SVD: descomposición en valores singulares: resultados, aplicaciones, cálculo

Héctor Manuel Mora Escobar
Universidad Central, Bogotá
hectormora@yahoo.com

Julio de 2011

1 Definiciones

Sea $A \in \mathbb{C}^{m\times n}$. Su SVD (singular value decomposition), *descomposición en valores singulares*, es:

$$A = USV^* \tag{1}$$

donde,
$U \in \mathbb{C}^{m\times m}$ es unitaria ($UU^* = I_m$),
$S \in \mathbb{R}^{m\times n}$ es "diagonal", con "diagonal" no negativa y no creciente,
$V \in \mathbb{C}^{n\times n}$ es unitaria.

En lo que sigue, sea $p = \min\{m, n\}$. Usualmente la definición de matriz diagonal se aplica únicamente a matrices cuadradas. En este documento, una matriz S rectangular es *diagonal* si $s_{ij} = 0$ para $i \neq j$. Su diagonal está compuesta por los elementos diagonales, es decir, los elementos s_{ii}, $i = 1, ..., p$. Su diagonal es no negativa si $s_{ii} \geq 0$ para $i = 1, ..., p$. Su diagonal es no creciente si $s_{11} \geq s_{22} \geq \cdots \geq s_{pp}$.

El resultado fundamental indica que, *toda matriz tiene descomposición en valores singulares.*

Los valores s_{ii} se llaman los valores singulares de A (a veces los restringen a los positivos). Frecuentemente los valores singulares se denotan $\sigma_1 = s_{11}$, $\sigma_2 = s_{22}$, ...

Ejemplo 1.

```
A  =
   2  + 2 i    3  - i     4  + 3 i
   5           6  + i     7  - 2 i

V  =
  -0.3878638 + 0.2684640 i  -0.0941940  -0.1450840 i   0.1849632 + 0.8446038 i
  -0.4517127 + 0.2478474 i   0.1193960 + 0.8382315 i  -0.0681730  -0.1139840 i
  -0.6663825 + 0.2606731 i   0.0307402  -0.5022408 i   0.0478029  -0.4821909 i

S  =
   12.116154    0            0
   0            3.3464638    0

U  =
  -0.4842705 + 6.060D-18 i   0.8749183 + 4.397D-17 i
  -0.7461752 + 0.4568420 i  -0.4130107 + 0.2528637 i
```

Si A es real, U y V son ortogonales.

En lo que sigue, supondremos, mientras no se diga lo contrario, que A **es real** y que $m \geq n$. Cuando $m \leq n$ se considera A^{T}.

Ejemplo 2.

```
A  =
    1       2       3       4       5
    6       7       8       9      10
   11      12      13      14      15

U  =
  -0.2016649     0.8903171     0.4082483
  -0.5168305     0.2573316    -0.8164966
  -0.8319961    -0.3756539     0.4082483

S  =
   35.127223     0              0       0       0
   0             2.4653967      0       0       0
```

```
    0              0              0      0      0

 V  =
   -0.3545571   -0.6886866   -0.5407780   -0.1172445    0.3062890
   -0.3986964   -0.3755545    0.7159048    0.4167279    0.1175510
   -0.4428357   -0.0624224    0.2035532   -0.6740223   -0.5515977
   -0.486975     0.2507097   -0.3917089    0.5668389   -0.4746133
   -0.5311143    0.5638418    0.0130289   -0.1923000    0.6023711
```

Hay varias demostraciones de la existencia. Se puede hacer por inducción utilizando la descomposición espectral de $A^{\mathrm{T}}A$ [NoD88]. También se puede hacer usando la descomposición de Schur [MaN99].

1.1 Otros tipos de SVD

- SVD *delgada* (thin) o *económica*:

$$\begin{aligned} A &= U_n S_n V^{\mathrm{T}} \\ U_n &= U(:,1:n) \in \mathbb{R}^{m\times n} \\ S_n &= S(1:n,1:n) \in \mathbb{R}^{n\times n} \end{aligned}$$

  Si $m \gg n$, esta descomposición se obtiene mucho más rapidamente y utiliza menos memoria.

- SVD *compacta*:

$$\begin{aligned} A &= U_r S_r V_r^{\mathrm{T}} \\ U_r &= U(:,1:r) \in \mathbb{R}^{m\times r} \\ S_r &= S(1:r,1:r) \in \mathbb{R}^{r\times r} \\ V_r &= V(:,1:r) \in \mathbb{R}^{n\times r} \end{aligned}$$

  con $r = \text{rango}(A)$.

- SVD *truncada*:

$$\begin{aligned} A_t &= U_t S_t V_t^{\mathrm{T}} \\ U_t &= U(:,1:r) \in \mathbb{R}^{m\times r} \\ S_t &= S(1:r,1:r) \in \mathbb{R}^{r\times r} \\ V_t &= V(:,1:r) \in \mathbb{R}^{n\times r} \end{aligned}$$

con $t \leq r$. Esta matriz A_t es la solución del problema da aproximación matricial de rango bajo (ver adelante).

1.2 Algo de historia

La SVD fue desarrollada independientemente por dos geómetras diferenciales, Eugenio Beltrami 1873 y Camille Jordan 1874. De manera independiente James Joseph Sylvester obtuvo la SVD para matrices cuadradas reales en 1889 (multiplicadores canónicos). Autone, también de manera independiente, llegó a la SVD, via descomposición polar.

Los principales nombres en el cálculo de la SVD son Kogbetliantz, Hestenes, Gene Golub, William Kahan, Christian Reinsch.

