CS5321 Numerical Optimization Homework 2

Due Dec 2

1. (20%) Check out the TRUST-REGION NEWTON- LANCZOS METHOD in Section 7.1 in the . What kind of problem it wants to solve? and how the Lanczos method solves it.

   雖然可以接受所有方向的反向曲率，它也會把沒有很大變化之曲率納入，所以會影響到取最小值。

   因為置信域的運作方式就是可以想像，先畫一個大圈，這個大圈就會有一個local model(根據老師上課筆記)，然後在其內找尋較小值，計算其與初始點的位置，藉此將位置帶入模型與真實函數，可以獲得模擬模型的下降量與真實函數的下降量，再用此兩個得比值，來控制置信域半徑大小，直到找到下一個位置後，就可以再做一次上述的步驟。

   那為何這邊使用要採用lanczos method?

   第一點，它可以解決如果hessian矩陣不是正定矩陣的問題，對於牛頓法來說，如果hessian矩陣為非正定，計算步長會導致分母出現零，導致整個步長趨近於無限大。

   第二點，使用lanczos method後會產生一個Q_j矩陣，此矩陣可以寫成

   $$Q_j^T B Q_j = T_j$$

   又T_j為Tridiangonal，就是只有對角線上與上下各多一排的矩陣，再根據課本上對置信域model的function

   $$\min m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T H_k p$$

   將$p_k = Q_j w$帶入

   $$\min_{w \in R^j} f_k + e_1^T Q_j (\nabla f_k) e_1^T w + \frac{1}{2} w^T T_j w$$

   至於該何時結束iteration，則採用講義公式7.3

   $$\|r_k\| \leq \eta_k \|\nabla f_k\|$$

   $$rk = \nabla^2 f_k p_k + \nabla f_k$$

第二條為residual，因為它是乘上inexact newton step，所以兩者相加不一定等於零，依據這個殘值，來決定何時停止iteration，因為之前所教的牛頓法需要正確的函數，但有時目標函數過於複雜或是非線性，不適合用來計算其Hessian，則會採用擬牛頓法進行近似評估在計算Hessian。

這個方法就是可以幫助加速運算，因為tridiangonal會將H_k替換成T_j，可以簡化運算過程，而且T_j可以使用cholesky分解，就可以快速的算出答案。

2. (20%) Check out section 8.2 in the deep learning textbook. Give a summary about the major challenges in neural network optimization.

總共有八個問題

第一點ill-conditioning 又稱為病態情況，其實在線性代數中就有提到這個問題，其實就是當輸入值有微小的擾動也會造成很大的變化，那這邊所探討的就是有關於Hessian matrix的病態情況，只要增加一小步，他就會大幅增加cost function。

第二點local minima，就跟老師上課講的一樣，對於convex function 其找到的最小值，也會是global minimum 對於深度學習，只要是model identifiability的問題(表示此模型可以經過多次訓練可以完美詮釋整體參數)都會有多個local minima，這是很正常的情況，通常為了描述周遭世界的模型，都會較為複雜，層數越多，局部最小值會越多，那要如何將每個參數找到最小值，就可以用相當多的方法處理優化局部最小值的問題，因為有時得到局部最小值需要high cost，如果每次找local minima都需要花費很高得代價，那就會導致問題產生。

第三點為Plateaus, Saddle Points and Other Flat Regions，在真實問題上，Gradient為零不一定為局部最小值，比如說馬鞍點(上課內容)周遭點同時會有大於它的點也會有小於它的點，對於此點來說，可以當作局部最小值，相對於較高的位置，而對於較低的位置則會是局部最大值，這就是問題所在，因為對於高維度問題大部分的最小值位置通常都是鞍點，但這會造成hessian matrix的特徵值有正有負，會影響到最佳化的運算。那如果是flat regions，會讓其hessian matrix 為零，讓其難以找出search direction，影響最佳化的運行。

第四點為Cliffs and Exploding Gradients，Cliffs又有懸崖的意思，當gradient計算到邊緣時，因為會有很大的高度差，就會有很大的Gradient，讓其直接跳離整個cliffs，如果這之中有最小值的存在，可能因為跳太大步導致無法找到，當然這個問題已經有解決方法，gradient clipping heuristic，當他發現找到的gradient過大，則降低其step size，就可以避免掉這個問題。

