Iteración de Arnoldi y GMRES

Azul Fatalini - Luis Ferroni [*]

Análisis Numérico II - FaMAF U.N.C.

1. Preliminares

1.1. Introducción

Al momento de resolver sistemas lineales $Ax = b$ con matrices cuadradas $n \times n$, suele ser de gran utilidad el poder escribir A de una manera *más simple*. Más aún, cuando el problema se trata de estimar los autovalores de una matriz de este tipo, muchos de los métodos conocidos para llevar tal tarea a cabo, consisten en trabajar con matrices *semejantes* a A (pues los autovalores son un invariante de las matrices bajo semejanza).

El problema de calcular los autovalores de una matriz $A \in \mathbb{C}^{n\times n}$, según puede verse, es equivalente a hallar las soluciones de un polinomio mónico de grado n. Sin embargo, el siguiente resultado, debido a Niels Henrik Abel, echa por tierra cualquier intención de poder resolver dicho problema en general:

Teorema 1.1 (Teorema de Abel) *En general, dado un polinomio p de grado $n \geq 5$, no es posible expresar sus raíces (reales o complejas) por medio de radicales.*

Una consecuencia inmediata del Teorema de Abel es que no es posible obtener las raíces de p mediante un número finito de sumas, multiplicaciones y potenciaciones racionales. Por lo tanto, el problema de hallar los autovalores de una matriz no puede resolverse algorítmicamente de manera exacta.

No obstante, el carácter iterativo de muchos de los métodos conocidos permite estimar de manera más que razonable el valor de los autovalores de una matriz. En este artículo se procurará explicar de manera detallada uno de los métodos más conocidos para llevar esta tarea a cabo pensado por Walter E. Arnoldi.

Además, utilizando este algoritmo, apoyándonos en algunos cálculos realizados de manera inteligente, podremos resolver el problema $Ax = b$ con matrices $A \in \mathbb{C}^{n\times n}$ generales en una cantidad razonable de iteraciones.

1.2. Semejanza vía matrices unitarias

Vamos a recopilar algunos resultados que luego nos motivarán a llevar a cabo los métodos que nos interesan.

Proposición 1.1 *Dos matrices semejantes A y B tienen los mismos autovalores contando multiplicidades.*

[*]Proyecto correspondiente a la materia *Análisis Numérico II* de la Lic. en Matemática de la Facultad de Matemática, Astronomía y Física de la Universidad Nacional de Córdoba. Noviembre de 2014.

Proposición 1.2 *Si $U \in \mathbb{C}^{n\times n}$ es una matriz triangular superior, entonces los elementos diagonales de U son sus autovalores contando multiplicidades.*

Teorema 1.2 (Teorema de Schür) *Sea $A \in \mathbb{C}^{n\times n}$ una matriz cualquiera, entonces existe una matriz unitaria $Q \in \mathbb{C}^{n\times n}$ tal que QAQ^* es una matriz triangular superior.*

Debido a estos tres resultados, dada una matriz A, si conseguimos hallar una matriz Q unitaria, tal que $QAQ^* = U$ sea triangular superior (dicha matriz existe, según lo asegura el teorema 1.2), entonces ciertamente bastará hallar los autovalores de U para obtener los de A. Pero aprovechando la estructura triangular superior de U, resultará inmediato hallar dichos autovalores, pues serán los elementos de la diagonal de U.

Pero entonces, debido al teorema 1.1, no será posible hallar tal matriz Q unitaria mediante un algoritmo finito que utilice sumas, productos y exponenciaciones racionales, debido a que esto violaría tal teorema.

Sin embargo, a pesar de que no podremos hallar exactamente las matrices Q unitarias tales que QAQ^* sea triangular superior, sí podremos hallar matrices $\overline{Q}$ tales que $\overline{Q}A\overline{Q}^*$ tenga una estructura muy similar a la de una matriz triangular superior. De esto se tratará la iteración de Arnoldi.