2 Resultados varios

1. Si se conoce la SVD de A, se tiene inmediatamente la SVD de A^{T}:
$$A^{\mathrm{T}} = VS^{\mathrm{T}}U^{\mathrm{T}} .$$

2. $$A\,V_{\cdot j} = \begin{cases} \sigma_j U_{\cdot j}\,, & j = 1, ..., p \\ 0 & j = p+1, ..., n \end{cases}$$
Las columnas $V_{\cdot j}\,,\ j = 1, ..., p$ se llaman los vectores singulares por derecha de A.

3. $$(U_{\cdot j})^{\mathrm{T}} A = \begin{cases} \sigma_j (V_{\cdot j})^{\mathrm{T}}\,, & j = 1, ..., p \\ 0 & j = p+1, ..., m \end{cases}$$
Las columnas $U_{\cdot j}$ se llaman los vectores singulares por izquierda de A.

4. $$\lambda_i(AA^{\mathrm{T}}) = \lambda_i(A^{\mathrm{T}}A), \quad i = 1, ..., p.$$
Para una matriz X con todos sus valores propios reales, éstos se pueden denotar en orden decreciente: $\lambda_1 = \lambda_1(X) \geq \lambda_2 = \lambda_2(X) \geq \cdots$.

5. $\sigma_i(A) = \sqrt{\lambda_i(A^{\mathrm{T}}A)}\,, \quad i = 1, ..., p.$

6. Sea r el número de valores singulares positivos.

$$r = \mathrm{rango}(S) = \mathrm{rango}(A).$$

7. $$\mathcal{N}(A) = \mathrm{gen}\{V_{\cdot r+1},\, ...,\, V_{\cdot n}\} \subseteq \mathbb{R}^{n\times 1}$$

8. $$\mathcal{R}(A) = \mathrm{gen}\{U_{\cdot 1},\, ...,\, U_{\cdot r}\} \subseteq \mathbb{R}^{m\times 1}$$

9. Sea E el hiperelipsoide $E = \{Ax : ||x||_2 = 1\}$. Los valores singulares corresponden a las longitudes de los semiejes de E.

10. Expansión SVD de A.

$$A = \sum_{i=1}^{r} \sigma_i U_{\cdot i}\, V_{\cdot i}{}^{\mathrm{T}}$$

11. $$\begin{aligned} |||A|||_2 = \sigma_1 = \sqrt{\rho(A^{\mathrm{T}}A)} &= \sqrt{\lambda_1(A^{\mathrm{T}}A)} \\ &\text{norma generada por la norma euclidiana,} \\ &\text{también llamada norma espectral.} \\ ||A||_F = (\sigma_1^2 + ... + \sigma_r^2)^{1/2} &\quad \text{norma de Frobenius.} \end{aligned}$$

12. Cuando $m \geq n$

$$\min_{x\neq 0} \frac{||Ax||_2}{||x||_2} = \sigma_n$$

13. Normas de Schatten o Schatten-Von Neumann

$$\begin{aligned} ||A||_q^S &= ||\sigma||_q = \left(\sum_{i=1}^{p} \sigma_i^q\right)^{1/q} \\ ||A||_1^S &= \sigma_1 + \cdots + \sigma_r \quad \text{norma nuclear o de Ky-Fan} \\ &= \mathrm{tr}\left(\sqrt{A^{\mathrm{T}}A}\right) \quad \text{también llamada norma de la traza} \\ ||A||_2^S &= ||A||_F \\ ||A||_\infty^S &= \sigma_1 = |||A|||_2 \end{aligned}$$

Las normas de Schatten son realmente normas y son (sub)multiplicativas $(||AB|| \leq ||A)|| \, ||B)||)$.

Si M es simétrica y semidefinida positiva, existe R simétrica tal que $RR = M$; se llama la *raíz cuadrada* de M y se puede denotar $\sqrt{M}$

14. Normas de Ky-Fan de orden q:

$$\begin{aligned}
||A||_q^{KF} &= \sum_{i=1}^{q} \sigma_i \,, \\
||A||_1^{KF} &= \sigma_1 = ||A||_\infty^S = |||A|||_2 \,, \\
||A||_r^{KF} &= ||A||_1^S
\end{aligned}$$

15. La matriz A_t definida en la descomposición truncada es la matriz que mejor aproxima A entre las matrices $m \times n$ de rango no superior a t utilizando la norma espectral o la norma de Frobenius:

$$A_t = \underset{\substack{X \in \mathbb{R}^{m\times n} \\ \text{rango}(X) \leq t}}{\text{argmin}} ||A - X||$$

16. Sea $A \in \mathbb{R}^{m\times n}$ y $A^+ \in \mathbb{R}^{n\times m}$ la inversa generalizada de Moore-Penrose,

$$\begin{aligned}
&AA^+ \text{ es simétrica,} \\
&A^+A \text{ es simétrica,} \\
&AA^+A = A \\
&A^+AA^+ = A^+
\end{aligned}$$

Entonces
$$A^+ = VS^+U^{\mathrm{T}} \,.$$

La obtención de S^+ es muy sencilla, los elementos diagonales de S^+ son de la forma
$$\left(S^+\right)_{ii} = \frac{1}{\sigma_i} \quad \text{si} \quad \sigma_i \neq 0.$$

17. $\sigma(\alpha A) = |\alpha| \, \sigma(A)$

18. $\sigma_1(A + B) \leq \sigma_1(A) + \sigma_1(B)$

19. $\sigma_i(A+B) \leq \sigma_i(A) + \sigma_i(B)$????

20. Si P y Q son ortogonales,

$$\sigma(A) = \sigma(PA)\,,$$
$$\sigma(A) = \sigma(AQ)\,.$$

En Matlab y en Scilab, la función y sintaxis para la obtención de la SVD son iguales:

`s = svd(A)` produce un vector columna con los valores singulares.

`[U, S, V] = svd(A)` produce la descomposición en valores singulares.