第五點為Long-Term Dependencies，會發生在當學習深度很深，表示他有好幾層layers，課本上有舉一個例子

$$W^t = (Vdiag(\lambda)V^-1)^t = Vdiag(\lambda)^tV^-1$$

其實從這是式子可以很明顯的看出來λ會大大影響這個W^t，尤其當t的次數很大的時候，這個問題又稱作vanishing and exploding gradient prob-

lem，當λ小於1，則W^t會等於零(vanishing);反之則W^t會變得很大(exploding)，那為何會使用這個例子呢?因為這樣可以找出最大的eigunvalue(此方法又稱作power method)，使用的場合就是RNN，而W其實就是大家所熟知的權重，在此網路中W是共享的，所以上述例子才會成立，那解決方法則是前饋神經網路。

第六點為Inexact Gradients ，我們前幾章的講義的方法，牛頓法之類的，都是假設在我們可以獲得確切的Gradient and Hessian matrix ，所以後來才出現擬牛頓法(inexact newton method)，那對於機器學習來說，則常常會使用最大概似估計法，藉由資料來推其參數，但這樣的題目會受到收集的資料影響，如果資料不夠全面，則會造成與真實環境的參數會有出入(當然要觀察到真實環境的參數也是很難的)，所以其gradient 和Hessian無法得到正確的值，進而影響最佳化演算法的表現。

第七點為Poor Correspondence between Local and Global Structure，根據課本上的定義，初始值對於最佳化來說是相當重要的，因為它會影響他是否可以收斂到最小值的問題，而會有這個問題的起因，都是因為我們無法得知整個全域模型的樣子，如果可以得知global minima 的位置，那演算法在計算時就可以往目標點移動，就算前面必須越過一個hill，但事實上很難知道整個global structure，而gradient就是一個local structure只能知道局部區域的資訊，所以現在這些問題，通常都是透過找到較好的初始值來解決複雜的model。

第八點為Theoretical Limits of Optimization，那這一部分就是探討最佳化法的問題，因為有的最佳化法的理論輸出會是離散的，但大部分的NN則是輸出連續的數字; 有些研究顯示有些問題是難以解決的，但卻無法分辨某些問題是否屬於難以解決的問題;NN訓練通常只希望cost function可以小一點，而在執行前通常都需要找一個最佳化法看是否可以達成目標，但這就最難的部分，需要經驗跟調適參數才可以達成。

3. (10%) Let J be an $m \times n$ matrix, $m \geq n$. Show that J has full column rank if and only if $J^T J$ is positive definite.

雙向證法
第一步
假設J為full column rank，表示每個column vector are all independent to each other
所以

$$J\vec{x} \neq 0, \forall \vec{x} \neq \vec{0}$$

左右乘上$(J\vec{x})^T$

$$\vec{x}^T J^T J\vec{x} > 0$$

上述式子就是判斷是否正定，可以把它當成矩陣的平方，所有的值必定大於零，所以可以得知JJ^T為正定

第二步
假設JJ^T為正定，可以表示成

$$\vec{x}^T J^T J\vec{x} > 0$$

$$\Rightarrow (J\vec{x})^T(J\vec{x}) > 0$$

表示

$$J\vec{x} \neq 0, \forall \vec{x} \neq \vec{0}$$

x可以看成係數，當J乘上任意係數都可以有值，表示所有column vectors are all independent to each column vector

所以可以得知J是full column rank
故得證

4. (30%) Simplex method (the algorithm is shown in Figure 2): Consider the following linear program:

$$\begin{array}{ll} \max_{x_1,x_2} & z = 8x_1 + 5x_2 \\ \text{s.t.} & 2x_1 + x_2 \leq 1000 \\ & 3x_1 + 4x_2 \leq 2400 \\ & x_1 + x_2 \leq 700 \\ & x_1 - x_2 \leq 350 \\ & x_1, x_2 \geq 0 \end{array}$$

(a) Transform it to the standard form.

$$A = \begin{bmatrix} 2 & 1 \\ 3 & 4 \\ 1 & 1 \\ 1 & -1 \end{bmatrix} \vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \vec{b} = \begin{bmatrix} 1000 \\ 2400 \\ 700 \\ 350 \end{bmatrix}$$

