Software Implementation of SVD based on Jacobi algorithm

基於 Jacobi 演算法之軟體實現

專題領域：通訊組

組別：A224

指導教授：翁詠祿

組員姓名：莫明勳、王向梵

Abstract

通訊世代約每十年更迭一次。2020 年，5G 通訊成功推入人們的生活。為了預計在 2030 年推行的下一世代的 6G 通訊，基於低軌道衛星之通訊技術，是未來第六代通訊系統中的關鍵環節之一。

考量到部屬至衛星酬載時，須優先採納低能耗、低複雜度的架構，故實驗室採用介於純類比與純數位波束成形的架構，即混和波束成形(Hybrid Beamforming)，則是將發射端對應之預編碼器(Precoder)拆分成射頻預編碼器(RF Precoder)與基頻預編碼器(Baseband Precoder)，成為兩階段的波束成形結構。

本專題基於實驗室之前的基礎，為了取得 optimal precoder，針對通道矩陣之奇異值分解(Singular Value Decomposition, SVD)，以獲取右奇異矩陣(Right-sigular-matrix)，做為理想預編碼器矩陣 Fopt 為一必要之過程，然隨著巨量多輸入多輸出(massive MIMO)概念之引入，其通道陣列之維度將會隨天線陣列之大小而上升，於 LEO 酬載以傳統硬體實現 SVD 的複雜度極為困難。故在本專題，我們將透過軟體模擬 SVD 演算法，進行運算精準度、CORDIC 硬體實作的可行性上探討，期望減少 SVD 分解之複雜度並優化上述演算法於硬體實現上的限制。

Introduction

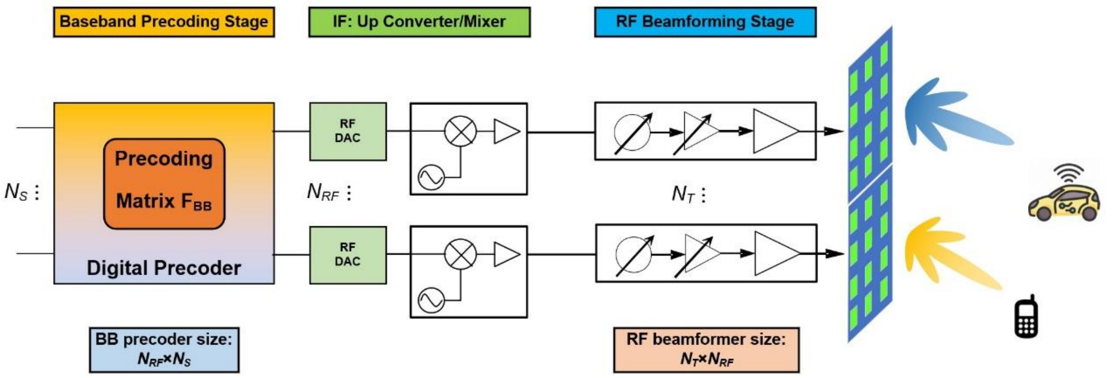


Fig1. 混合波束成形之通訊系統架構圖

在實驗室計畫中，實驗室負責 Baseband Tx/Rx 的部分，本專題所實現之 SVD 演算法將作為 hybrid beamforming 演算法中，理想 precoder 的最佳解，hybrid beamforming 演算法會嘗試逼近最佳解，但是受限於 quantization 的議題，實際上是無法達到跟最佳解相同的結果。

整體 HBF 的架構圖可見 Fig1，假設訊號經過 Digital precoder，進行基頻處理之後，頻率約為 3.5GHz 的中頻訊號，再給後面的 IF Tx，先經過 mixer 升頻至約 28GHz 的訊號，再經過 pulse shape filter 處理，pulse shape filter 的用意在於抑制訊號的 sidelob，處理多個 carrier component，讓彼此頻寬在規格內，才不會互相影響，進而達到降低 ISI 的用途，pulse shape filter 處理完後便經過 RF Beamforming 的 stage，便可傳至 RF Tx 傳出，以下便介紹本專題所使用的 SVD 分解演算法。