3 Aplicaciones

- Cálculo del rango
- Cálculo de la seudoinversa
- Procesamiento de señales
- Estadística
- Mínimos cuadrados
- Obtención del rango (recorrido de la transformación).
- Obtención del núcleo.
- Aproximación de matrices.

3.1 Ejemplos de tiempos

En las dos tablas siguientes están los tiempos aproximados de diferentes procesos numéricos para matrices 1000×1000 y 2000×2000 usando Scilab 5.2, en Kubuntu 10.04, con un procesador Intel Core 2 Duo T5250. En cada caso se corrió el proceso 5 veces.

$n = 1000$

producto	2.4	2.4	2.4	2.4	2.4
inversa	2.3	2.2	2.2	2.2	2.2
solución $Ax = b$	0.8	0.8	0.8	0.8	0.8
matriz escalonada red.	22.6	22.5	22.5	22.5	22.5
determinante	0.7	0.7	0.7	0.7	0.7
seudoinversa	22.7	22.9	22.8	22.7	22.8
valores singulares	3.3	3.4	3.3	3.3	3.3
SVD	21.0	21.3	21.2	21.0	21.2
rango	3.4	3.4	3.3	3.3	3.4
valores propios complejos	11.0	11.2	11.0	11.1	11.1
valores propios reales	1.5	1.5	1.5	1.5	1.5
vectores propios reales	10.7	10.7	10.7	10.7	10.7

$n = 2000$

producto	19.7	21.5	40.8	21.5	45.2
inversa	17.9	19.8	42.0	19.9	18.8
solución $Ax = b$	5.3	6.1	6.5	6.1	5.0
matriz escalonada red.	178.1	184.4	198.9	185.0	183.2
determinante	5.2	6.2	9.6	6.1	6.2
seudoinversa	192.1	205.6	259.9	239.2	244.9
valores singulares	27.1	28.5	28.3	28.2	42.7
SVD	180.2	187.7	245.3	209.3	186.1
rango	27.8	28.6	41.1	28.1	28.3
valores propios complejos	79.0	80.8	99.9	80.5	98.6
valores propios reales	12.5	12.9	12.5	12.8	12.6
vectores propios reales	86.3	88.1	86.5	112.2	86.6

En un curso usual de Álgebra Lineal, una de las maneras enseñadas para obtener el rango de una matriz consiste en determinar el número de filas no nulas de la matriz escalonada reducida. Para matrices pequeñas lo anterior funciona bien. Para matrices grandes, el método más rápido es por medio de la obtención de los valores singulares.

4 Cálculo de la SVD

4.1 Cálculo mediante la descomposición espectral

Los valores singulares de A se pueden obtener suponiendo que se conocen $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$ los valores propios de $A^{\mathrm{T}}A$,

$$\sigma_i = \sqrt{\lambda_i}\,.$$

Supongamos ahora que se conoce la descomposición espectral de $A^{\mathrm{T}}A$, es decir, se conoce una matriz diagonal Λ y una matriz ortogonal $W \in \mathbb{R}^{n\times n}$ tales que

$$A^{\mathrm{T}}A = W\Lambda W^{\mathrm{T}}$$

En el caso trivial, $A = 0$, entonces $U = I_m$, $V = I_n$, $S = 0 \in \mathbb{R}^{m\times n}$. Para los demás casos, sea r el rango de A, o sea, el número de valores singulares positivos. Inicialmente se construye una matriz $U' \in \mathbb{R}^{m\times r}$, de la siguiente manera:

$$U'(:,j) = \frac{1}{\sigma_j} A\,W(:,j)\,, \quad j = 1, ..., r. \tag{2}$$

Las columnas de U' son linealmente independientes. Se puede construir $U'' \in \mathbb{R}^{m\times m}$ agregando columnas a U' de tal forma que las m columnas de U'' sean linealmente independientes (resultado clásico de Álgebra Lineal). En seguida, a las columnas de U'' se les aplica el proceso de ortogonalización de Gram-Schmidt para obtener la matriz U. La matriz V será simplemente W.

Ejemplo 3.

```
A:
    -1      2      1
     3      4      7
     5      6     11
     8      9     17

W:
     0.373876      0.725868      0.577350
     0.441682     -0.686719      0.577350
```

```
     0.815558     0.039148    -0.577350

val. propios:      691.579685     4.420315     0

val. singulares:    26.297903     2.102454     0

S:
    26.297903     0            0
     0            2.102454     0
     0            0            0
     0            0            0

rango = 2

U inicial:
     0.050386    -0.979883
     0.326918    -0.140426
     0.512992    -0.028704
     0.792102     0.138877

U completada:
     0.050386    -0.979883     1     0
     0.326918    -0.140426     0     1
     0.512992    -0.028704     0     0
     0.792102     0.138877     0     0

U ortonormalizada:
     0.050386    -0.979883     0.193110     0
     0.326918    -0.140426    -0.797849     0.486664
     0.512992    -0.028704    -0.279501    -0.811107
     0.792102     0.138877     0.498020     0.324443

V = W:
     0.373876     0.725868     0.577350
     0.441682    -0.686719     0.577350
     0.815558     0.039148    -0.577350
```

4.2 Matrices de Householder

Sea $v \in \mathbb{R}^{q\times 1}$, $v \neq 0$, $u = v/||v||$ (vector columna de norma 1). Una matriz de Householder es una matriz de la forma

$$H = H_v = H(v) = I_q - \frac{2}{v^{\mathrm{T}} v} v\, v^{\mathrm{T}} = I_q - 2uu^{\mathrm{T}}.$$

A veces, al mismo tiempo que se obtiene el vector v deseado, se calcula el número

$$\beta = \frac{2}{v^{\mathrm{T}} v}\ ,$$

entonces es común expresar H en función de v y de β, aunque β no es ncesario. Simplemente, desde el punto de vista de eficiencia, si se conoce β no es interesante volverlo a calcular (son $2n-1$ "flops").

