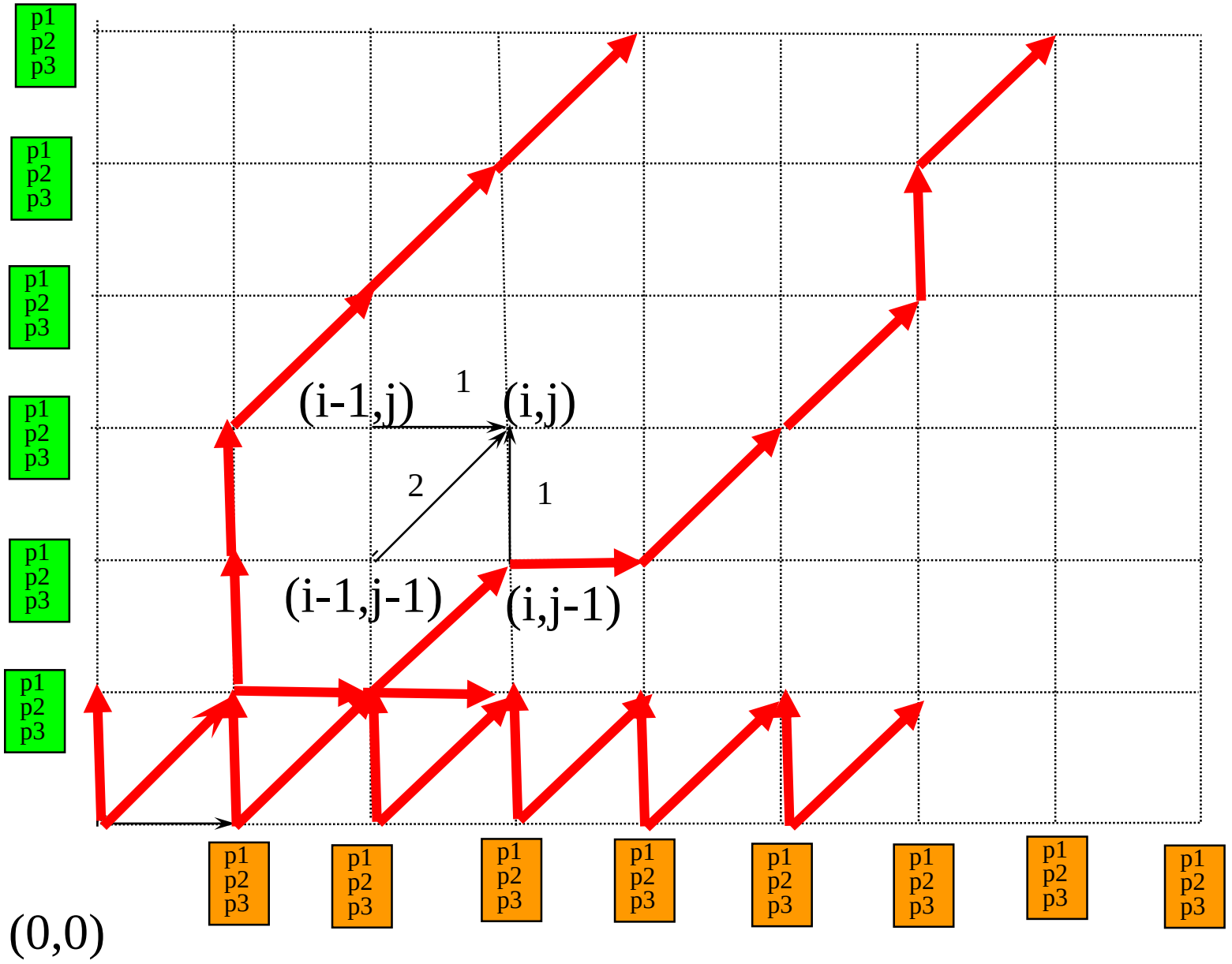
**Tarea**: Realizar la búsqueda de un segmento corto de audio en un audio largo

**Datos**: Se proporcionan el audio en el que buscar y varios segmentos cortos. Tods están ya paremetrizados por lo que son secuencias de vectores de parámetros.

**Algoritmo**: Se usará SDTW, tal como se explica a continuación. Se puede hacer una versión sencilla (sin normalizar el coste), o una versión más ajustada en la que en cada punto se compara el coste acumulado dividido por la longitud del camino. Para ello hay que almacenar en cada posición de la matriz: coste, longitud del camino, y punto inicial de ese camino.

**Resultado**. Los puntos en los que se ha encontrado. Se sugiere que se genera una lista ordenada de los primeros 10 puntos (o incluso más) que son candidatos a contener la pronunciación.

# Problema: Alineamiento temporal



## Búsqueda sin normalizar

Función SDTW (I,J:**N**):**R**;
var T: matriz[0..I,0..J] de **R**;

**para** i=0 **hasta** I **hacer** T[i,0]=0 **fpara**
**para** j=0 **hasta** J **hacer** T[0,j]=T[0,j-1]+d(0,j) **fpara**
**para** i=1 **hasta** I **hacer**
  **para** j=1 **hasta** J **hacer**

$$T[i,\ j] = \min\begin{Bmatrix} T[i-1,\ j] + d(i,\ j), \\ T[i,\ j-1] + d(i,\ j), \\ T[i-1,\ j-1] + d(i,\ j) \end{Bmatrix}$$

  **fpara**
**fpara**
Devolver lista_min (T[i,J])
**fin**

**Búsqueda normalizando:**

Cada elemento de la Matriz T tiene 3 componentes

- El coste acumulado
- La longitud del camino (el número de transiciones usadas)
- La posición i inicial del camino mejor hasta ese punto (es decir de la línea 0)

La minimización compara CosteAcumulado/longituddelCamino

Función SDTW (I,J:**N**):**R**;
var T: matriz[0..I,0..J] de **R**;

**para** i=0 **hasta** I **hacer** T[i,0]=0 **fpara**
**para** j=0 **hasta** J **hacer** T[0,j]=T[0,j-1]+d(0,j) **fpara**
**para** i=1 **hasta** I **hacer**
  **para** j=1 **hasta** J **hacer**

$$T[i,\ j] = \min\begin{Bmatrix} T[i-1,\ j] + d(i,\ j), \\ T[i,\ j-1] + d(i,\ j), \\ T[i-1,\ j-1] + d(i,\ j) \\ \end{Bmatrix}$$

  **fpara**
**fpara**
Devolver lista_min (T[i,J])
**fin**