將不定式改成相等的方程式，所以可以改寫成下列四條式子

$$\begin{aligned} 2x_1 + x_2 + x_3 &= 1000 \\ 3x_1 + 4x_2 + x_4 &= 2400 \\ x_1 + x_2 + x_5 &= 700 \\ x_1 - x_2 + x_6 &= 350 \end{aligned}$$

將z function 用$\vec{c}$來表達

$$\vec{c} = \begin{bmatrix} 8 \\ 5 \end{bmatrix}$$

又x_3, x_4, x_5, x_6 是slack variables

因此

$$, \quad A = \begin{bmatrix} 2 & 1 & 1 & 0 & 0 & 0 \\ 3 & 4 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & 0 & 1 \end{bmatrix}, \vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}, \quad \vec{b} = \begin{bmatrix} 1000 \\ 2400 \\ 700 \\ 350 \end{bmatrix}, \quad \vec{c} = \begin{bmatrix} 8 \\ 5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

從講義中可以得知$\max_{\vec{x}} \vec{c}^T\vec{x}$，換成最小值，才可以使用講義上的simplex method，

$$-\min_{\vec{x}} -\vec{c}^T\vec{x}$$

$$-\min_{\vec{x}} z = \begin{bmatrix}-8\\-5\\0\\0\\0\\0\end{bmatrix}^{\mathrm{T}} \begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\\x_6\end{bmatrix}$$

範圍條件寫成$A\vec{x} = \vec{b}$

$$\begin{bmatrix}2 & 1 & 1 & 0 & 0 & 0\\3 & 4 & 0 & 1 & 0 & 0\\1 & 1 & 0 & 0 & 1 & 0\\1 & -1 & 0 & 0 & 0 & 1\end{bmatrix} \begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\\x_6\end{bmatrix} = \begin{bmatrix}1000\\2400\\700\\350\end{bmatrix}, \vec{x} \geq 0$$

(b) Suppose the initial guess is $(0, 0)$. Use the simplex method to solve this problem. In each iteration, show

- Basic variables and non-basic variables, and their values.
- Pricing vector.
- Search direction.
- Ratio test result.

第一次疊代
分別有儲存index和每一個index的值

$$\mathcal{B}_0 = \{3, 4, 5, 6\}, \mathcal{N}_0\{1, 2\}$$

$$B_0 = \begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & 1\end{bmatrix}, N_0 = \begin{bmatrix}2 & 1\\3 & 4\\1 & 1\\1 & -1\end{bmatrix}$$

先計算basic variable 和non-basic variable

$$\vec{x_B} = \begin{bmatrix}x_3\\x_4\\x_5\\x_6\end{bmatrix} = B_0^{-1}\vec{b} = \begin{bmatrix}1000\\2400\\700\\350\end{bmatrix}$$

$$\vec{x_N} = \begin{bmatrix}x_1\\x_2\end{bmatrix} = \begin{bmatrix}0\\0\end{bmatrix}$$

$\vec{c}$為紀錄要minimum 的function

$$\vec{c_B} = \vec{c}(\mathcal{B}_0) = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}, \vec{c_N} = \vec{c}(\mathcal{N}_0) = \begin{bmatrix}-8\\-5\end{bmatrix}$$

計算pricing vector，會發現其第一次pricing vector等於minimum function的係數

$$\vec{s_0} = \vec{c_N} - {N_0}^T {B_0}^{-T} \vec{c_B} = \begin{bmatrix}-8\\-5\end{bmatrix} - \begin{bmatrix}2&1\\3&4\\1&1\\1&-1\end{bmatrix}^{\mathrm{T}} \left(\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}^{-1} \right)^{\mathrm{T}} \begin{bmatrix}0\\0\\0\\0\end{bmatrix} = \begin{bmatrix}-8\\-5\end{bmatrix} \quad q_0 = 1, i_q = 1$$

當pricing vector 小於零，計算serach direction，來確定遞減方向

$$\vec{s_0} < 0, \vec{d_0} = {B_0}^{-1} N(:, q_0) = {B_0}^{-1} \begin{bmatrix}2\\3\\1\\1\end{bmatrix} = \begin{bmatrix}2\\3\\1\\1\end{bmatrix}$$