$$H = H(v, \beta) = I_n - \beta v\, v^{\mathrm{T}}\,.$$

La matriz H tiene dos características importantes, es *simétrica y ortogonal.*

Además, si $x \in \mathbb{R}^{q\times 1}$ se puede escoger v para que

$$H_v x \in < e^1 > .$$

es decir,

$$H_v x = \begin{bmatrix} \times \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

La deducción de cómo escoger v es sencilla. A continuación está un algoritmo eficiente y que busca disminuir los errores de redondeo [GoV96].

```
[v, β] = vHouse(x)
    n = dim (x)
    t = x(2 : n)^T x(2 : n)
    v = [ 1 ; x(2 : n) ]
    si t = 0
        β = 0
    sino
        ν = sqrt(x_1^2 + t)
        si x_1 ≤ 0
            v_1 = x_1 − ν
        sino
            v_1 = −t/(x_1 + ν)
        fin-si
        β = 2v_1^2/(t + v_1^2)
        v = v/v_1
    fin-si
fin vHouse
```

Ejemplo 4.

$$x = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}, \quad \beta = 0.6286093 \ , \quad v = \begin{bmatrix} 1 \\ -0.8862198 \\ -1.1816264 \end{bmatrix}$$

$$H = \begin{bmatrix} 0.3713907 & 0.5570860 & 0.7427814 \\ 0.5570860 & 0.5062994 & -0.6582675 \\ 0.7427814 & -0.6582675 & 0.1223100 \end{bmatrix}, \quad Hx = \begin{bmatrix} 5.3851648 \\ 0 \\ 0 \end{bmatrix}$$

En la mayoría de las aplicaciones no es necesario construir explícitamente la matriz H. Lo realmente importante es poder efectuar de **manera eficiente** el producto HA o AH. Para esto basta con conocer β y v.

Algunas veces es necesario utilizar matrices de Householder definidas por bloques. Si H es un matriz de Householder, también lo es una matriz de la forma

$$\begin{bmatrix} I_s & 0 \\ 0 & H \end{bmatrix}$$

4.3 Matrices de Givens

Esta es otra clase de matrices ortogonales. Sea θ un ángulo y

$$\begin{aligned} c &= \cos(\theta) \\ s &= \operatorname{sen}(\theta), \end{aligned}$$

La matriz de Givens, en $\mathbb{R}^{q\times q}$, es simplemente un rotación definida en el plano de las variables i y k:

$$G = G(i,k,c,s,q) = \begin{array}{cc} \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & & \ddots & & & & & \\ 0 & 0 & \cdots & c & & s & & 0 \\ \vdots & & & & \ddots & & & \\ 0 & 0 & \cdots & -s & & c & & 0 \\ \vdots & & & & & & \ddots & \\ 0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} & \begin{array}{c} \\ \\ \\ i \\ \\ k \\ \\ \\ \end{array} \end{array}$$

El producto $y = G^{\mathrm{T}}x$ se calcula muy fácilmente:

$$y_j = \begin{cases} cx_i - sx_k & \text{si} \quad j = i, \\ sx_i + cx_k & \text{si} \quad j = k, \\ x_j & \text{en los demás casos.} \end{cases}$$

Si se desea que $y_k = 0$, basta con tomar

$$\begin{aligned} c &= \frac{x_i}{\sqrt{x_i^2 + x_k^2}}\ , \\ s &= \frac{-x_k}{\sqrt{x_i^2 + x_k^2}}\ . \end{aligned}$$

En la práctica, es mejor utilizar la siguiente versión para el cálculo de c y s (ver [GoVa96]),

$[c,\ s] = \texttt{csGivens}(a, b)$
si $b = 0$
$c = 1$
$s = 0$
sino
si $|b| > |a|$
$t = -a/b$
$s = 1/\sqrt{1+t^2}$
$c = st$
sino
$t = -b/a$
$c = 1/\sqrt{1+t^2}$
$s = ct$
fin-si
fin-si
fin `csGivens`

Por medio de esta función

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}.$$

Ejemplo 5.

Para el vector

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 2 \\ -3 \end{bmatrix} \quad \text{por medio de la función se obtiene} \quad \begin{array}{l} c = 0.5547002 \\ s = 0.8320503 \end{array}$$

y así

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} 2 \\ -3 \end{bmatrix} = \begin{bmatrix} 3.6055513 \\ 0 \end{bmatrix}.$$

4.4 Bidiagonalización de una matriz

Sea B una matriz $m \times n$ con $m \geq n$. Se dice que B es *bidiagonal* (superior) si todos los elementos por fuera de la diagonal y de la superdiagonal son nulos.

El proceso de bidiagonalización utiliza matrices de Householder. Primero se buscan ceros, por debajo de la diagonal, en la columna 1. Después se buscan ceros, a la derecha de la superdiagonal en la fila 1. Después ceros en la columna 2. Después ceros en la fila 2. Después ceros en la columna 3. ...

