第二章　背景說明

2.1 語音合成系統設計

本論文基於實驗室前人發展的語音合成系統——「清華大學多媒體資訊檢索實驗室語音合成系統」（Multimedia Information Retrieval Lab Text-To-Speech System, MIR TTS），針對韻律產生器的部分，做進一步地研究與改良。本節先簡單說明此語音合成系統的語料庫設計與合成方式。

2.1.1 承載句設計

雖然以大量語料庫為基礎（Corpus-Based）的中文文句轉語音系統是現今國語的語音合成技術的主流，但是在片段接合上的不一致容易使自然度下降，可以參考本實驗室早期以大量語料庫為基礎的語音合成系統[14]。有別於主流的設計，我們改採用承載句（Carrier Sentence）語料庫[8]設計為基本的合成單元。

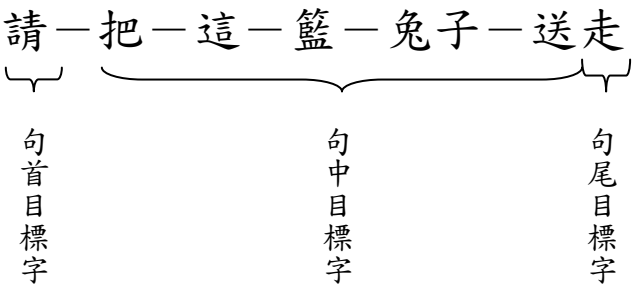
中文含 411 個基本單音節（Base Syllable），考慮聲調變化，中文所有可能的發音有 1465 個帶聲調音節（Tonal Syllable），以這帶聲調 1465 個音節以承載句的方式錄製，每句承載句含三個目標字（Target Word），分別在句首、句中和句尾，句子設計如下：

"㖞"是一個常見的字，一般人常把**"㖞"**字掛在嘴邊，講話時動不動就提到**"㖞"**

每句把句首、句中和句尾三個目標字取出來，作為基本合成單元，此錄製方式可以減少合成時音節間的不一致性，使合成結果更為自然。

2.1.2 聲音的合成方式

我們在合成時，單元選擇方式，依單元在句中出現的位置選擇合適的目標字，例如：



本系統的合成是在時域上對訊號進行處理，我們調整音長、音量和基週頻率這三個韻律參數，測試其對自然度的幫助。我們先用基頻同步累加法（PSOLA）來調整基週頻率，再用波形相似性疊加法（WSOLA）來調整音長。

PSOLA 是個常見的語音合成方法之一，其詳細介紹可參考[12]。這個方法可以對基週頻率，也就是音高，做彈性的調整，而能保留聲音的特性。然而，使用 PSOLA 之前需要有 Pitch Mark 資訊，於是我們使用實驗室學長提出的方法[15]取得基週頂點標記（Pitch Mark）。因為能保持基週頂點標記上主要部分的特徵，所以對於原先頻譜的破壞就相對減低很多，因此音色能保留原先聲音的特色。

WSOLA 則是針對音長的處理，因為音高與基週頻率皆不改變，而是要改變聲音的長短，所以並不需要基週頂點標記的資訊，是用平均幅度差函數法（Average Magnitude Difference Function, AMDF）進行音框比對，找出最適合的音框作波形疊加。更詳細的 WSOLA 介紹請參考[13]。

2.2 迴歸模型簡介

我們嘗試使用多種統計迴歸方式，來求得韻律參數的最佳預測結果，包含線性迴歸、支撐向量機，與類神經網路。本節將分別介紹這三種迴歸工具。

2.2.1 線性迴歸

線性迴歸是數學上一種曲線擬合的方式，可用來找出兩個或兩個以上計量變數間的關係，並進而從一群變數中可以預測資料趨勢。我們建立一個數學模型，並依此模型來進行預測。

在這裡介紹本論文使用的「利用最小平方法（Least Squared Error Estimation, LSE）的線性迴歸」。舉一個簡單的例子，假設 y 為此模型的輸出，x 為此模型的輸入。此外，x 與 y 皆為一維的數列，我們假設數學模型為一個二次線性方程式，如下式：

$$y = f(x) = a_0 + a_1 x + a_2 x^2 \qquad (1)$$

假設資料量有 N 筆，我們希望此 N 筆觀察點皆通過此二次方程式，則可列式如下：

$$\begin{aligned} y_1 &= a_0 + a_1 x_1 + a_2 {x_1}^2 \\ y_2 &= a_0 + a_1 x_2 + a_2 {x_2}^2 \\ &\cdots \\ y_N &= a_0 + a_1 x_N + a_2 {x_N}^2 \end{aligned} \qquad (2)$$