Definición 1.1 *Diremos que una matriz $H \in \mathbb{C}^{n\times n}$ tiene forma triangular superior de Hessenberg si:*

$$H = \begin{bmatrix} * & * & * & \cdots & * & * \\ * & * & * & \cdots & * & * \\ 0 & * & * & \cdots & * & * \\ 0 & 0 & * & \cdots & * & * \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & * & * \end{bmatrix}$$

donde las entradas marcadas con $*$ pueden ser no nulas.

Vamos a tratar de hallar matrices unitarias que lleven a una matriz A en una matriz triangular superior de Hessenberg mediante semejanza.

2. La Iteración de Arnoldi

2.1. Deducción de la Iteración de Arnoldi

Hemos motivado centralmente la iteración de Arnoldi debido a su evidente utilidad para la estimación de los autovalores de una matriz. Sin embargo, este método, según veremos, también será muy útil para la resolución de sistemas lineales. En lo que sigue de esta sección trataremos de deducir la iteración de Arnoldi, teniendo en mente la iteración de Gram-Schmidt (modificado).

Sea ahora A una matriz de tamaño $m \times m$. Supongamos que tenemos una matriz H triangular superior de Hessenberg, y una matriz unitaria Q tales que:

$$A = QHQ^*$$

Esto claramente es equivalente a $AQ = QH$.

Sea ahora $n < m$, y supongamos que las columnas de Q son $\mathbf{q}^1, \ldots, \mathbf{q}^m$. Entonces, ciertamente:

$$A[\mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^n, \mathbf{q}^{n+1}, \ldots, \mathbf{q}^m] = [\mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^n, \mathbf{q}^{n+1}, \ldots, \mathbf{q}^m] \times$$
$$\times \begin{bmatrix} h_{11} & h_{12} & & \cdots & & h_{1n} & \cdots & & h_{1m} \\ h_{21} & h_{22} & & \cdots & & h_{2n} & \cdots & & h_{2m} \\ 0 & h_{32} & h_{33} & & & h_{3n} & \cdots & & h_{3m} \\ & 0 & h_{43} & \ddots & & & & & \\ & & 0 & \ddots & h_{n-1,n-2} & \vdots & & & \vdots \\ \vdots & & & \ddots & h_{n,n-1} & h_{nn} & & & \\ & & & & 0 & h_{n+1,n} & & & \\ & & & & & 0 & \ddots & \ddots & \\ 0 & & & \cdots & & & 0 & h_{m,m-1} & h_{mm} \end{bmatrix}$$

Ahora, consideramos sólo una parte de estas multiplicaciones. Esto es, tomemos:

$$Q_n = [\mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^n]$$
$$Q_{n+1} = [\mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^n, \mathbf{q}^{n+1}]$$
$$\overline{H}_n = \begin{bmatrix} h_{11} & h_{12} & & \cdots & & h_{1n} \\ h_{21} & h_{22} & & \cdots & & h_{2n} \\ 0 & h_{32} & h_{33} & & & h_{3n} \\ & 0 & h_{43} & \ddots & & \\ & & 0 & \ddots & h_{n-1,n-2} & \vdots \\ \vdots & & & \ddots & h_{n,n-1} & h_{nn} \\ & & & & 0 & h_{n+1,n} \end{bmatrix}$$

La multiplicación de matrices por bloques revela que $AQ_n = Q_{n+1}\overline{H}_n$. Además, nótese que ambos lados de esta igualdad son matrices de tamaño $m \times n$. Comparando las n-ésimas columnas a ambos lados de la ecuación, se tiene:

$$A\mathbf{q}^n = h_{1n}\mathbf{q}^1 + h_{2n}\mathbf{q}^2 + \ldots + h_{nn}\mathbf{q}^n + h_{n+1,n}\mathbf{q}^{n+1}$$

la cual puede ser reescrita de la siguiente manera:

$$\mathbf{q}^{n+1} = \frac{A\mathbf{q}^n - \sum_{i=1}^{n} h_{in}\mathbf{q}^i}{h_{n+1,n}}.$$

Luego, hemos encontrado una manera recursiva de construir las columnas de la matriz unitaria Q, que es conocida como *la iteración de Arnoldi*.