$$
\begin{bmatrix}
\times & \times & \times & \times \\
\times & \times & \times & \times \\
\times & \times & \times & \times \\
\times & \times & \times & \times \\
\times & \times & \times & \times
\end{bmatrix}
\begin{bmatrix}
\times & \times & \times & \times \\
 & \times & \times & \times \\
 & \times & \times & \times \\
 & \times & \times & \times \\
 & \times & \times & \times
\end{bmatrix}
\begin{bmatrix}
\times & \times & & \\
 & \times & \times & \times \\
 & \times & \times & \times \\
 & \times & \times & \times \\
 & \times & \times & \times
\end{bmatrix}
\begin{bmatrix}
\times & \times & & \\
 & \times & \times & \times \\
 & & \times & \times \\
 & & \times & \times \\
 & & \times & \times
\end{bmatrix}
$$

$$
\begin{bmatrix}
\times & \times & & \\
 & \times & \times & \\
 & & \times & \times \\
 & & \times & \times \\
 & & \times & \times
\end{bmatrix}
\begin{bmatrix}
\times & \times & & \\
 & \times & \times & \\
 & & \times & \times \\
 & & & \times \\
 & & & \times
\end{bmatrix}
\begin{bmatrix}
\times & \times & & \\
 & \times & \times & \\
 & & \times & \times \\
 & & & \times \\
 & & &
\end{bmatrix}
$$

datos: $A \in \mathbb{R}^{m\times n}, \quad m \geq n$
$B \leftarrow A$
para $k = 1 : n$
    si $k \leq m - 1$
        $[\beta, v] =$ `vHouse`$(\, B(k : m, k)\,)$
        $B(k : m, k : n) \leftarrow H(\beta, v)\, B(k : m, k : n)$
    fin-si
    si $k \leq n - 2$
        $[\beta, v] =$ `vHouse`$(\, B(k, k+1 : n)^{\mathrm{T}}\,)$
        $B(k : m, k+1 : n) \leftarrow B(k : m, k+1 : n)\, H(\beta, v)$
    fin-si
fin-para

Si se desea obtener las matrices P y Q ortogonales tales que $B = PAQ$, entonces es necesario ir almacenando las operaciones que se efectúan sobre B. Así, inicialmente $P = I_m$. Las multiplicaciones por la izquierda que se hacen sobre B, también deben hacerse sobre P. Por ejemplo

$$P(k : m, k : n) \leftarrow H(\beta, v)\, P(k : m, k : n)$$

De manera análoga, en Q se almacenan las multiplicaciones por derecha.

Ejemplo 6.

```
    3          2          0          4
   -1          0          5          4
    2          5          3          5
   -1          4          1          3
    4         -1          4          5
columna  1
beta = 0.461184
v^T:     1          0.389444   -0.778888    0.389444   -1.557775
H izq
    0.538816   -0.179605    0.359211   -0.179605    0.718421
   -0.179605    0.930054    0.139892   -0.069946    0.279785
    0.359211    0.139892    0.720215    0.139892   -0.559569
   -0.179605   -0.069946    0.139892    0.930054    0.279785
    0.718421    0.279785   -0.559569    0.279785   -0.119139
B
    5.567764    1.436842    2.873685    6.286186
    0          -0.219318    6.119139    4.890341
    0           5.438636    0.761722    3.219318
    0           3.780682    2.119139    3.890341
    0          -0.122727   -0.476555    1.438636
fila  1
beta = 0.796471
v^T:     1         -0.511076   -1.117979
H der
    0.203529    0.407057    0.890438
    0.407057    0.791963   -0.455082
    0.890438   -0.455082    0.004509
B
    5.567764    7.059654    0           0
    0           6.800750    2.531349   -2.957949
    0           4.283588    1.352040    4.510639
    0           5.096197    1.446810    2.419622
    0           1.062053   -1.082068    0.114077
columna  2
beta = 0.289808
v^T:     1         -1.543533   -1.836346   -0.382697
H izq
    0.710192    0.447328    0.532188    0.110909
```

```
    0.447328    0.309534   -0.821450   -0.171191
    0.532188   -0.821450    0.022719   -0.203666
    0.110909   -0.171191   -0.203666    0.957556
B
    5.567764    7.059654    0           0
    0           9.575934    3.052514    1.217372
    0           0           0.547605   -1.934107
    0           0           0.489771   -5.247711
    0           0          -1.281516   -1.483804
fila  2
beta = 0.071143
v^T:     1          -5.206975
H der
    0.928857    0.370437
    0.370437   -0.928857
B
    5.567764    7.059654    0           0
    0           9.575934    3.286310    0
    0           0          -0.207818    1.999363
    0           0          -1.489020    5.055805
    0           0          -1.740002    0.903521
columna  3
beta = 1.090373
v^T:     1           0.593855    0.693953
beta = 1.090373
v^T:     1           0.593855    0.693953
H izq
   -0.090373   -0.647524   -0.756667
   -0.647524    0.615465   -0.449351
   -0.756667   -0.449351    0.474909
B
    5.567764    7.059654    0           0
    0           9.575934    3.286310    0
    0           0           2.299560   -4.138108
    0           0           0           1.411035
    0           0           0          -3.355594
columna  4
beta = 0.612374
```

```
v^T:     1          1.505317
H izq
    0.387626   -0.921817
   -0.921817   -0.387626
B
    5.567764    7.059654    0           0
    0           9.575934    3.286310    0
    0           0           2.299560   -4.138108
    0           0           0           3.640197
    0           0           0           0
```

4.5 Cálculo directo

Se puede suponer que $A \in \mathbb{R}^{m\times n}$ con $m \geq n$. Si no es así, se calcula la SVD de A^{T}.

El cálculo se puede dividir en dos partes. En la primera parte, un proceso finito, se bidiagonaliza A mediante matrices ortogonales de Householder. En la segunda parte, proceso iterativo que converge rápidamente, se aplica el algoritmo de Golub-Kahan para ir anulando paso a paso los elementos superdiagonales. También es necesario tratar el caso de una matriz bidiagonal con superdiagonal sin ceros y por lo menos un elemento diagonal nulo. Se utilizan matrices ortogonales de Givens

4.6 Algoritmo de Golub-Kahan

Dada una matriz $B \in \mathbb{R}^{n\times n}$, *bidiagonal estricta* (no tiene ceros en la diagonal ni en la superdiagonal), este proceso iterativo utiliza matrices de Givens buscando anular un elemento superdiagonal, el de la posición $(n-1, n)$.