SVD 分解是相當有效之矩陣分解技術，在混合波束成形的架構中，我們透過 SVD 分解通道矩陣 **H** 並取得其前右奇異值向量，作為設計過程中的理想預編碼器。在本專題中，我們將嘗試透過基於雙邊雅可比與 CORDIC 的 SVD 演算法，化簡硬體實現之複雜度。

(A) 雙邊雅可比(Two-sided Jacobi)演算法:

我們首先由對矩陣 **A** 進行奇異值分解(SVD)的通式，

$$\mathbf{A}=\mathrm{U}\sum V^T$$

可得兩關係，

$$A^T A = \mathrm{V}\Sigma^2 V^T$$

$$V^T(A^T A)V = \Sigma^2$$

其中正交矩陣 **V** 為 **A** 的右奇異向量，也是$\boldsymbol{A}^T\boldsymbol{A}$的特徵向量。同理，矩陣 **U** 為 **A** 的左奇異向量，也是$\boldsymbol{A}^T\boldsymbol{A}$的特徵向量。

雙邊雅可比的概念是將對稱矩陣$\boldsymbol{A}^T\boldsymbol{A}$轉變成對角矩陣，其轉變矩陣是 **V**。該矩陣 **V** 即為 SVD 分解中欲分解矩陣 **A** 構成$\boldsymbol{A}^T\boldsymbol{A}$的特徵向量。

首先，選擇一個雅可比旋轉矩陣 **J**=$\boldsymbol{G}^T$:

$$
\mathbf{J}(\mathrm{p},\mathrm{q},\emptyset) = \begin{array}{c} \\ \\ p \\ \\ q \\ \\ \end{array}
\begin{array}{c}
\begin{array}{cccccc} & & \mathrm{p} & & \mathrm{q} & \end{array} \\
\begin{bmatrix}
1 & & & & & \\
 & \ddots & & & & \\
 & & cos\emptyset & & -sin\emptyset & \\
 & & & \ddots & & \\
 & & sin\emptyset & & cos\emptyset & \\
 & & & & & 1
\end{bmatrix}
\end{array}
$$

$$
\mathrm{G}(\mathrm{p},\mathrm{q},\emptyset) = \begin{array}{c} \\ \\ p \\ \\ q \\ \\ \end{array}
\begin{array}{c}
\begin{array}{cccccc} & & \mathrm{p} & & \mathrm{q} & \end{array} \\
\begin{bmatrix}
1 & & & & & \\
 & \ddots & & & & \\
 & & cos\emptyset & & sin\emptyset & \\
 & & & \ddots & & \\
 & & -sin\emptyset & & cos\emptyset & \\
 & & & & & 1
\end{bmatrix}
\end{array}
$$

假設此為 nxn 的矩陣，先在上對角線的元素中($\frac{n(n-1)}{2}$個元素)找出最大值進行消除，此最大元素為$\mathrm{A}_{p,q}$，也就是說第 p 列第 q 行的元素，便能求出此次旋轉的$\mathbf{J}(\mathrm{p},\mathrm{q},\emptyset)$，此時也有相對應的$\boldsymbol{J}^T(\mathrm{p},\mathrm{q},\emptyset)$來消除下對角線的元素。消除完後，便再由上對角線的元素中找出最大值，再進行相同的步驟，以次類推。

雅可比旋轉矩陣可以把 p、q 行列上的四個元素進行交換，使非對角線上的元素$\boldsymbol{A}_{p,q}$化成 0。其中的旋轉角度$\emptyset$的公式則為:

$$\mathrm{Cot}(2\emptyset) = \frac{\mathrm{A}_{q,q}-\mathrm{A}_{p,p}}{2\mathrm{A}_{p,q}}$$