求出最小值，設其為step size ，並且記錄其在矩陣的位置i_p

$$[\gamma_0, i_p] = min \frac{\vec{x_B}(i)}{\vec{d_0}(i)} = min \begin{bmatrix}500\\800\\700\\350\end{bmatrix} \quad \gamma_0 = 350, i_p = 6$$

最後就可以得到第一次iteration完的$\vec{x}$，除此之外，你會發現$\vec{d_0}$多一個負號，可以在計算$\vec{d_0}$時加，也可以在最後計算家加上負號

$$\vec{x_1} \begin{pmatrix}\mathcal{B}_0\\\mathcal{N}_0\end{pmatrix} = \begin{pmatrix}\vec{x_B}\\\vec{x_N}\end{pmatrix} + \gamma_0 \begin{pmatrix}-\vec{d_0}\\\vec{e_{i_q}}\end{pmatrix} = \begin{bmatrix}0\\0\\1000\\2400\\700\\350\end{bmatrix} + 350 \begin{bmatrix}1\\0\\-2\\-3\\-1\\-1\end{bmatrix} = \begin{bmatrix}350\\0\\300\\1350\\350\\0\end{bmatrix}$$

第二次疊代

$$\mathcal{B}_1 = \{3, 4, 5, 1\}, \quad \mathcal{N}_1\{6, 2\}$$

$$B_1 = \begin{bmatrix}1&0&0&2\\0&1&0&3\\0&0&1&1\\0&0&0&1\end{bmatrix}, N_1 = \begin{bmatrix}0&1\\0&4\\0&1\\1&-1\end{bmatrix}$$

$$\vec{x_B} = \begin{bmatrix}x_3\\x_4\\x_5\\x_1\end{bmatrix} = \begin{bmatrix}300\\1350\\350\\350\end{bmatrix} = B_1^{-1}\vec{b}$$

$$\vec{x_N} = \begin{bmatrix} x_6 \\ x_2 \end{bmatrix} = \vec{x_1}\,(\mathcal{N}_1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\vec{c_B} = \vec{c}(\mathcal{B}_1) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -8 \end{bmatrix}, \vec{c_N} = \vec{c}(\mathcal{N}_1) = \begin{bmatrix} 0 \\ -5 \end{bmatrix}$$

$$\vec{s_1} = \vec{c_N} - {N_1}^{\mathrm{T}} {B_1}^{-T} \vec{c_B} = \begin{bmatrix} 0 \\ -5 \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ 0 & 4 \\ 0 & 1 \\ 1 & -1 \end{bmatrix}^{\mathrm{T}} \left(\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \right)^{\mathrm{T}} \begin{bmatrix} 0 \\ 0 \\ 0 \\ -8 \end{bmatrix} = \begin{bmatrix} 8 \\ -13 \end{bmatrix} \quad q_1 = 2, i_q = 2$$

$$\vec{s_1} < 0, \vec{d_1} = {B_1}^{-1} \begin{bmatrix} 1 \\ 4 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 2 \\ -1 \end{bmatrix}$$

$$[\gamma_1, \quad i_p] = min\, \frac{\vec{x_B}(i)}{\vec{d_1}(i)} = min \begin{bmatrix} 100 \\ \frac{1350}{7} \\ 175 \\ -350 \end{bmatrix} \quad \gamma_1 = 100, i_p = 3$$

得到第二次疊代的$\vec{x_2}$

$$\vec{x_2} \begin{pmatrix} \mathcal{B}_1 \\ \mathcal{N}_1 \end{pmatrix} = \begin{bmatrix} 350 \\ 0 \\ 300 \\ 1350 \\ 350 \\ 0 \end{bmatrix} + 100 \begin{bmatrix} 1 \\ 1 \\ -3 \\ -7 \\ -2 \\ 0 \end{bmatrix} = \begin{bmatrix} 450 \\ 100 \\ 0 \\ 650 \\ 150 \\ 0 \end{bmatrix}$$