Dado $\varepsilon > 0$ pequeño, se considera que B bidiagonal, es estricta, si

$$|b_{ii}| \geq \varepsilon ||B||, \quad i = 1, ..., n,$$
$$|b_{i,i+1}| \geq \varepsilon \left(|b_{ii}| + |b_{i+1,i+1}|\right), \quad i = 1, ..., n-1.$$

datos: $B_0 \in \mathbb{R}^{n\times n}, \quad \varepsilon$
$B \leftarrow B_0$
mientras B es bidiagonal estricta
    $T \leftarrow B^{\mathrm{T}}B$
    obtener los valores propios de $T(n-1:n, n-1:n)$
    $\mu \leftarrow$ valor propio más cercano a t_{nn}
    $y \leftarrow t_{11} - \mu$
    $z \leftarrow t_{12}$
    para $k = 1 : n-1$
        $[c, s] \leftarrow$ `csGivens`(y, z)
        $B \leftarrow B\, G(k, k+1, c, s)$
        $y \leftarrow b_{kk}$
        $z \leftarrow b_{k+1,k}$
        $[c, s] \leftarrow$ `csGivens`(y, z)
        $B \leftarrow G(k, k+1, c, s)^{\mathrm{T}}\, B$
        si $k \leq n-2$
            $y \leftarrow b_{k,k+1}$
            $z \leftarrow b_{k,k+2}$
        fin-si
    fin-para
fin-mientras

Este algoritmo está basado en el algoritmo QR simétrico para la matriz $B^{\mathrm{T}}B$ pero sin utilizar explícitamente $B^{\mathrm{T}}B$.

La presentación anterior del algoritmo es clara y precisa pero hay operaciones ineficientes o que nunca se hacen explícitamente. Por ejemplo, no se debe construir explícitamente T. Tampoco se debe construir explícitamente G para hacer el producto BG o $G^{\mathrm{T}}B$. Lo indispensable es, por ejemplo, poder actualizar B como el resultado de BG.

Mejor aún, B se puede representar por medio de dos vectores, d con los elementos de la diagonal y f con los elementos de la superdiagonal. Así todo el algoritmo se puede efectuar simplemente con d y f.

$$B = \begin{bmatrix} d_1 & f_1 & 0 & 0 & 0 & 0 \\ 0 & d_2 & f_2 & 0 & 0 & 0 \\ 0 & 0 & d_3 & f_3 & 0 & 0 \\ & & & & & \\ & & & & & \\ 0 & 0 & 0 & 0 & d_{n-1} & f_{n-1} \\ 0 & 0 & 0 & 0 & 0 & d_n \end{bmatrix}$$

Ejemplo 7.

```
B inicial:
    2          -3           0           0
    0           4          -5           0
    0           0          -6          -7
    0           0           0           8
iter.  1  ------------------------------
submatriz inferior derecha de T
   61          42
   42         113
valores propios:
   37.603644  136.396356
mu =   136.396356   y =  -132.396356   z =    -6.000000
k =  1 ...................
c =     0.998975    s =    -0.045272
B:
    1.862133   -3.087468    0           0
    0.181088    3.995899   -5           0
    0           0          -6          -7
    0           0           0           8
y =     1.862133    z =     0.181088
c =     0.995305    s =    -0.096791
B final k:
    1.870918   -2.686205   -0.483955    0
    0           4.276976   -4.976524    0
    0           0          -6          -7
    0           0           0           8
y =    -2.686205    z =    -0.483955
```

```
k =  2 ....................
c =     0.984155    s =    -0.177308
B:
    1.870918   -2.729452    0           0
    0           3.325845   -5.655839    0
    0          -1.063851   -5.904932   -7
    0           0           0           8
y =     3.325845    z =    -1.063851
c =     0.952459    s =     0.304667
B final k:
    1.870918   -2.729452    0           0
    0           3.491851   -3.587918    2.132668
    0           0          -7.347352   -6.667213
    0           0           0           8
y =    -3.587918    z =     2.132668
k =  3 ....................
c =     0.859608    s =     0.510953
B:
    1.870918   -2.729452    0           0
    0           3.491851   -4.173898    0
    0           0          -2.909210   -9.485347
    0           0          -4.087627    6.876867
y =    -2.909210    z =    -4.087627
c =    -0.579849    s =     0.814724
B final k:
    1.870918   -2.729452    0           0
    0           3.491851   -4.173898    0
    0           0           5.017191   -0.102687
    0           0           0         -11.715484

...

iter.  2 ------------------------------
B final de iteracion:
    1.770452   -2.547800    0           0
    0           3.209002   -3.539764    0
    0           0           5.768936    0.000000
    0           0           0         -11.716056
```

```
iter.  3  ------------------------------
B final de iteracion:
    1.687270   -2.406186    0           0
    0           3.068430   -2.757695    0
    0           0           6.330662    0
    0           0           0         -11.716056

B final:
    1.687270   -2.406186    0           0
    0           3.068430   -2.757695    0
    0           0           6.330662    0
    0           0           0         -11.716056
```

4.7 Tratamiento de un elemento diagonal nulo

Sea $B \in \mathbb{R}^{n\times n}$ una matriz bidiagonal con todos sus elementos superdiagonales diferentes de cero y sea i el mayor índice tal que $b_{ii} = 0$. Es necesario considerar dos casos: $i < n$ e $i = n$.

Supongamos que $i < n$. El objetivo es anular, paso a paso, toda la fila i por medio de matrices de Givens. Inicialmente $b_{i,i+1} \neq 0$. Se busca anular $b_{i,i+1}$ pero aparece un elemento no nulo, $b_{i,i+2}$. Entonces se buscar anular $b_{i,i+2}$ pero aparece $b_{i,i+3} \neq 0$. Y así sucesivamente.