Notemos que el primer paso de la iteración de Arnoldi se lleva adelante como sigue: se comienza con un vector $\mathbf{q}^1$ unitario. Generamos $\mathbf{q}^2$ con la iteración:

$$\mathbf{q}^2 = \frac{A\mathbf{q}^1 - h_{11}\mathbf{q}^1}{h_{21}}.$$

Debemos destacar que, hasta este punto, los h_{ij} son inmateriales. Sin embargo, si queremos que la matriz Q sea ortogonal, ciertamente debe ser $\langle \mathbf{q}^1, \mathbf{q}^2 \rangle = 0$, lo cual se traduce, en términos de matrices, a que $(\mathbf{q}^1)^* \mathbf{q}^2 = 0$. Por lo tanto, en términos de la iteración, debe ocurrir que:

$$(\mathbf{q}^1)^* A\mathbf{q}^1 - h_{11}(\mathbf{q}^1)^*\mathbf{q}^1 = 0$$

y como habíamos tomado $\mathbf{q}^1$ unitario, resulta finalmente:

$$h_{11} = (\mathbf{q}^1)^* A\mathbf{q}^1$$

Finalmente, si llamamos $v = A\mathbf{q}^1 - h_{11}\mathbf{q}^1$, y se define $h_{21} = \|v\|_2$ se obtiene que:

$$\mathbf{q}^2 = \frac{v}{h_{21}}$$

La reiteración de estos pasos nos conduce de manera inmediata al algoritmo de la iteración de Arnoldi.

2.2. Algoritmo de la Iteración de Arnoldi

2.2.1. Método de Arnoldi

Input : Una matriz $A \in \mathbb{C}^{m\times m}$, un vector $b \in \mathbb{C}^m$ no nulo y un número $n \leq m$
Output: Matrices Q_n y $\overline{H}_n$, y si $n = m$ matrices Q unitaria y H Hessenberg tales que $Q^*AQ = H$

$\mathbf{q}^1 = \dfrac{b}{\|b\|_2}$
for $k = 1, \ldots, n$ **do**
  $v = A\mathbf{q}^k$
  for $j = 1, \ldots, k$ **do**
    $h_{jk} = (\mathbf{q}^j)^* v$
    $v = v - h_{jk}\mathbf{q}^j$
  end
  $h_{k+1,k} = \|v\|_2$
  $\mathbf{q}^{k+1} = \dfrac{v}{h_{k+1,k}}$
end
$Q_n = [\mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^n]$
$\overline{H}_n = H$

$$H = \begin{bmatrix} h_{11} & \ldots & h_{1n} \\ \vdots & \ddots & \vdots \\ h_{n1} & \ldots & h_{nn} \end{bmatrix}$$

$$Q = \begin{bmatrix} q_{11} & \ldots & q_{11} \\ \vdots & \ddots & \vdots \\ q_{n1} & \ldots & q_{nn} \end{bmatrix}$$

end

2.2.2. Conteo Operacional

Nos disponemos a realizar el conteo operacional del algoritmo expuesto anteriormente. Es remarcable el hecho de que el paso más costoso del algoritmo es aquel que requiere realizar el producto $A\mathbf{q}^k$. Una implementación efectiva de este producto puede reducir notablemente el costo operacional del algoritmo (puesto que este producto se repite en cada iteración).

Conviene notar que antes del primer **for**, para calcular $\mathbf{q}^1$ se realizan $2m - 1$ operaciones en el producto escalar de b consigo mismo, y 1 operación para tomar raíz cuadrada. A estas operaciones hay que agregarle las m divisiones de cada componente de b por la norma de dicho vector. En total, $3m$ operaciones.

Luego, dentro del **for** se realiza el producto *matriz-vector* $A\mathbf{q}^n$, el cual requiere $(2m - 1)m$ operaciones.