第三次疊代

$$\mathcal{B}_2 = \{2, 4, 5, 1\}, \mathcal{N}_2\{6, 3\}$$

$$B_2 = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 4 & 1 & 0 & 3 \\ 1 & 0 & 1 & 1 \\ -1 & 0 & 0 & 1 \end{bmatrix}, N_2 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}$$

$$\vec{x_B} = \begin{bmatrix} x_2 \\ x_4 \\ x_5 \\ x_1 \end{bmatrix} = \begin{bmatrix} 100 \\ 650 \\ 150 \\ 450 \end{bmatrix} = {B_2}^{-1} \vec{b}$$

$$\vec{x_N} = \begin{bmatrix} x_6 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\vec{c_B} = \vec{c}(\mathcal{B}_2) = \begin{bmatrix} -5 \\ 0 \\ 0 \\ -8 \end{bmatrix}, \vec{c_N} = \vec{c}(\mathcal{N}_2) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\vec{s_2} = \vec{c_N} - {N_2}^T {B_2}^{-T} \vec{c_B} = \begin{bmatrix}0\\0\end{bmatrix} - \begin{bmatrix}0&1\\0&0\\0&0\\1&0\end{bmatrix}^{\mathrm{T}} \left(\begin{bmatrix}1&0&0&2\\4&1&0&3\\1&0&1&1\\-1&0&0&1\end{bmatrix}^{-1} \right)^{\mathrm{T}} \begin{bmatrix}-5\\0\\0\\-8\end{bmatrix} = \begin{bmatrix}\frac{-2}{3}\\\frac{13}{3}\end{bmatrix} \quad q_0 = 1, i_q = 6$$

$$\vec{s_2} < 0, \vec{d_2} = {B_2}^{-1} \begin{bmatrix}0\\0\\0\\1\end{bmatrix} = \begin{bmatrix}\frac{-2}{3}\\\frac{5}{3}\\\frac{1}{3}\\\frac{1}{3}\end{bmatrix}$$

$$[\gamma_2, \quad i_p] = min \frac{\vec{x_B}(i)}{\vec{d_2}(i)} = min \begin{bmatrix}-150\\390\\450\\1350\end{bmatrix} \quad \gamma_2 = 390, \quad i_p = 4, \quad p_2 = 2$$

得到第三次疊代的$\vec{x_3}$

$$\vec{x_3} \begin{pmatrix} \mathcal{B}_2 \\ \mathcal{N}_2 \end{pmatrix} = \begin{bmatrix}450\\100\\0\\650\\150\\0\end{bmatrix} + 390 \begin{bmatrix}\frac{-1}{3}\\\frac{2}{3}\\0\\\frac{-5}{3}\\\frac{-1}{3}\\1\end{bmatrix} = \begin{bmatrix}320\\360\\0\\0\\20\\390\end{bmatrix}$$

第四次疊代

$$\mathcal{B}_3 = \{2, 6, 5, 1\}, \quad \mathcal{N}_3\{4, 3\}$$

$$B_3 = \begin{bmatrix}1&0&0&2\\4&0&0&3\\1&0&1&1\\-1&1&0&1\end{bmatrix}, \quad N_3 = \begin{bmatrix}0&1\\1&0\\0&0\\0&0\end{bmatrix}$$

$$\vec{x_B} = \begin{bmatrix}x_2\\x_6\\x_5\\x_1\end{bmatrix} = \begin{bmatrix}360\\390\\20\\320\end{bmatrix} = {B_3}^{-1} \vec{b}$$

$$\vec{x_N} = \begin{bmatrix}x_4\\x_3\end{bmatrix} = \begin{bmatrix}0\\0\end{bmatrix}$$

$$\vec{c_B} = \vec{c}(\mathcal{B}_3) = \begin{bmatrix}-5\\0\\0\\-8\end{bmatrix}, \vec{c_N} = \vec{c}(\mathcal{N}_3) = \begin{bmatrix}0\\0\end{bmatrix}$$

$$\vec{s_3} = \vec{c_N} - {N_3}^T {B_3}^{-T} \vec{c_B} = \begin{bmatrix}0\\0\end{bmatrix} - \begin{bmatrix}0&1\\1&0\\0&0\\0&0\end{bmatrix}^{\mathrm{T}} \left(\begin{bmatrix}1&0&0&2\\4&0&0&3\\1&0&1&1\\-1&1&0&1\end{bmatrix}^{-1} \right)^{\mathrm{T}} \begin{bmatrix}-5\\0\\0\\-8\end{bmatrix} = \begin{bmatrix}\frac{2}{5}\\\frac{17}{5}\end{bmatrix}$$