雙邊雅可比演算法中，在每一次對矩陣進行旋轉時會優先選擇非對角線元素中的最大值，作為構成欲處理的旋轉矩陣。矩陣 **A** 透過右連乘$\boldsymbol{J}(\mathrm{p},\mathrm{q},\emptyset)$把上對角線元素全歸零，並透過左連乘$\boldsymbol{J}^T(\mathrm{p},\mathrm{q},\emptyset)$則是把下對角線元素歸零，使得 **A** 為一個對稱矩陣，$\boldsymbol{A}_{i,j} = \boldsymbol{A}_{j,i}$。上述之過程可以表示為

$$J_k^T \ldots J_2^T J_1^T (A^T A) J_1 J_2 \ldots J_k = \Lambda = \Sigma^2$$

因此，右奇異矩陣即為 $\mathbf{V}=J_1 J_2 \ldots J_k$，其中$k = \max\{\frac{m}{2},\frac{n}{2}\}$。奇異值$\sigma$是$\boldsymbol{\Lambda}$矩陣

之對角線元素開根號，奇異值對應的特徵向量V_i按σ大小排列。又根據 SVD 通式關係可得$U_i=\frac{AV_i}{\sigma}$，因而求得左奇異向量 **U**，進而完成對於矩陣 **A** 完整之奇異值分解運算。

Forsythe and Henrici 提出如果要使一個 nxn 的矩陣對角線化，可以進行一連串 2x2 的奇異值運算。
一個 2x2 的奇異值分解可以寫成:

$$\begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix}^T \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} cos\varphi & sin\varphi \\ -sin\varphi & cos\varphi \end{bmatrix} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}$$

其中$\theta+\varphi=tan^{-1}\frac{c+b}{d-a}$，$\theta-\varphi=tan^{-1}\frac{c-b}{d+a}$。
有了上述的 2x2 雙邊雅可比演算模塊後，可以將此技術推廣到較大維度的矩陣，其方法如下:

對於原始矩陣 A 作取樣，將 A 拆解為多個 2x2 的子矩陣，並對子矩陣進行運算，假設 A 的第 i 列、第 j 行的元素表達為a_{ij}，其相對應的子矩陣為

$$A_{ij}=\begin{bmatrix} a_{2I-1,2J-1} & a_{2I-1,2J} \\ a_{2I,2J-1} & a_{2I,2J} \end{bmatrix}$$

以簡單幾個例子舉例，a_{11}所對應的子矩陣為$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$、$a_{12}$所對應的子矩陣為$\begin{bmatrix} a_{13} & a_{14} \\ a_{23} & a_{24} \end{bmatrix}$、$a_{21}$所對應的子矩陣為$\begin{bmatrix} a_{31} & a_{32} \\ a_{41} & a_{42} \end{bmatrix}$，經過這樣的取樣編碼，假設原始矩陣為一 10x10 的矩陣 A，它可以被取樣為 5x5 的矩陣 P，此 P 矩陣的每個元素皆為 2x2 子矩陣。

考量到複數的情況下，我們可以採用 Q-transformation 來對矩陣做對角化，定義 Q-變換為下:

$$\begin{bmatrix} c_\phi e^{i\Theta_\alpha} & -s_\phi e^{i\Theta_\beta} \\ s_\phi e^{i\Theta_\alpha} & c_\phi e^{i\Theta_\beta} \end{bmatrix} \begin{bmatrix} Ae^{i\Theta_a} & Be^{i\Theta_b} \\ Ce^{i\Theta_c} & De^{i\Theta_d} \end{bmatrix} \begin{bmatrix} c_\psi e^{i\Theta_\gamma} & s_\psi e^{i\Theta_\gamma} \\ -s_\psi e^{i\Theta_\delta} & c_\psi e^{i\Theta_\delta} \end{bmatrix}$$