可以觀察到，上述有多筆方程式，卻只有 3 個未知數，這種情形大多是無解的，只能求出最逼近解。利用線性代數的觀念，上式可表示成：

$$\underbrace{\begin{bmatrix} 1 & x_1 & x_1^{\,2} \\ 1 & x_2 & x_2^{\,2} \\ \cdots & & \\ 1 & x_N & x_N^{\,2} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}}_{\theta} \cong \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_3 \end{bmatrix}}_{y} \qquad (3)$$

其中 A、y 為已知，θ 未知向量，欲求解最佳 θ 值，使得左式最接近向量 y。因為 A 不保證為可逆矩陣，我們先乘上 A 的轉置矩陣使之保證可逆，再乘上他們的反矩陣，以求得 θ 值。推導過程如下：

$$\begin{aligned} & A\theta \cong y \\ & A^T A\theta \cong A^T y \\ & \left(A^T A\right)^{-1} A^T A\theta \cong \left(A^T A\right)^{-1} A^T y \\ & \Rightarrow \theta \cong \left(A^T A\right)^{-1} A^T y \end{aligned} \qquad (4)$$

2.2.2 類神經網路

類神經網路為模擬生物腦的神經網路學習機制的一種系統[9]，利用運算元模擬生物神經元，以得到學習、記憶、回想、平行等處理能力。類神經網路的架構，由小至大，分別是運算元或稱為處理單元（Processing Element）、層（Layer）、和網路（Network），如圖 3 所示。