Ejemplo 8. Para $n = 5$ e $i = 2$.

$$B = \begin{bmatrix} \times & \times & & & \\ & & \bar{\times} & & \\ & & \times & \times & \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Para anular b_{23}:

$$\begin{array}{l} [c, s] = \texttt{csGivens}(b_{33}, b_{23}) \\ B \leftarrow G(c, s, 3, 2)^{\mathrm{T}} B \end{array}$$

Entonces,

$$B = \begin{bmatrix} \times & \times & & & \\ & & 0 & \bar{\times} & \\ & & \times & \times & \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Para anular b_{24}:

$$\begin{aligned} &[c, s] = \texttt{csGivens}(b_{44}, b_{24}) \\ &B \leftarrow G(c, s, 4, 2)^{\mathrm{T}} B \end{aligned}$$

Entonces,

$$B = \begin{bmatrix} \times & \times & & & \\ & & & 0 & \bar{\times} \\ & & \times & \times & \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Para anular b_{25}:

$$\begin{aligned} &[c, s] = \texttt{csGivens}(b_{55}, b_{25}) \\ &B \leftarrow G(c, s, 5, 2)^{\mathrm{T}} B \end{aligned}$$

Entonces,

$$B = \begin{bmatrix} \times & \times & & & \\ & & & & 0 \\ & & \times & \times & \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Así acaba el proceso de tratamiento de elemento diagonal nulo.

Consideremos ahora el caso $i = n$. El objetivo es anular, paso a paso, toda la columna n por medio de matrices de Givens. Inicialmente $b_{n-1,n} \neq 0$. Se busca anular $b_{n-1,n}$ pero aparece un elemento no nulo, $b_{n-2,n}$. Entonces se buscar anular $b_{n-2,n}$ pero aparece $b_{n-3,n} \neq 0$. Y así sucesivamente.

Ejemplo 9. Para $i = n = 5$:

$$B = \begin{bmatrix} \times & \times & & & \\ & \times & \times & & \\ & & \times & \times & \\ & & & \times & \bar{\times} \\ & & & & 0 \end{bmatrix}$$

Para anular b_{45}:

$$\begin{array}{l} [c, s] = \texttt{csGivens}(b_{44}, b_{45}) \\ B \leftarrow B\, G(c, s, 4, 5) \end{array}$$

$$B = \begin{bmatrix} \times & \times & & & \\ & \times & \times & & \\ & & \times & \times & \bar{\times} \\ & & & \times & 0 \\ & & & & 0 \end{bmatrix}$$

Para anular b_{35}:

$$\begin{array}{l} [c, s] = \texttt{csGivens}(b_{33}, b_{35}) \\ \text{y } B \leftarrow B\, G(c, s, 3, 5) \end{array}$$

$$B = \begin{bmatrix} \times & \times & & & \\ & \times & \times & & \bar{\times} \\ y & & \times & \times & 0 \\ & & & \times & 0 \\ & & & & 0 \end{bmatrix}$$

Para anular b_{25}:

$$\begin{array}{l} [c, s] = \texttt{csGivens}(b_{22}, b_{25}) \\ B \leftarrow B\, G(c, s, 2, 5) \end{array}$$

$$B = \begin{bmatrix} \times & \times & & & \bar{\times} \\ & \times & \times & & 0 \\ & & \times & \times & 0 \\ & & & \times & 0 \\ & & & & 0 \end{bmatrix}$$

Para anular b_{15}:

$$\begin{aligned} &[c, s] = \texttt{csGivens}(b_{11}, b_{15}) \\ &B \leftarrow B\, G(c, s, 1, 5) \end{aligned}$$

$$B = \begin{bmatrix} \times & \times & & & 0 \\ & \times & \times & & 0 \\ & & \times & \times & 0 \\ & & & \times & 0 \\ & & & & 0 \end{bmatrix}$$

Así acaba el proceso de tratamiento del elemento diagonal nulo b_{nn}.

datos: $B_0, \quad i$
$B \leftarrow B_0$
si $i < n$
    para $k = i + 1 : n$
        $[c, s] = \texttt{csGivens}(b_{kk}, b_{ik})$
        $B \leftarrow G(c, s, k, i)^{\mathrm{T}}\, B$
        $b_{ik} \leftarrow 0$
    fin-para
sino
    para $k = n - 1 : -1 : 1$
        $[c, s] = \texttt{csGivens}(b_{kk}, b_{kn})$
        $B \leftarrow B\, G(c, s, k, n)$
        $b_{kn} \leftarrow 0$
    fin-para
fin-si

También este proceso se puede llevar a cabo con toda la información de B en dos vectores d y f.

Ejemplo 10.

```
B
    7           5           0           0           0
    0           0           8           0           0
    0           0           2           3           0
    0           0           0           6           1
    0           0           0           0           9
i = 2

k = 3
Datos para csGivens: B(3,3),  B(2,3) :  2.000000   8.000000
c, s:   -0.242536   0.970143
G(c,s,3,2)T  B
B ..
    7           5           0           0           0
    0           0           0           2.910428    0
    0           0          -8.246211   -0.727607    0
    0           0           0           6           1
    0           0           0           0           9

k = 4
Datos para csGivens: B(4,4),  B(2,4) :  6.000000   2.910428
c, s:   0.899735   -0.436436
G(c,s,4,2)T  B
B ..
    7           5           0           0           0
    0           0           0           0          -0.436436
    0           0          -8.246211   -0.727607    0
    0           0           0           6.668627    0.899735
    0           0           0           0           9

k = 5
Datos para csGivens: B(5,5),  B(2,5) :  9.000000   -0.436436
c, s:   0.998826   0.048436
```

```
G(c,s,5,2)T  B
B ..
    7          5          0          0          0
    0          0          0          0          0
    0          0          -8.246211  -0.727607  0
    0          0          0          6.668627   0.899735
    0          0          0          0          9.010576
```