可以發現pricing vector都已經大於0，表示已經是最佳解，就算再做下去也不會得到更大的值，所以就可以直接得到最佳解$\vec{x}$在第三步的最後一次疊代的結果，

$$\vec{x_3}\begin{pmatrix}\mathcal{B}_2\\ \mathcal{N}_2\end{pmatrix}=\begin{bmatrix}320\\360\\0\\0\\20\\390\end{bmatrix}, x_1=320, x_2=360$$

將其帶入$max_{x_1,x_2}\quad z=8x_1+5x_2$
得其最大值為4360

5. (20%) Farkas lemma: Let A be an $m \times n$ matrix and $\vec{b}$ be an m vector. Prove that exact one of the following two statements is true:

   (a) There exist a $\vec{x} \in \mathbb{R}^n$ such that $A\vec{x} = \vec{b}$ and $\vec{x} \geq 0$.

   (b) There exists a $\vec{y} \in \mathbb{R}^m$ such that $A^T\vec{y} \geq 0$ and $\vec{b}^T\vec{y} < 0$.

   (Hint: prove if (a) is true, then (b) cannot be true, and vice versa.)

   根據網路上的理解，可以知道沒辦法直接證明定理中的a與b這兩個條件，因為這兩個條件是只有一方可以成立的，那採取的方法即為選擇其中一個是成立，證明另一個條件不會成立。

   將A表示成下面這樣的方式，就是將column vector 拆成n個

   $$A = [v_1, v_2, v_3, ..., v_n]$$

   設定conic combination of the columns of A 為Q

   $$Q = cone(v_1, ..., v_n) = \left\{ s \in R^m : s = \sum_{i=1}^{n} \lambda_i v_i, \lambda_i > 0, \forall i \right\}$$

   改寫成

   $$Ax = \sum_{i=1}^{n} x_i v_i$$

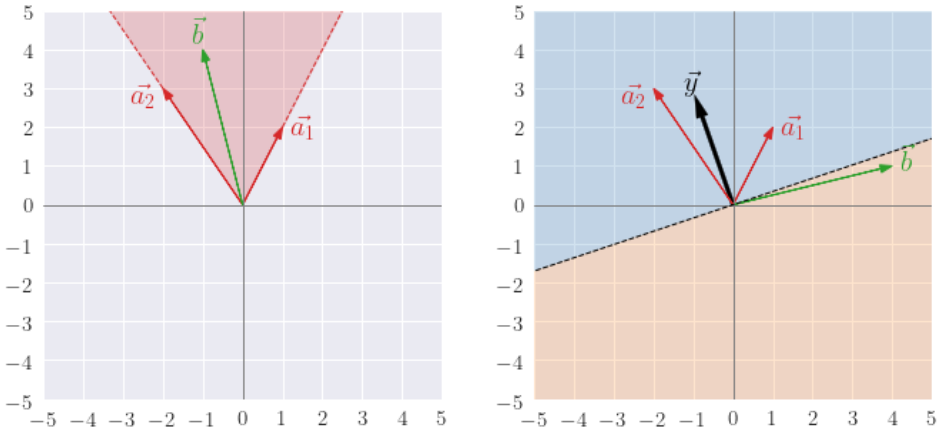
   上述為矩陣基本性質。

   假設$b \notin Q$
   根據(a)選項的性質，先假設$\vec{x} > 0$ ，然後使用separating hyperplane theorem ，這個方法表示會有一個$\alpha \in R^m, \alpha \neq 0$ ，然後設定β讓$\alpha^T b$ 會依據$s \in Q$大於或小於β，當$\alpha^T b < \beta$又$s = \lambda_i v_i$所以

   $$\alpha^T v_i < \frac{\beta}{\lambda}, \forall \lambda > 0$$

   當$\beta > 0$和$\lambda \to \inf$，可以設定$y = -\alpha$，我們可以得到對於所有的i

   $$y^T v_i \geq 0$$



case (a) case (b)

Figure 1: Two cases of Farkas Lemma.