運算元經由網路可以取得輸入，經過三種運算之後，再經由網路傳送給別的神經元，這些運算分別說明如下：

（1） 集成函數（Summation）：把透過網路傳來的訊息集成。

（2） 作用函數（Activity function）：將集成函數值與處理單元目前狀態綜合。

（3） 轉換函數（Transfer function）：將作用函數輸出值轉換成神經元的輸出。

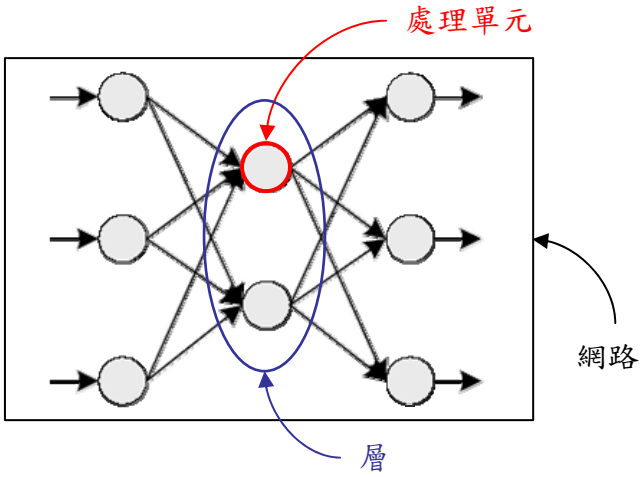


圖 3　類神經網路架構圖

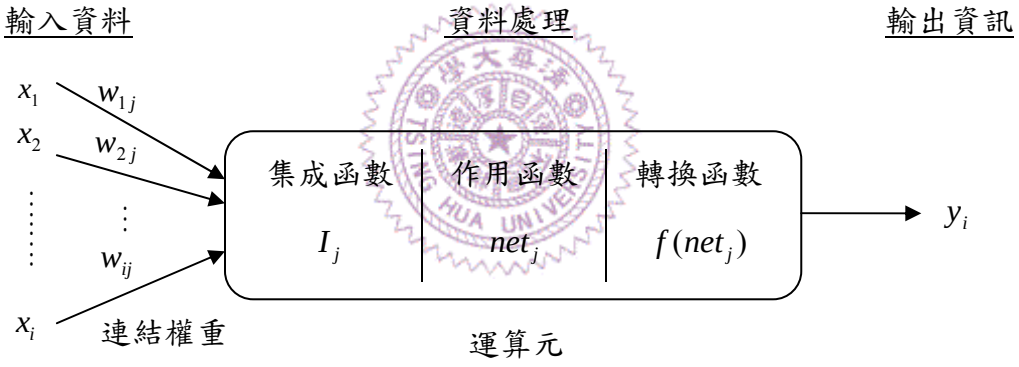


圖 4　類神經網路的運算元模型

對於上述三種函數，在這邊只列舉本論文使用的函數如下。

- 集成函數 I_j：加權乘積和 $I_j = \sum_i W_{ij} X_i$
- 作用函數 net_j：直接使用集成函數輸出 $net_j^n = I_j^n$
- 轉換函數 Y_j：雙曲線正切函數（Hyperbolic Tangent Function）