Dentro del segundo **for** para calcular cada h_{jk} se realizan $2m - 1$ operaciones, y luego, para obtener v, se realizan m multiplicaciones y m restas, es decir, $2m$ operaciones más.

Finalmente fuera del segundo **for** al calcularse $h_{k+1,k}$ se realizan $2m$ operaciones más, y para calcular $\mathbf{q}^{k+1}$ se realizan m divisiones.

Sumando todo, el conteo operacional es:

$$3m + \sum_{k=1}^{n} \left((2m-1)m + \left(\sum_{j=1}^{k} 4m - 1 \right) + 3m \right)$$

lo cual, tras una serie de reducciones, resulta ser $\mathcal{O}(2m^2n + 2mn^2)$.

Esto último revela que la iteración de Arnoldi es aún mejor que aquella que utiliza las reflexiones de Householder, cuyo algoritmo, en el caso de calcular explícitamente H y Q, tiene un costo de $\mathcal{O}(\frac{14n^3}{3})$, que es algo más que el $\mathcal{O}(4m^3)$ que acabamos de ver que cuesta Arnoldi. Más aún, la iteración de Arnoldi permite que el proceso de obtener las columas de Q se detenga en cualquier momento (no hace falta iterar las m veces en el primer **for**), en contraposición al algoritmo de Householder, que requiere que se lleven a cabo todas las iteraciones hasta que se tenga un conjunto ortonormal de columnas para Q.

Sin embargo, en favor de la iteración de Householder, debe decirse que esta última es algo más estable que la de Arnoldi. Este último hecho podría entenderse como algo intuitivo, ya que la iteración de Arnoldi se apoya en una idea muy similar a la de Gram-Schmidt modificado.

2.3. Los Subespacios de Krylov y la Iteración de Arnoldi

Hemos descripto un camino para deducir la iteración de Arnoldi de una manera razonablemente estable y computacionalmente eficiente. Sin embargo, podemos dar una interpretación teórica (que será útil en la discusión del método GMRES) de lo que sucede al llevar adelante dicho método.

Hay que advertir que, de la recursividad obtenida para generar los vectores $\mathbf{q}^{n+1}$, dada por

$$\mathbf{q}^{n+1} = \frac{A\mathbf{q}^n - \sum_{i=1}^{n} h_{in}\mathbf{q}^i}{h_{n+1,n}}$$

resulta claro que:

$$\begin{aligned}
\left\langle \{\mathbf{q}^1, \mathbf{q}^2\} \right\rangle &= \left\langle \{\mathbf{q}^1, A\mathbf{q}^1\} \right\rangle \\
\left\langle \{\mathbf{q}^1, \mathbf{q}^2, \mathbf{q}^3\} \right\rangle &= \left\langle \{\mathbf{q}^1, A\mathbf{q}^1, A^2\mathbf{q}^1\} \right\rangle \\
\left\langle \{\mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^n\} \right\rangle &= \left\langle \{\mathbf{q}^1, A\mathbf{q}^1, \ldots, A^{n-1}\mathbf{q}^1\} \right\rangle
\end{aligned}$$

lo cual nos remite a pensar en los subespacios de Krylov, que están definidos por:

$$\mathcal{K}_n = \left\langle \{\mathbf{q}^1, A\mathbf{q}^1, \ldots, A^{n-1}\mathbf{q}^1\} \right\rangle$$

para cada $n \leq m$. En particular, se observa que lo que la iteración de Arnoldi hizo fue hallar una base ortogonal para cada subespacio de Krylov $\mathcal{K}_n$, por lo tanto, suena razonable pensar en un enfoque de ortogonalización.

Lema 2.1 *Si H es una matriz de Hessenberg $m \times m$ y $n \in \mathbb{N}$, entonces para cada $1 \leq i \leq n$ la i-ésima columna de H^n tiene sólo las primeras $n + i$ entradas posiblemente no nulas.*

Demostración: Por inducción en n.

