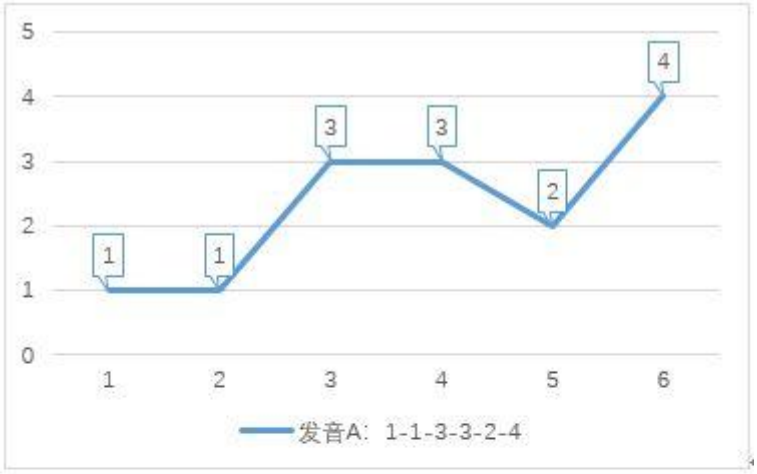
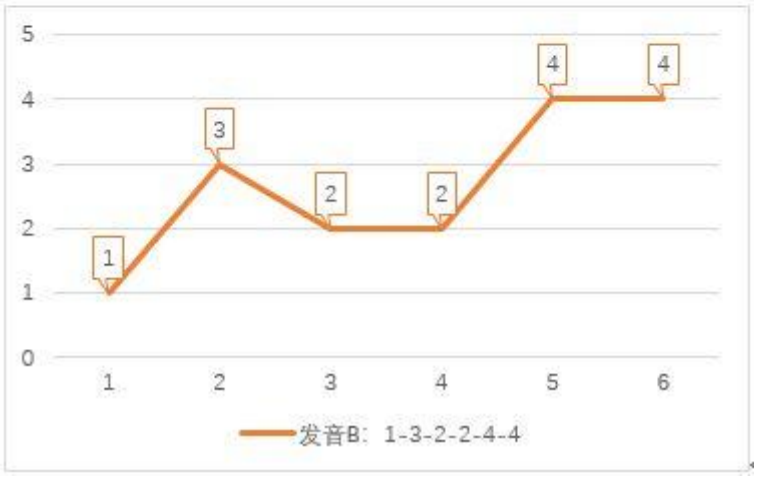
動態時間規整（DTW）算法簡介

DTW 最初用於識別語音的相似性。我們用數字表示音調高低，例如某個單詞發音的音調為 1-3-2-4。現在有兩個人說這個單詞，一個人在前半部分拖長，其發音為 1-1-3-3-2-4；另一個人在後半部分拖長，其發音為 1-3-2- 2-4-4。





現在要計算 1-1-3-3-2-4 和 1-3-2-2-4-4 兩個序列的距離（距離越小，相似度越高）。因為兩個序列代表同一個單詞，我們希望算出的距離越小越好，這樣把兩個序列識別為同一單詞的概率就越大。

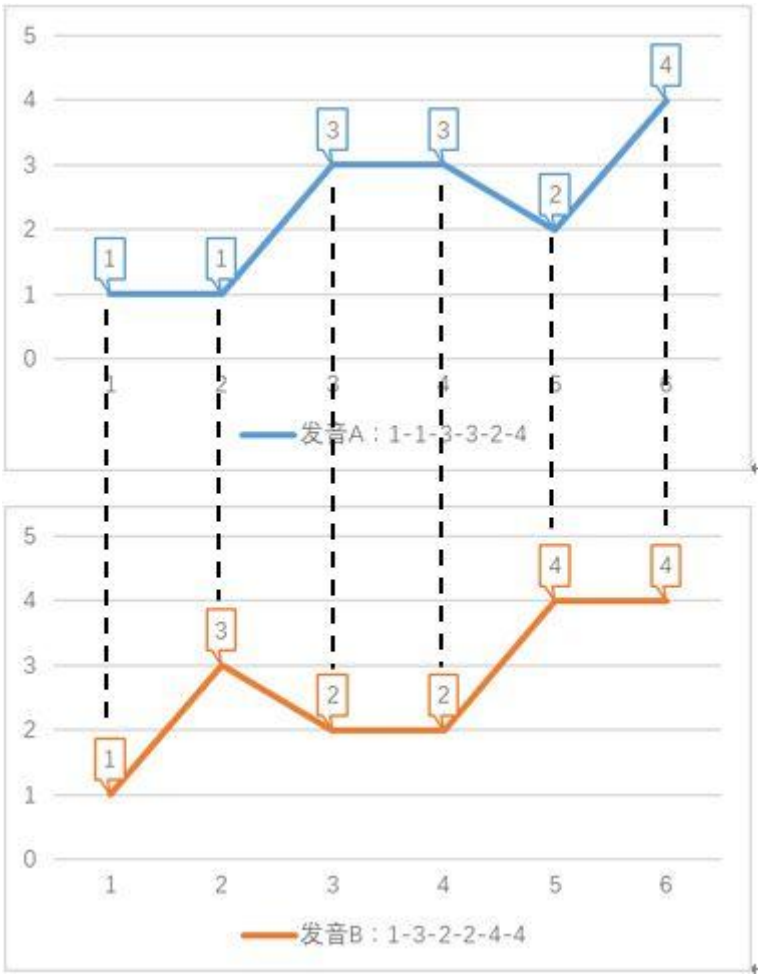
先用傳統方法計算兩個序列的歐幾里得距離，即計算兩個序列各個對應的點之間的距離之和。