完成對角化需要兩次的 Q-變換，第一次的變換是對原始矩陣做 QRD 分解，假設原始資料是給 polar coordinate，$\begin{bmatrix} Ae^{i\Theta_a} & Be^{i\Theta_b} \\ Ce^{i\Theta_c} & De^{i\Theta_d} \end{bmatrix}$，選擇$\Theta_\alpha=\Theta_\beta=\frac{-(\Theta_c+\Theta_d)}{2}$、$\Theta_\gamma=\Theta_\delta=\frac{(\Theta_d-\Theta_c)}{2}$、$\Theta_\phi=0$、$\Theta_\varphi=\tan^{-1}(\frac{C}{D})$，再經過上述的變換會得到

$\begin{bmatrix} We^{i\Theta_w} & Xe^{i\Theta_x} \\ 0 & Z \end{bmatrix}$。

第二步的 Q-變換則是完成對角化，從第一步得到的矩陣，根據 Q 變換的定義，又可以被寫成

$$\begin{bmatrix} c_\phi e^{i\Theta_{\alpha 1}} & -s_\phi e^{i\Theta_{\beta 1}} \\ s_\phi e^{i\Theta_{\alpha 1}} & c_\phi e^{i\Theta_{\beta 1}} \end{bmatrix} \begin{bmatrix} We^{i\Theta_w} & Xe^{i\Theta_x} \\ 0 & Z \end{bmatrix} \begin{bmatrix} c_\psi e^{i\Theta_{\gamma 1}} & s_\psi e^{i\Theta_{\gamma 1}} \\ -s_\psi e^{i\Theta_{\delta 1}} & c_\psi e^{i\Theta_{\delta 1}} \end{bmatrix} = \begin{bmatrix} P & 0 \\ 0 & Q \end{bmatrix}$$

此時的角度選擇為 $\Theta_{\alpha 1} = \frac{-(\Theta_x+\Theta_w)}{2}$、$\Theta_{\beta 1} = \Theta_{\gamma 1} = \frac{(\Theta_x-\Theta_w)}{2}$、$\Theta_w = \frac{(\Theta_w-\Theta_x)}{2}$、

$\tan(\Theta_\phi \pm \Theta_\psi) = -(\frac{X}{Z\mp W})$。

接著再對 P 矩陣的每個元素做上述的雙邊雅可比演算後，即可完成大矩陣的 SVD。

(B) 座標旋轉數字計算機(Coordinate Rotation Digital Computer, CORDIC):

CORDIC 演算法是一個計算三角函數、雙曲函數、平方根、乘法、除法、指數和具有任意底數的對數的簡單而有效的算法。CORDIC 乃由遞迴的旋轉以及一系列提前固定的角度集合組成，這些角度之順序關係可表示為

$$\emptyset_i = \tan^{-1}(2^{-i});\ for\ i = 0, 1, 2, \ldots$$

由高到低排列，即 45º, 26.7º, 14º, 7.1º, 3.57º 等角度。判斷一系列的旋轉向左轉或向右轉，是取決於給定目標以及現在旋轉的結果來決定，如果當次旋轉超越目標角度則下次遞迴將反向旋轉，反之若還未目標就繼續旋轉，而目標向量是透過原始向量乘上旋轉矩陣所得，如以下所示。

$$\begin{matrix} x_{i+1} \\ y_{i+1} \end{matrix} = \begin{matrix} x_i \\ y_i \end{matrix} \begin{bmatrix} cos\emptyset & -sin\emptyset \\ sin\emptyset & cos\emptyset \end{bmatrix} = \prod_i \begin{bmatrix} cos\emptyset_i & -sin\emptyset_i \\ sin\emptyset_i & cos\emptyset_i \end{bmatrix} \begin{matrix} x_i \\ y_i \end{matrix}$$

隨著遞迴次數越多，CORDIC 近似之結果也將更加準確。而 CORDIC 演算法採用提前固定之角度集由二的冪次方角度構成，因此硬體上可以單純透過加法器和位移器實現。

CORDIC 有兩種模式，向量模式(Vectoring Mode)以及旋轉模式(Rotating Mode)，Vectoring Mode 的目的在於獲取給定之向量所需旋轉的角度(產生角度 Z)，而 Rotating Mode 這是把原始向量旋轉到目標向量(將輸入的 Z 消除)。

