Parte 3

Pregunta 1

Dada una matriz A de rango r, es posible ver cómo la SVD se puede utilizar para determinar una aproximación L_k de A de rango reducido, esto es, que su rango sea a lo sumo k, donde $k < r$. Esta aproximación se relaciona con la SVD reducida. En efecto, si $A = U\Sigma V^*$ y $r(A) = r$, se tiene que $A = U_r\Sigma_r V_r^* = \sigma_1 u_1 v_1^* + \sigma_2 u_2 v_2^* + ... + \sigma_r u_r v_r^*$. Luego, la mejor aproximación de A de rango a lo sumo k, está dada por: [REFA1]

$$L_k = \sigma_1 u_1 v_1^* + \sigma_2 u_2 v_2^* + ... + \sigma_k u_k v_k^* = U_k \Sigma_k V_k^*, k < r$$

Así, $min_{r(Y)\leq k}\|A - Y\|_{fr}^2 = \|A - L_k\|_{fr}^2$, donde $\|\cdot\|_{fr}$ denota a la norma de Frobenius . Este resultado se conoce como el teorema de Eckart-Young. El algoritmo iterativo GoDec, presentado, emplea la SVD para determinar dos matrices L y S tales que $AS + L$, donde L es de rango reducido y S es una matriz dispersa o rala (la mayor parte de sus entradas son ceros) [REFA1].

Se busca determinar una matriz L de rango a lo sumo $k, k < r$, y una matriz dispersa S con cardinalidad a lo sumo c_0 (parámetro de entrada en el algoritmo). Se suele escribir $A = L + S + G$, donde G es una matriz de ruido que modela el error de aproximación. En cada iteración del algoritmo (ver figura A3) se definen matrices L_t y S_t [REFA1].

Las sucesiones $\{L_t\}$ y $\{S_t\}$ convergen linealmente a un mínimo local del problema de optimización. La función P_{c_0} en dicho algoritmo (ver línea 6 de la figura A3) recibe una matriz densa A' y retorna otra matriz dispersa de cardinalidad a lo sumo c_0, del mismo tamaño de A', la cual se construye manteniendo de A' las c_0 entradas de mayor magnitud (en valor absoluto) y las demás entradas iguales a cero [REFA1].

La figura A4, muestra los resultados numéricos del algoritmo de la figura A3 aplicado a una matriz de tamaño 5x4. Es posible apreciar las matrices L y S resultantes después de 8 iteraciones del algoritmo. Finalmente se demuestra la rapidez de convergencia del algoritmo para alcanzar una aproximación de tal calidad, con tan sólo un 0.000089948 de error obtenido [REFA1].

Algoritmo 1 GoDec Clásico

Entrada: k, c_0, $A_{m\times n}$, ε.

Salida: L, S.

$L_0 := A$, $S_0 := \mathbf{0}_{m\times n}$, $t := 0$
mientras Verdadero **hacer**
  $t := t + 1$.
  $[U, \Sigma, V^*] = \mathrm{svd}(A - S_{t-1})$.
  $L_t := U_k \Sigma_k V_k^*$.
  $S_t = \mathcal{P}_{c_0}(A - L_t)$.
  $E_t := \|A - L_t - S_t\|_{fr}^2 / \|A\|_{fr}^2$.
  si $|E_t - E_{t-1}| < \varepsilon$ **entonces**
    salir.
  fin si
fin mientras

Figura A3. Pseudocódigo del algoritmo GoDec clásico. [REFA1]

```
>> p3_p1

A =

   19   10    8   11
  -15    7    4  -13
   -5   -8   17    2
   21   11   -6   23
   22  -12    9    1

L =

   18.8131    -1.6657     2.1146    11.2699
  -15.2369    -2.9439     1.0850   -12.6579
    1.1935    -8.0438     5.3072    -5.8127
   20.9485    10.9461    -5.9874    23.0756
   22.0453   -11.9495     8.9931     4.9849

S =

    0.00000   11.66568    5.88541    0.00000
    0.00000    9.94393    2.91500    0.00000
   -6.19351    0.00000   11.69277    7.81271
    0.00000    0.00000    0.00000    0.00000
    0.00000    0.00000    0.00000   -3.98487

error =  0.000089948
```

Figura A4. Resultados obtenidos al aplicar el algoritmo GoDec claśico implementado en GNU Octave a una matriz de 5x4.