距离之和

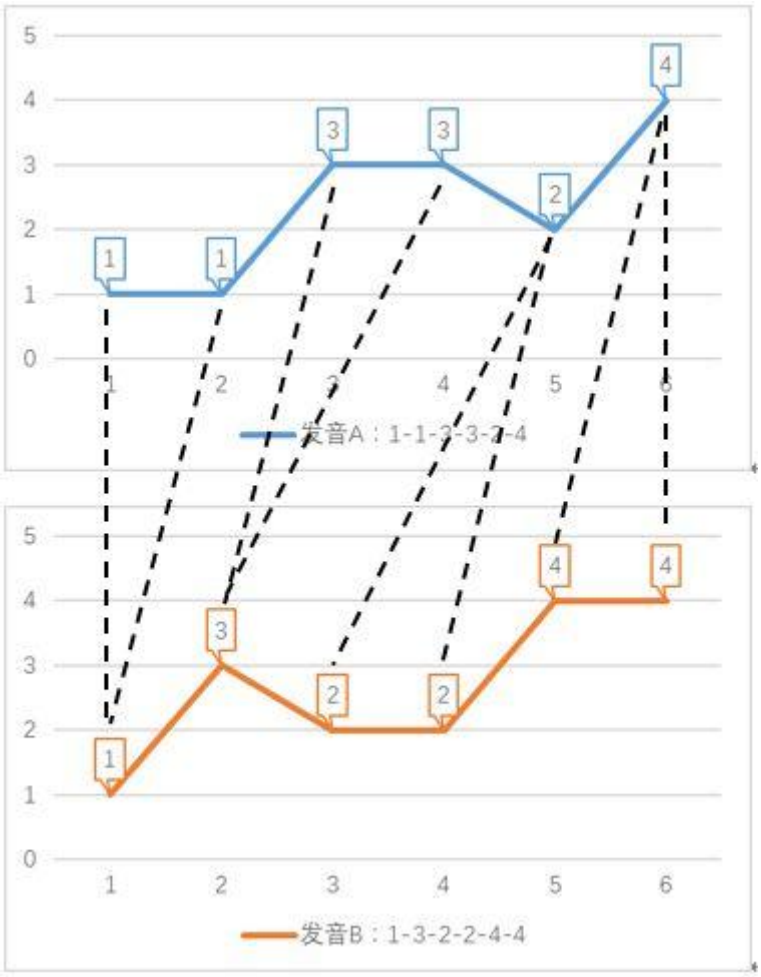
= |A(1)-B(1)| + |A(2)-B(2)| + |A(3)-B(3)| + |A(4)-B(4)| + |A(5)-B(5)| + |A(6)-B(6)|

= |1-1| + |1-3| + |3-2| + |3-2| + |2-4| + |4-4|

= 6



如果我們允許序列的點與另一序列的多個連續的點相對應（相當於把這個點所代表的音調的發音時間延長），然後再計算對應點之間的距離之和。如下圖：B(1)與 A(1)、A(2)相對應，B(2)與 A(3)、A(4)相對應，A(5)與 B(3)、B(4)相對應，A(6)與 B(5)、B(6)相對應。



