x}) = 0$$
$$\Rightarrow A^T\vec{b} = A^T A\vec{x}$$
$$\Rightarrow \vec{x} = (A^T A)^{-1} A^T \vec{b}$$

但由於 $A^T A$ 只保證半正定(Semidefinite)，只能保證 $\det(A^T A) \geq 0$ ，故無法保證 $A^T A$ 可逆，如何解決稍後會說明

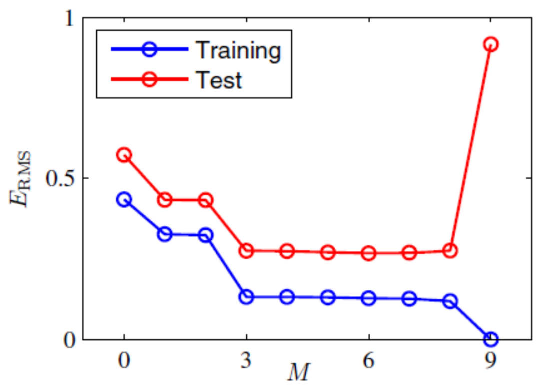
有兩個關鍵步驟要做(假設是 basis function 是 polynomial)

1. 決定 $\mathbf{w}$(決定每項的參數)　　2. 決定次方(決定 basis function 的形式)

1. 由於 LSE 是二次多項式，要找極小值對其做一次微分， $E'(\mathbf{w})$ 為一次多項式，故解 $E'(\mathbf{w}) = 0$ 必定有惟一解
2. 這裡只 show 出次方太大或太小會是甚麼情況，同樣一個例子，我們有 9 筆 data，下圖中可看到，M=0,1 時都無法完整表達此函式，此時稱為 underfitting，代表我們的 model 不足以描述 data 的特性，誤差太大。但是若是次方太大，M=9 時，反而是太詳細，也就是我們把問題想得太複雜了，其實實際問題並沒有那麼複雜，而因為我們希望 error 能最小，M=9 時，會有 8 個轉折點，此時可保證所有點都在我們預估的 function 上，但是只保證我們取的 data point 上是正確的，如果我們再從 sin 函式取一個點出來，誤差就會很大，這種狀況我們稱為 overfitting



我們隨機取 100 筆數據，用 RMS(不知道的話不用在意 RMS 是甚麼，就是計算 error 而已)，算總體的 error，結果如下



training 是從 sin 函式中取點(training data)，用這些點來找參數，再用這些參數代表的 model，算出這些點的 LSE
test 是從同一個 sin 函式取點，不過和 training 拿的點是不同的(test data)，用 training 找出的參數，算出 test data 的 LSE

我們會很好奇的是，其實 9 次方也能拿來表示 3 次方的多項式的，但為何 9 次方的 test 會得到一個這麼差的結果？

其實這就是上面所講的 overfitting，如果我們將 model 設定的很複雜，model 會找出使得 training data error 最小的參數，這個 model 對於這些 training data 來說是完美的(也可以想成是客製化的)，

但如果我們看其他的 case，誤差就會很大。

以下暫時無法解釋其原因，但是我們能從多項式係數看出一些端倪，以下是得到上述圖的係數

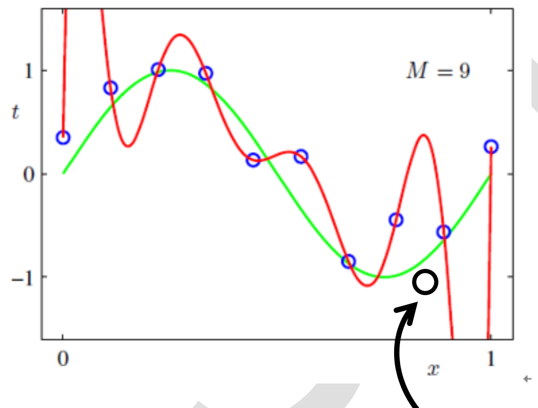
	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