$$Y_j = \frac{e^{net_j} - e^{-net_j}}{e^{net_j} + e^{-net_j}}$$

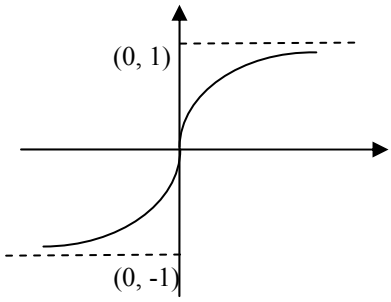


圖 5　雙曲線正切函數圖

而層可視為數個神經元的集合，關於層的設計又可分為下列三種，而我們使用的是正規化輸出。

（1） 正規化輸出：正規化為單位長度向量再輸出

（2） 競爭化輸出：高強度的處理單元為 Winner，令其為 1，其餘為 0

（3） 競爭化學習：高強度的處理單元為 Winner，只調整他的網路

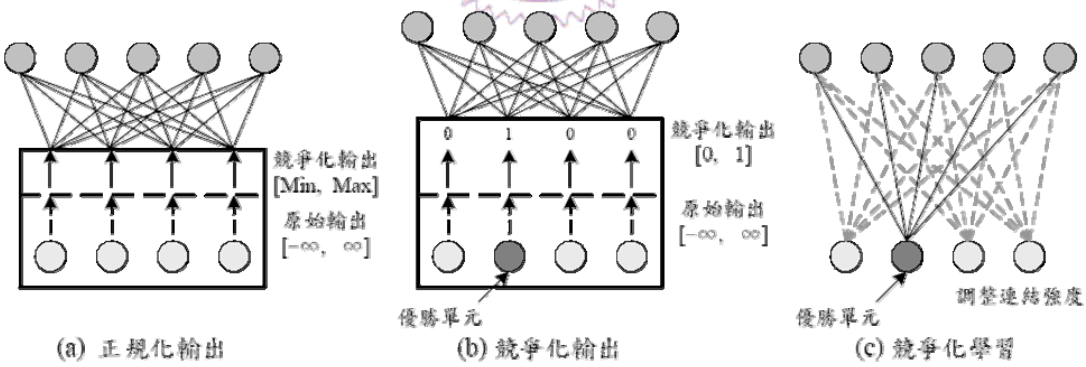


圖 6　正規化輸出、競爭化輸出、與競爭化學習

類神經網路最大的單位是由處理單元、連結權重與層所構成的，稱之為網路。網路依它的架構，又可分為兩種：

（1）前向式架構（Forward）：處理單元分層排列，依序為輸入層、隱藏層、輸出層，每一 layer 只接受前一 layer 的輸出作為輸入。

（2）回饋式架構（Feedback）：輸出層回饋到輸入層，或層內各處理單元有連結，或處理單元不分層排列，其均可相互連結。

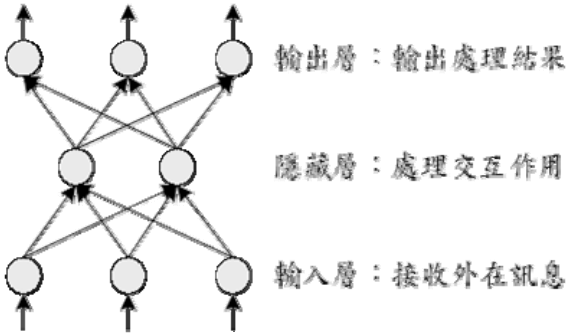


圖 7　前向式網路架構圖

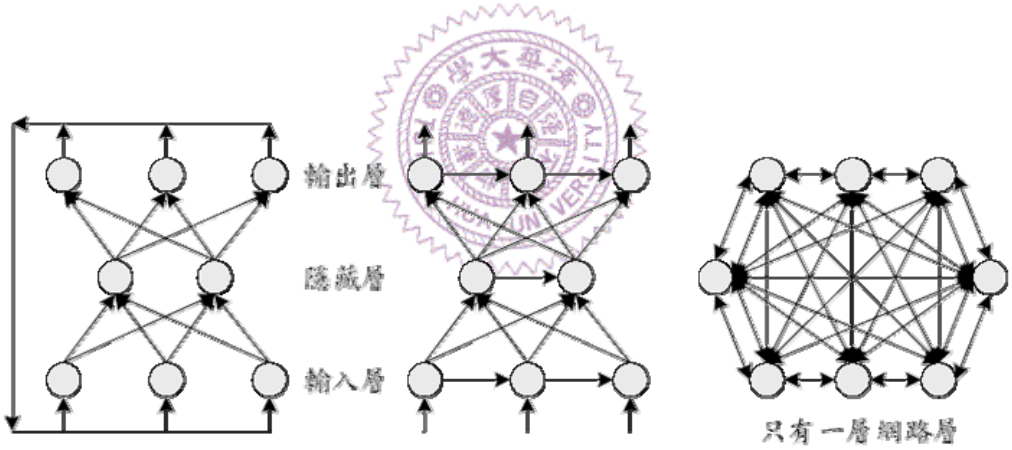


圖 8　回饋式網路架構圖

網路的處理階段，分為學習過程與回想過程。網路的學習目標即是讓網路的能量函數，或者是誤差函數為最小。依據不同的學習演算法遞迴調整網路之連結權重值，最後目的是要求出一個最適合之網路連結權重值。依學習演算法對網路分類，可分為以下四類。

（1）監督式學習（Supervised Learning）：有輸入變數，也有輸出變數。。

（2）無監督式學習（Unsupervised Learning）：只有輸入變數，而需推論它與哪些訓練範例屬同一聚類。

（3）聯想式學習（Associate Learning）：只有不完整的狀態變數值，而需推論其完整的狀態變數值。

（4）最適化應用（Optimization Application）：對一問題決定其設計變數值，使其在滿足設計限制下，使目標達最佳狀態。

我們用來訓練韻律參數的情況，因為目標向量為已知的韻律參數，所以是屬於第一類，監督式學習。其誤差函數為：$E=\frac{1}{2}\sum_{j}\left(T_j-Y_j\right)^2$ 。

將學習過程得到的網路連結權重組合，作為回想演算法之記憶參數，然後依回想演算法，即可以輸入資料決定網路輸出資料。

在類神經網路中，最具代表性，最普遍被使用的是倒傳遞網路（BPN）的學習模式。其基本原理是使用可微分的轉換函數，利用最陡坡降法（The Steepest Descent Method）的觀念，修改網路的權重值，將誤差函數予以最小化。並且加入隱藏層，使網路可以表現神經元之間的交互影響。標準的倒傳遞類神經網路的架構如圖 9。