首先 Vectoring Mode 是由y_i決定旋轉方向，最後要得到 y 座標等於 0，以及新座標(x’, 0)及旋轉角度 Z，如果起始值的 y 小於 0，那代表要向逆時鐘轉(加上固定角度一)，才能讓新座標的 y 趨近於 0，同理，如果經過第一次旋轉後的座標

y 值大於 0，那此次旋轉便是向順時鐘轉(減上固定角度二)，經過多次疊代計算，最後座標的 y 趨近於 0，則完成，並輸出角度 Z。

Rotating Mode 的目的是將 Z 消除，每一次的計算是透過Z_i控制旋轉方向，起始值為 0，計算每次旋轉完的角度與 Z 的差值，若差值大於 0，代表下次旋轉的角度為負;若差值小於 0，代表下次旋轉的角度為正，才會有辦法將起始值 0 逼近角度 Z，進而將一個座標旋轉到欲旋轉的角度，或是兩個互相對應的旋轉，在進行矩陣運算時，由 Vectoring Mode 所產生的 Z，會傳給同一列的其他元素做旋轉，而其他元素透過 Rotating Mode，以及 Z，可做行與行之間的相對應旋轉，或是列相消。

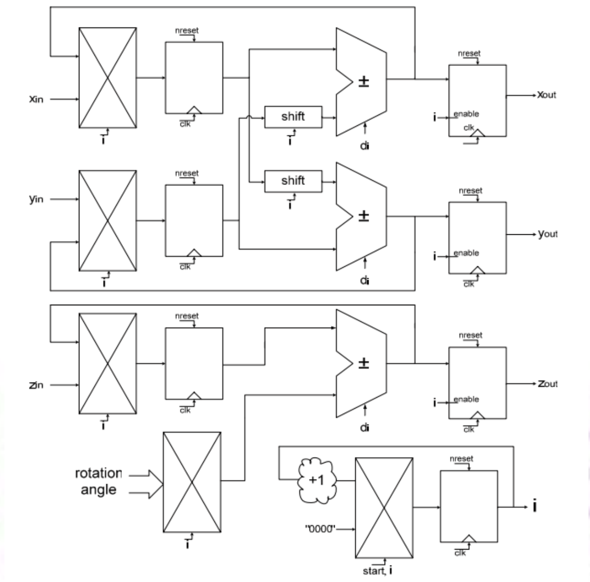


Figure 1. CORDIC - sequential architecture

Fig2. CORDIC 之架構

在結合以上提及之雙邊 Jacobi 與 CORDIC 技術，可以實現低複雜度之硬體 SVD 分解。以下給定一大小 2x2 的矩陣 **M**=$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$為例，我們所需旋轉的角度為 argtan($\frac{c+b}{d-a}$)，而每一次的旋轉需要用到 CORDIC 的兩種模式分別各兩次，即先決定好欲旋轉的角度，再旋轉目標向量。完整之過程可用下列關係表示。

$$\begin{cases} x_{i+1} = x_i + \delta_i * SHIFT_i(y_i) \\ y_{i+1} = y_i - \delta_i * SHIFT_i(x_i) \end{cases}$$

其中的$\delta_i = +1/-1$，分別代表為向左轉或向右轉，取決於目標角度與現在旋轉到的結果角度進行比較來決定。

給定一個 2x2 矩陣 G，$G = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$，在雙邊雅可比演算法中，G 的對角線矩陣可

被表示為

$$\begin{bmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{bmatrix}=\begin{bmatrix}cos\theta_L & sin\theta_L\\ -sin\theta_L & cos\theta_L\end{bmatrix}\begin{bmatrix}a & b\\ c & d\end{bmatrix}\begin{bmatrix}cos\theta_R & -sin\theta_R\\ sin\theta_R & cos\theta_R\end{bmatrix}=\frac{1}{2}\begin{bmatrix}A+C & B+D\\ B-D & C-A\end{bmatrix}$$