我們可以看到 overfitting 時，其係數絕對值都非常的大

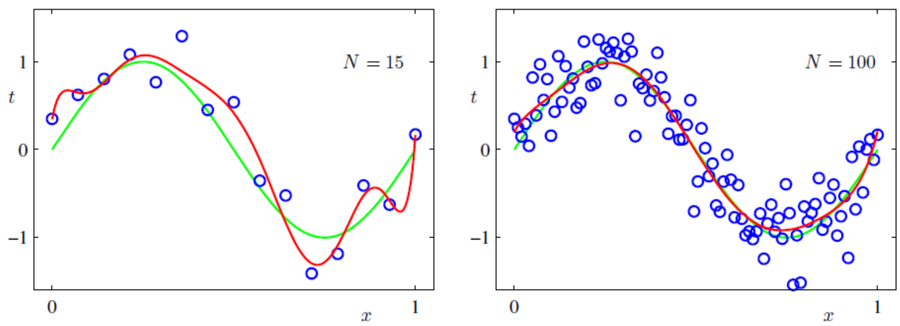
整理一下 overfitting 的成因：

1. 訓練資料量太少(下面會說明)
2. 有雜訊(不過即使沒有雜訊，也有可能造成 overfitting)

解釋一下第一點，其實 9 次方多項式是可以表示 3 次多項式的，但由於我們的數據太少，數據不夠複雜去告訴(引導)9 次多項式說 3 次方就夠了，我們很容易受幾個點而影響我們推估出來多項式的趨勢，但如果數據夠多，我們可以用其他附近的點去強迫多項式不要做劇烈的改變，因為有一堆其他的點去拉正



以這個圖來說，如果我們又有一筆數據在黑點處，那麼倒數第二個轉折點附近就不會那麼的陡峭



上兩圖都是 M=9 時的 solution，課本上說通常會取資料量的 1/5 或是 1/10 作為 M

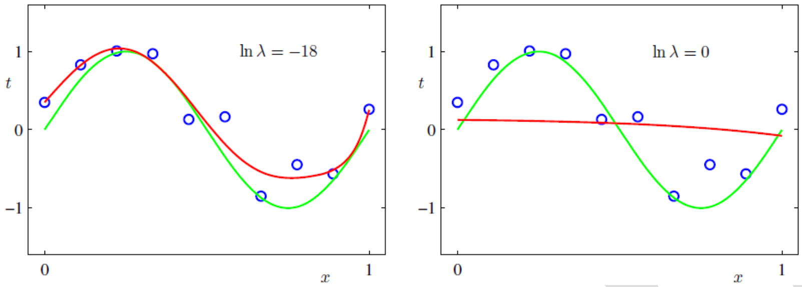
若是像例子，處理一個不是一次方的問題，我們常用的是 polynomial basis function

rLSE

為了解決係數太大的問題，我們常使用 regularization，也就是增加一個懲罰項懲罰係數太大的 **w**

$$\tilde{E}(\mathbf{w})=\sum_{n=1}^{N}\{(y(x_n,\mathbf{w})-t_n\}^2+\lambda\|\mathbf{w}\|^2$$

當 λ 越大時，懲罰項的影響就會越大，我們就會越會傾向找越小的係數，但是同時可能就會使得 error(training data error)增加，故選擇一個不錯的 λ 也是很重要的，下圖可看出 λ 的影響



	$\ln\lambda=-\infty$	$\ln\lambda=-18$	$\ln\lambda=0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

當 $\lambda=0$ 時，容易有 overfitting(等同於 LSE)