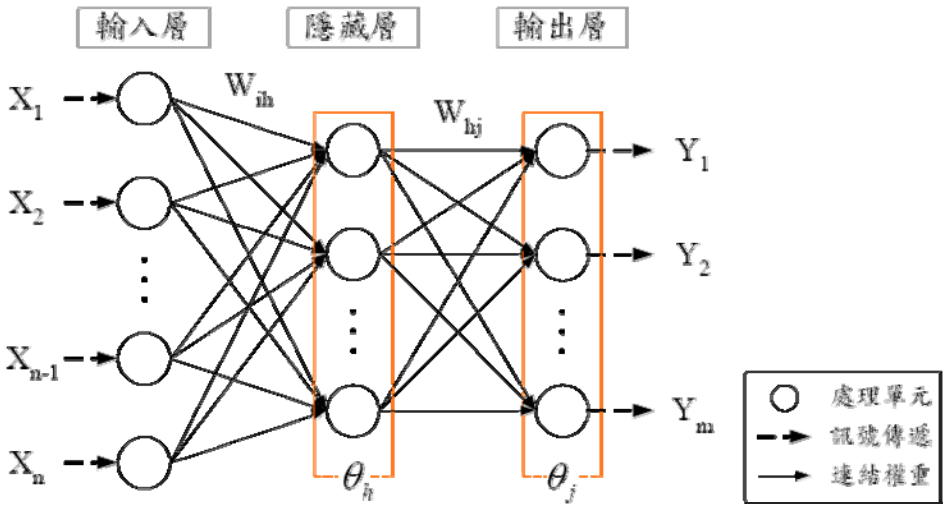


圖 9　倒傳遞類神經網路架構示意圖

2.2.3 支撐向量機

支撐向量機（Support Vector Machine），簡稱 SVM，是由 Vladimir Vapnik 在 1979 年開始研究的一種分類方法[16]。假設一簡單的兩類別分類問題，給予訓練向量 x，輸出值為 y。其中 y 的定義如下：

$$y = \begin{cases} 1 & , \mathrm{x} \in \mathrm{class1} \\ -1 & , \mathrm{x} \in \mathrm{class2} \end{cases} \quad (5)$$

應用在分類時，SVM 就是建立一個超平面（Hyper-Plane）作為一個分類的面。超平面的表示式子如下：

$$f(x) = w^T x + b = 0 \quad (6)$$

$$\begin{cases} w^T x + b > 0 & , \mathrm{y} = 1 \\ w^T x + b < 0 & , \mathrm{y} = -1 \end{cases}$$

根據超平面的定義，我們可以重新定義：

$$\begin{cases} w^T x + b \geq 1 & , \mathrm{y} = 1 \\ w^T x + b \leq -1 & , \mathrm{y} = -1 \end{cases} \quad (7)$$

我們要找到這個最佳的超平面，即是找到最佳的 w 與 b 值，使得 margin 最大，而 margin 的定義是每個類別裡的資料點，離其他的類別的資料點的最小距離。該超平面的示意如圖 10。

資料點 x 到超平面 $f(x)$ 的距離表示如下：

$$d(w,b,x) = \frac{w^T x + b}{\|w\|} \quad (8)$$

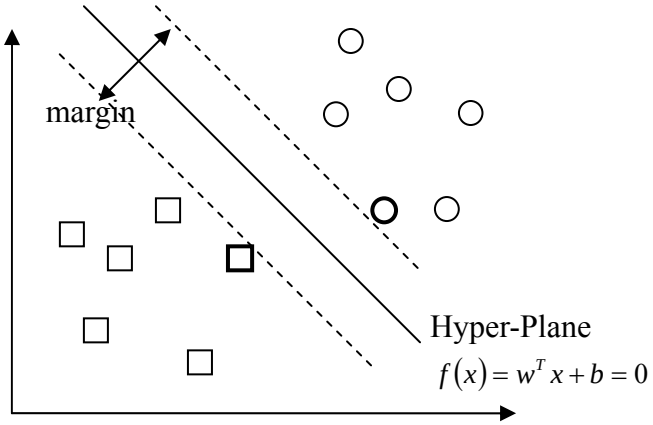


圖 10　支撐向量機的超平面示意圖

欲求得最佳超平面，也就是求最大 margin，而最大 margin 則可表示成如下式。

$$M(w,b) = \min_{y=1} d(w,b,x) + \min_{y=-1} d(w,b,x) = \frac{2}{\|w\|} \qquad (9)$$

所以要求得最佳 w 與 b 值，也就是求最小 $\|w\|$，利用 Lagrange Multiplier 求得最佳解，其詳細求解方式請參考[11]。

而資料群有時是線性不可分割的，因此我們必須將觀察點映射（Map）到另一更高維度的特徵空間，使其成為線性可分割的問題，再求得其解。因此定義了核心函數（Kernel Function），用來對資料群作映射。定義如下式：

$$K(x,y) = \Phi(x) \cdot \Phi(y) \qquad (10)$$

其中 Φ 為將資料映射到比原先更高維度的特徵空間的函數，支撐向量機中定義的核心函數如下列：

- Simple Dot: $K(x,y) = x \cdot y$
- Vovk's Polynomial: $K(x,y) = (1 + x \cdot y)^p$