Ejemplo 11.

```
    7          8          0          0          0
    0          1          4          0          0
    0          0          2          5          0
    0          0          0          6          9
    0          0          0          0          0
i = 5


k = 4
Datos para csGivens: B(4,4),  B(4,5) :  6.000000   9.000000
c, s:   -0.554700   0.832050
B G(c,s,4,5)
B ..
    7          8          0          0          0
    0          1          4          0          0
    0          0          2          -2.773501  4.160251
    0          0          0         -10.816654  0
    0          0          0          0          0

k = 3
Datos para csGivens: B(3,3),  B(3,5) :  2.000000   4.160251
c, s:   -0.433273   0.901263
B G(c,s,3,5)
B ..
    7          8          0          0          0
    0          1          -1.733093  0          3.605051
    0          0          -4.616026  -2.773501  0
```

```
    0          0          0         -10.816654    0
    0          0          0           0           0

k = 2
Datos para csGivens: B(2,2),  B(2,5) :  1.000000   3.605051
c, s:   -0.267296   0.963615
B G(c,s,2,5)
B ..
    7         -2.138366    0           0           7.708916
    0         -3.741175   -1.733093    0           0
    0          0          -4.616026   -2.773501    0
    0          0           0         -10.816654    0
    0          0           0           0           0

k = 1
Datos para csGivens: B(1,1),  B(1,5) :  7.000000   7.708916
c, s:   -0.672246   0.740327
B G(c,s,1,5)
B ..
  -10.412847   -2.138366    0           0           0
    0         -3.741175   -1.733093    0           0
    0          0          -4.616026   -2.773501    0
    0          0           0         -10.816654    0
    0          0           0           0           0
```

4.8 Algoritmo completo

datos: $A \in \mathbb{R}^{m\times n}$ con $m \geq n$, ε

$B \leftarrow$ bidiagonalización(A)
$B \leftarrow B(1:n, 1:n)$
`fin` $\leftarrow 0$
mientras `fin` $= 0$
    para $i = 1 : n-1$
        si $|b_{i,i+1}| \leq \varepsilon\,(|b_{ii}| + |b_{i+1,i+1}|)$ **ent** $b_{i,i+1} \leftarrow 0$
    fin-para
    obtener B^3, B^2, B^1 cuadradas tales que:

- $B = \begin{bmatrix} B^1 & & \\ & B^2 & \\ & & B^3 \end{bmatrix}$
- B^3 es diagonal
- la superdiagonal de B^2 no tiene ceros

    si $B^3 = B$
        `fin` $\leftarrow 1$
    sino
        si algún $b^2_{ii} = 0$ (en la matriz B^2)
            $i = \max\{i : b^2_{ii} = 0\}$
            tratamiento de elemento diagonal nulo para B^2
        sino
            aplicar el algoritmo de Golub-Kahan a B^2
        fin-si
    fin-si
fin-mientras
$b_{ii} \leftarrow |\,b_{ii}\,|$ para todo i
ordenar de manera decreciente los valores singulares

Ejemplo 12.

```
A
    2            3            4            5
    6            7            8            9
   10           11           12          -13
```

```
   14          15          16         -17
   18          19         -20         -21

 bidiagonalizacion:
   25.690465   38.352907    0           0
    0          11.191634   13.501245    0
    0           0          10.474484  -27.485191
    0           0           0           2.523431
    0           0           0           0

iteracion 1
B^1 : []
B^2 :
   25.690465   38.352907    0           0
    0          11.191634   13.501245    0
    0           0          10.474484  -27.485191
    0           0           0           2.523431
B^3 : []
FIN de Golub Kahan.
   43.104656  -19.165800    0           0
    0           3.576374   13.098407    0
    0           0          -1.645441    0
    0           0           0         -29.959881
Nueva matriz B(1:n,1:n) :
   43.104656  -19.165800    0           0
    0           3.576374   13.098407    0
    0           0          -1.645441    0
    0           0           0         -29.959881

iteracion 2
B^1 : []
B^2 :
   43.104656  -19.165800    0
    0           3.576374   13.098407
    0           0          -1.645441
B^3 :
  -29.959881
FIN de Golub Kahan.
```

```
   47.197868    0.013213    0
    0          -0.395548    0
    0           0          13.587131
Nueva matriz B(1:n,1:n) :
   47.197868    0.013213    0           0
    0          -0.395548    0           0
    0           0          13.587131    0
    0           0           0         -29.959881

iteracion 3
B^1 : []
B^2 :
   47.197868    0.013213
    0          -0.395548
B^3 :
   13.587131    0
    0         -29.959881
FIN de Golub Kahan.
   47.197870    0
    0          -0.395548
Nueva matriz B(1:n,1:n) :
   47.197870    0           0           0
    0          -0.395548    0           0
    0           0          13.587131    0
    0           0           0         -29.959881

iteracion 4
B^3 = B
```

Bibliografía

[BaD87] Bai Z. et al., *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM, Philadelphia, 1987.

[DoM79] Dongarra J. et al., *LINPACK Users' Guide* by J.J Dongarra SIAM, Philadelphia, 1979.

[GoV96] Golub G.H. y Van Loan C.H., *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, 1996.

[MaN99] Magnus J.R., Neudecker H., *Matrix Differential Calculus with Applications in Statistics and Econometrics*, revised ed., Wiley, Chichester, 1999.

[NoD88] Noble B. y Daniel J.W., *Applied Linear Algebra*, Prentice-Hall, Englewood Cliffs, 1988.

[Ste01] G.W. Stewart, *Matrix Algorithms, Vol. II: Eigensystems*, SIAM, Philadelphia, 2001.

[Wat02] Watkins D.S., *Fundamentals of Matrix Computations*, 2nd ed., Wiley, New York, 2002.

`http://en.wikipedia.org/wiki/Singular_value_decomposition`