從兩個 0 可推出關係式，$\begin{cases}\theta_L+\theta_R=\tan^{-1}(\frac{c+b}{d-a})\\ \theta_L-\theta_R=\tan^{-1}(\frac{c-b}{d+a})\end{cases}$，將原式乘開後可得

$$\begin{cases}\begin{pmatrix}A\\ B\end{pmatrix}=\begin{bmatrix}cos\alpha & -sin\alpha\\ sin\alpha & cos\alpha\end{bmatrix}\begin{pmatrix}a-d\\ c+b\end{pmatrix}\\ \begin{pmatrix}C\\ D\end{pmatrix}=\begin{bmatrix}cos\beta & -sin\beta\\ sin\beta & cos\beta\end{bmatrix}\begin{pmatrix}a+d\\ b-c\end{pmatrix}\end{cases}$$

其中$\alpha=\theta_{SUM}=\theta_L+\theta_R$，$\beta=\theta_{DIFF}=\theta_L-\theta_R$。

運算時首先將矩陣的值輸入，得到和角與差角後相加減先得到兩個輸出θ_L、θ_R，接著便可以用兩個角度輸入向量旋轉模式得到特徵值。以下我們將基於上述的演算法原理，進行系統設計的規劃與後續的軟體模擬驗證。

軟體實作結果:

我們將透過 MSE(均方誤差)來比較經實作之 SVD 演算法與 MATLAB 計算的結果，MSE 是指參數估計值與參數真值之差平方的期望值，MSE 可以評價資料的變化程度，MSE 的值越小，說明預測模型描述實驗資料具有更好的精確度，以下表一為實驗範例資料。

表一 演算法模擬特徵值比較結果(以 4*4 矩陣為例)

	特徵值 1	特徵值 2	特徵值 3	特徵值 4
MATLAB	15.5095	1.3860	0.0501	0.0092
Jacobi	15.44083	1.25660	0.04082	0.00128

由表一可見，經我們實作之 SVD 演算法與 MATLAB 計算的結果進行比較的結果，以特徵值為計算 MSE 之標準

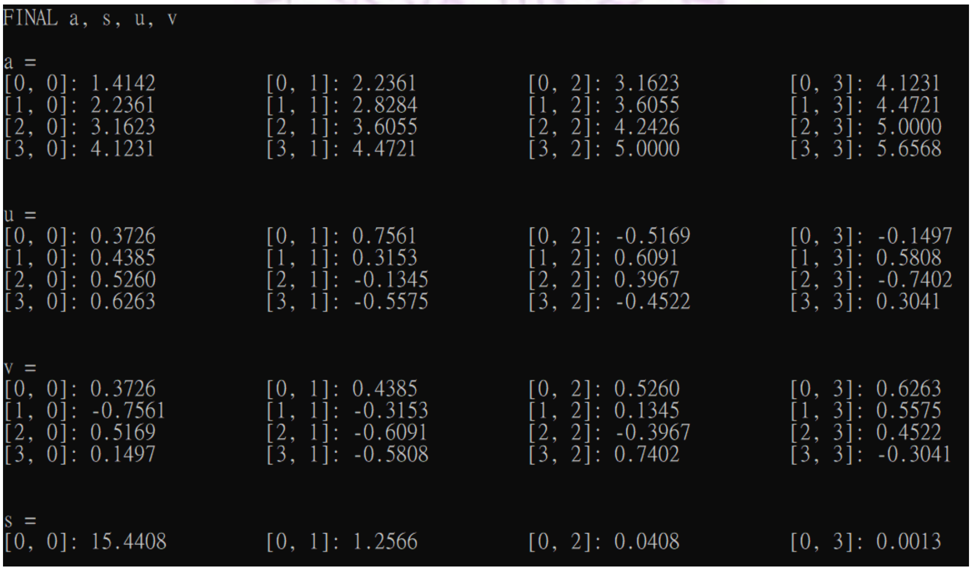
MSE

$$= \frac{(15.44083-15.5095)^2+(1.2566-1.386)^2+(0.04082-0.0501)^2+(0.00128-0.0092)^2}{4}$$