這樣的話，

距离之和

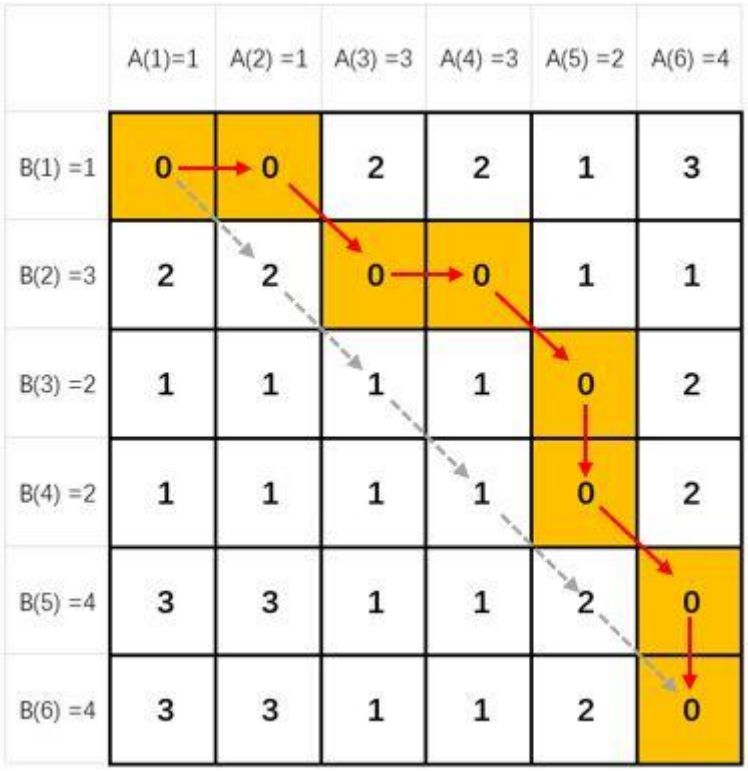
= |A(1)-B(1)| + |A(2)-B(1)| + |A(3)-B(2)| + |A(4)-B(2)| + |A(5)-B(3)| + |A(5)-B(4)| + |A(6)-B(5)| + |A(6)-B(6)|

= |1-1| + |1-1| + |3-3| + |3-3| + |2-2| + |2-2| + |4-4| + |4-4|

= 0

我們把這種“可以把序列某個時刻的點跟另一時刻多個連續時刻的點相對應”的做法稱為時間規整（Time Warping）。

現在我們用一個 6*6 矩陣 M 表示序列 A（1-1-3-3-2-4）和序列 B（1-3-2-2-4-4）各個點之間的距離，M(i, j)等於 A 的第 i 個點和 B 的第 j 個點之間的距離，即



我們看到傳統歐幾里得距離裡對應的點：

- A(1)-B(1)
- A(2)-B(2)
- A(3)-B(3)
- A(4)-B(4)
- A(5)-B(5)
- A(6)-B(6)

它們正好構成了對角線，對角線上元素和為 6。

時間規整的方法裡，對應的點為：

- A(1)A(2)-B(1)

- A(3)A(4)-B(2)
- A(5)-B(3)B(4)
- A(6)-B(5)B(6)

這些點構成了從左上角到右下角的另一條路徑，路徑上的元素和為 0。

因此，DTW 算法的步驟為：

1. **計算兩個序列各個點之間的距離矩陣。**
2. **尋找一條從矩陣左上角到右下角的路徑，使得路徑上的元素和最小。**

我們稱路徑上的元素和為路徑長度。那麼如何尋找長度最小的路徑呢？

矩陣從左上角到右下角的路徑長度有以下性質：

1. 當前路徑長度= 前一步的路徑長度+ 當前元素的大小
2. 路徑上的某個元素(i, j)，它的前一個元素只可能為以下三者之一：

a)左邊的相鄰元素(i, j-1)

b)上面的相鄰元素(i-1, j)

c)左上方的相鄰元素(i-1, j-1)

假設矩陣為 M，從矩陣左上角(1,1)到任一點(i, j)的最短路徑長度為 Lmin(i, j)。那麼可以用遞歸算法求最短路徑長度：

起始條件：

遞推規則：

遞推規則這樣寫的原因是因為當前元素的最短路徑必然是從前一個元素的最短路徑的長度加上當前元素的值。前一個元素有三個可能，我們取三個可能之中路徑最短的那個即可。