- Radial Basia Function: $K(x,y) = \exp(\frac{-\|x-y\|^2}{2\sigma^2})$
- Sigmoid Kernel: $K(x,y) = \tanh(kx \cdot y - \Theta)$
- Splines Kernel: $K(x,y) = 1 + (x \cdot y) + \frac{1}{2}(x \cdot y)\min(x \cdot y) + \frac{1}{6}\min(x \cdot y)^3$

以上簡單的描述 SVM 用於分類時的作法，而我們使用的是 SVM 的迴歸分析，處理迴歸與分類的不同，在於分類是預測出少數個固定的值，而迴歸可以看成是很細的分類器，並且容許一個數值之內的誤差量。所以在這邊定義一個 Loss Function[17]，又叫作 Error Function，而我們使用的 Error Function 為 ε-Insensitive，是由 Vapnik[18]提出最適合用於 SVM，如圖 11。

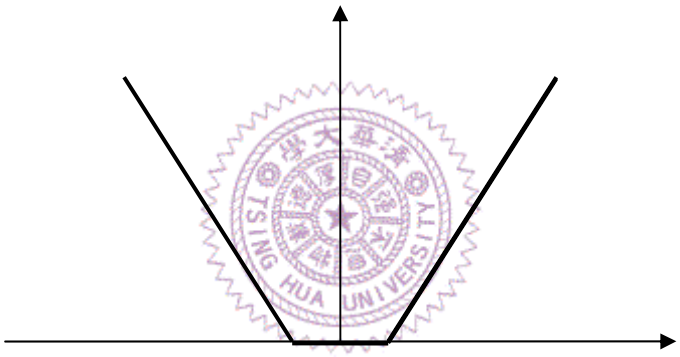


圖 11　ε-Insensitive Loss Function 示意圖

ε-Insensitive Loss Function 的區域表示如下式：

$$|y - f(x,w)|_\varepsilon = \begin{cases} 0 & ,\text{if } |y - f(x,w)| \le \varepsilon \\ |y - f(x,w)| & ,\text{otherwise} \end{cases} \qquad (11)$$

如果 Loss Function 等於 0，表示預測值 $f(x,w)$ 與實際值 y 的距離小於 ε，如圖 12。

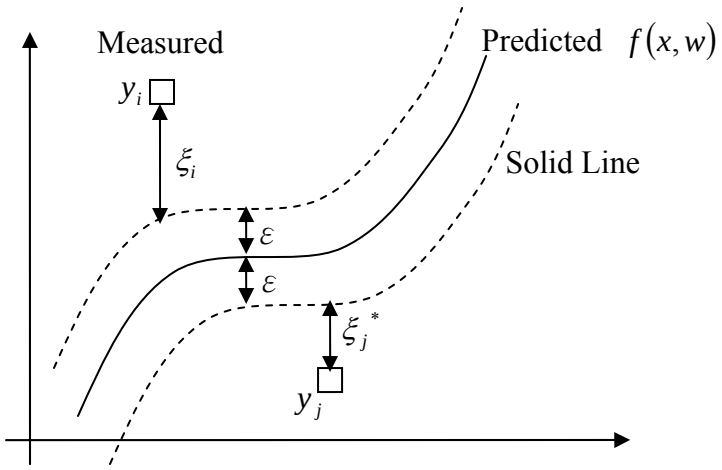


圖 12　一維的支撐向量迴歸

定義 Slack Variables ξ 與 ξ^* 如下：

$$|y - f(x,w)| - \varepsilon = \begin{cases} \xi & \text{, if } y \text{ is above the } \varepsilon \text{ tude} \\ \xi^* & \text{, if y is below the } \varepsilon \text{ tude} \end{cases} \quad (12)$$

定義 Empirical Risk 為平均 $|y - f(x,w)|_\varepsilon$，與分類問題相較，我們不僅要將 $\|w\|$ 最小化，也要最小化 Empirical Risk。Empirical Risk 如下列所示：

$$R_{w,\xi,\xi^*} = \frac{1}{2}\|w\|^2 + C(\sum_{i=1}^{N}\xi + \sum_{i=1}^{N}\xi^*) \quad (13)$$

把最佳化問題轉換為 Lagrange 問題，利用 Lagrange Multiplier 可求得最佳解，詳細求解過程請參考[11]。