又v_i是A的column vector, we get that $A^T y \geq 0$代表(b)成立

第二個的證法

假設(b)不成立，如果(b)不成立，這邊換一個想法，先用column vector展開成一個conic，(b)不成立代表其$\vec{b}$會出現在conic內，這也就符合(a)的條件。反之亦然，可以將上述第一部分快速的證明，當(a)不成立，表$\vec{b}$必定出現在conic的外面，又當在conic外面，(b)的條件一定會成立。
故得證

Farkas's Lemma (1902) plays an important role in the proof of the KKT condition. The most critical part in the proof of the KKT condition is to show that the Lagrange multiplier $\vec{\lambda}^* \geq 0$ for inequality constraints. We can say if the LICQ condition is satisfied at $\vec{x}^*$, then any feasible direction $\vec{u}$ at $\vec{x}^*$ must have the following properties:

1. $\vec{u}^T \nabla f(\vec{x}^*) \geq 0$ since $\vec{x}^*$ is a local minimizer. (Otherwise, we find a feasible descent direction that decreases f.)
2. $\vec{u}^T \nabla c_i(\vec{x}^*) = 0$ for equality constraints, $c_i = 0$.
3. $\vec{u}^T \nabla c_i(\vec{x}^*) \geq 0$ for inequality constraints, $c_i \geq 0$.

Here is how Farkas Lemma enters the theme. Let $\vec{b}$ be $\nabla f(\vec{x}^*)$, $\vec{y}$ be $\vec{u}$ (any feasible direction at $\vec{x}^*$), the columns of A be $\nabla c_i(\vec{x}^*)$). Since no such $\vec{u}$ exists, according to the properties of $\vec{y}$, statement (a) must hold. The vector $\vec{x}$ in (a) corresponds to $\vec{\lambda}^*$, which just gives us the desired result of the KKT condition.

(1) Given a basic feasible point $\vec{x}_0$ and the corresponding index set $\mathcal{B}_0$ and $\mathcal{N}_0$.

(2) For $k = 0, 1, \ldots$

(3) Let $B_k = A(:, \mathcal{B}_k), N_k = A(:, \mathcal{N}_k)$, $\vec{x}_B = \vec{x}_k(\mathcal{B}_k), \vec{x}_N = \vec{x}_k(\mathcal{N}_k)$, and $\vec{c}_B = \vec{c}_k(\mathcal{B}_k), \vec{c}_N = \vec{c}_k(\mathcal{N}_k)$.

(4) Compute $\vec{s}_k = \vec{c}_N - N_k^T B_k^{-1} \vec{c}_B$ (pricing)

(5) If $\vec{s}_k \geq 0$, return the solution $\vec{x}_k$. (found optimal solution)

(6) Select $q_k \in \mathcal{N}_k$ such that $\vec{s}_k(i_q) < 0$, where i_q is the index of q_k in $\mathcal{N}_k$

(7) Compute $\vec{d}_k = B_k^{-1} A_k(:, q_k)$. (search direction)

(8) If $\vec{d}_k \leq 0$, return `unbounded`. (unbounded case)

(9) Compute $[\gamma_k, i_p] = \min_{i, \vec{d}_k(i) > 0} \dfrac{\vec{x}_B(i)}{\vec{d}_k(i)}$ (ratio test)
(The first return value is the minimum ratio;
the second return value is the index of the minimum ratio.)

(10) $$x_{k+1} \begin{pmatrix} \mathcal{B} \\ \mathcal{N} \end{pmatrix} = \begin{pmatrix} \vec{x}_B \\ \vec{x}_N \end{pmatrix} + \gamma_k \begin{pmatrix} -\vec{d}_k \\ \vec{e}_{i_q} \end{pmatrix}$$
($\vec{e}_{i_q} = (0, \ldots, 1, \ldots, 0)^T$ is a unit vector with i_qth element 1.)

(11) Let the i_pth element in $\mathcal{B}$ be p_k. (pivoting)
$\mathcal{B}_{k+1} = (\mathcal{B}_k - \{p_k\}) \cup \{q_k\}$, $\mathcal{N}_{k+1} = (\mathcal{N}_k - \{q_k\}) \cup \{p_k\}$

Figure 2: The simplex method for solving (minimization) linear programming