當 $\lambda=\infty$ 時，前項的誤差有多少已經不重要了，因為後項只要不等於 0，後項的值就大到足以忽略前項的

當我們要解 $\min\tilde{E}(\mathbf{w})=\sum_{n=1}^{N}\{(y(x_n,\mathbf{w})-t_n\}^2+\lambda\|\mathbf{w}\|^2$ 時，我們將其微分，得到 $(A^TA+\lambda I)^{-1}A^T$

$$\frac{d}{d\vec{x}}\left(\|A\vec{x}-\vec{b}\|^2+\lambda\|\vec{x}\|^2\right)=\frac{d}{d\vec{x}}\left((A\vec{x}-\vec{b})^T(A\vec{x}-\vec{b})+\lambda\vec{x}^T\vec{x}\right)$$

$$=\frac{d}{d\vec{x}}\left(\vec{x}^TA^TA\vec{x}-2\vec{x}^TA^T\vec{b}+\vec{b}^T\vec{b}+\lambda\vec{x}^T\vec{x}\right)$$

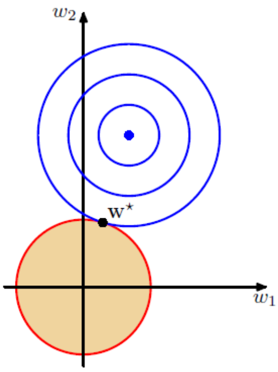
$$=2A^TA\vec{x}-2A^T\vec{b}+2\lambda\vec{x}=0$$

$$\Rightarrow(A^TA+\lambda I)\vec{x}=A^T\vec{b}$$

$$\Rightarrow\vec{x}=(A^TA+\lambda I)^{-1}A^T\vec{b}$$

如果 A^TA 為正定，則 $A^TA+\lambda I$ 必定為正定，至於若 A^TA 不可逆，則 $A^TA+\lambda I$ 也必定為正定，固可保證 $A^TA+\lambda I$ 可逆

用圖形解 $\min\tilde{E}(\mathbf{w})=\sum_{n=1}^{N}\{(y(x_n,\mathbf{w})-t_n\}^2+\lambda\|\mathbf{w}\|^2$ ：



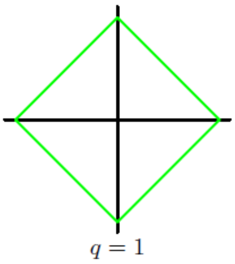
藍色的線假設是$\{(y(x_n,\mathbf{w})-t_n\}^2$中不同$\mathbf{w}$所得到的值，$\{(y(x_n,\mathbf{w})-t_n\}^2$中若得到相同的值，會在同一個圓內，也可用等高線圖來理解，紅線實心為土色的圓為$||\mathbf{w}||^2$的等高線圖，只要和原點距離相同，$||\mathbf{w}||^2$就會相同，所以也是以同心圓的方式呈現。根據λ的不同，我們可以決定是$\{(y(x_n,\mathbf{w})-t_n\}^2$比較重要，還是$||\mathbf{w}||^2$比較重要，$\lambda$值是我們可以自行決定的。

此方法稱為 ridge 或是 L_2 norm

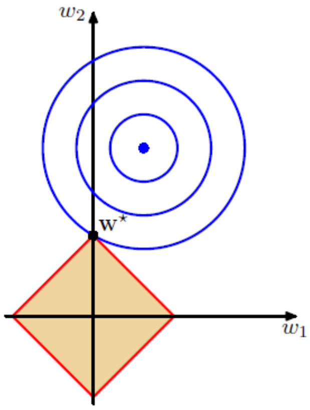
而根據這個概念，又出現了 Lasso 或稱為 L_1 norm

$$\min \tilde{E}(\mathbf{w}) = \sum_{n=1}^{N}\{(y(x_n,\mathbf{w})-t_n\}^2 + \lambda||\mathbf{w}||$$

$||\mathbf{w}||$可以想成絕對值的概念，故只要各軸的值取絕對值相加相同，就會在同一條線上，圖形就會變成



$\min \tilde{E}(\mathbf{w}) = \sum_{n=1}^{N}\{(y(x_n,\mathbf{w})-t_n\}^2 + \lambda||\mathbf{w}||$用圖形解就變成



會喜歡用這種方式而不是 L_2 norm 是因為其最佳點通常都在頂點上，頂點上意謂其他維度的值都為 0，可以簡化我們之後的運算。