$$= \frac{0.0216}{4} = 5.4 * 10^{-3}$$

可見 MSE 非常接近 0，為誤差可接受範圍內。

Fig3. 經演算法計算之結果



```
FINAL a, s, u, v

a =
[0, 0]: 1.4142    [0, 1]: 2.2361    [0, 2]: 3.1623    [0, 3]: 4.1231
[1, 0]: 2.2361    [1, 1]: 2.8284    [1, 2]: 3.6055    [1, 3]: 4.4721
[2, 0]: 3.1623    [2, 1]: 3.6055    [2, 2]: 4.2426    [2, 3]: 5.0000
[3, 0]: 4.1231    [3, 1]: 4.4721    [3, 2]: 5.0000    [3, 3]: 5.6568

u =
[0, 0]: 0.3726    [0, 1]: 0.7561    [0, 2]: -0.5169   [0, 3]: -0.1497
[1, 0]: 0.4385    [1, 1]: 0.3153    [1, 2]: 0.6091    [1, 3]: 0.5808
[2, 0]: 0.5260    [2, 1]: -0.1345   [2, 2]: 0.3967    [2, 3]: -0.7402
[3, 0]: 0.6263    [3, 1]: -0.5575   [3, 2]: -0.4522   [3, 3]: 0.3041

v =
[0, 0]: 0.3726    [0, 1]: 0.4385    [0, 2]: 0.5260    [0, 3]: 0.6263
[1, 0]: -0.7561   [1, 1]: -0.3153   [1, 2]: 0.1345    [1, 3]: 0.5575
[2, 0]: 0.5169    [2, 1]: -0.6091   [2, 2]: -0.3967   [2, 3]: 0.4522
[3, 0]: 0.1497    [3, 1]: -0.5808   [3, 2]: 0.7402    [3, 3]: -0.3041

s =
[0, 0]: 15.4408   [0, 1]: 1.2566    [0, 2]: 0.0408    [0, 3]: 0.0013
```

a 為輸入之矩陣，u、v 為特徵向量，s 為以遞減排序之特徵值

Fig4. Matlab 計算之結果

```
S =

   15.5095         0         0         0
         0    1.3860         0         0
         0         0    0.0501         0
         0         0         0    0.0092


V =

   -0.3703    0.7251    0.1965   -0.5463
   -0.4363    0.3313    0.2080    0.8103
   -0.5238   -0.0658   -0.8485   -0.0373
   -0.6310   -0.6001    0.4452   -0.2086
```

右特徵向量(v)與以遞減排序之特徵值(s)

硬體的部分，則等待軟體實現後，計算此演算法經過完整的 SVD 所需之 CORDIC 模塊的數量，來看此演算法適不適合硬體上的實現，是不是為低複雜度的硬體友善演算法，由於時間上的限制，無法放進專題摘要內。

心得感想

起初，我們對於波束成形、通訊系統沒有相關的知識，在實作專題一時，不論是尋找適當演算法，還是整體衛星通訊系統的架構概念都不太熟悉，所以很難在 6G 衛星通訊這麼巨大的範疇下找出適合的方向，因此我們花了將近一個學期修課來學習相關的知識，並且 Survey 各式各樣的相關論文來尋找方向。經過一個學期，開始實作專題後，也遇上了很多難題，像是原本決定硬體實作時，發現相關課程都還沒有修，如果以課業繁重的程度加上自學的速度來看，硬體實作上會時間非常緊迫，所以臨時又改為軟體實現，但這樣也使得軟體實現的時間所剩不多，最後只能完成實數的部分，複數的部分就差臨門一腳，但非常感謝翁詠祿老師以及碩士班子毅學長的解惑，讓我們能夠處理這些難題。在這次的專題中，我們學習到了自我學習與搜尋資料的重要性，透過學習解決問題，並了解到團隊溝通合作的方式。