Si $n = 1$, como H es matriz de Hessenberg, la primera columna tiene sólo las primeras dos entradas posiblemente no nulas, y en general, la i-ésima columna tiene sólo las primeras $i + 1$ entradas posiblemente no nulas.

Supongamos que vale para n. Sea $1 \leq i \leq m$. Entonces:

$$(H^nH)^i = H^n(H)^i = H^n \begin{bmatrix} h_{1i} \\ \vdots \\ h_{i+1,i} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \sum_{j=1}^{i+1} h_{ji}(H^n)^j$$

Por hipótesis inductiva, $(H^n)^j$ tiene sólo las primeras $n+j$ entradas posiblemente no nulas. Luego, como $j = 1, \ldots, i+1$, resulta que: $\sum_{j=1}^{i+1} h_{ji}(H^n)^j$ tiene a lo sumo las primeras $n+(i+1)$ no nulas y el resto nulas. Luego, la columna i de H^{n+1} tiene sólo las primeras $(n+1)+i$ entradas posiblemente no nulas, como queríamos probar. $\square$

Consideremos ahora la *matriz de Krylov*, dada por:

$$K_m = [\mathbf{q}^1, A\mathbf{q}^1, \ldots, A^{m-1}\mathbf{q}^1]$$

Supongamos que $Q^*AQ = H$, donde Q y H son las matrices $m \times m$ dadas por la iteración de Arnoldi. Notemos que:

$$Q^*K_m = [Q^*\mathbf{q}^1, Q^*A\mathbf{q}^1, \ldots, Q^*A^{m-1}\mathbf{q}^1]$$

Además, nótese que $Q^*A^k = H^kQ^*$, y luego:

$$Q^*K_m = [Q^*\mathbf{q}^1, HQ^*\mathbf{q}^1, \ldots, H^{m-1}Q^*\mathbf{q}^1]$$

Además, como los $\mathbf{q}^j$ son ortonormales, resulta que $\mathbf{q}^* = \mathbf{e}_1$, por lo que:

$$Q^*K_m = [\mathbf{e}_1, H\mathbf{e}_1, \ldots, H^{m-1}\mathbf{e}_1].$$

Por el lema 2.1, resulta que para cada n, la primera columna de H^n tiene sólo las primeras $n+1$ entradas no nulas. Luego, $H^n\mathbf{e}_1$ tiene sólo las primeras $n+1$ entradas no nulas. Se sigue que la matriz del lado derecho de la última igualdad es triangular superior, y la denotaremos por R. Resulta entonces que:

$$Q^*K_m = R$$

o, lo que es equivalente,

$$K_m = QR$$

donde Q es ortogonal y R es triangular superior. De esto se observa lo que afirmábamos anteriormente: la iteración de Arnoldi determina la matriz Q de una factorización QR para la matriz de Krylov.

Si suponemos que K_m es inversible (lo cual suele ocurrir, a pesar de que la matriz K_m está mal condicionada), sucede que:

$$AK_m = [Ab, A^2b, \ldots, A^mb]$$

Es decir:

$$AK_m = K_m[\mathbf{e}_2, \mathbf{e}_3, \ldots, \mathbf{e}_m, -\mathbf{c}]$$

donde $\mathbf{c} = -K_m^{-1}A^mb$. Denotando C a la matriz del lado derecho, resulta que:

$$A = K_mCK_m^{-1}$$

de donde A es una matriz semejante a C (que es una *matriz compañera*).

Veamos que la última ecuación conduce a otra deducción de la iteración de Arnoldi. En efecto, tomamos una factorización QR de K_m, pongamos $K_m = QR$. Entonces, ciertamente, vale que:

$$\begin{aligned} C &= K_m^{-1}AK_m \\ &= R^{-1}Q^*AQR \end{aligned}$$

lo cual sucede si y sólo si

$$Q^*AQ = \underbrace{RCR^{-1}}_{:=H}.$$

Lema 2.2 *Si R es triangular superior y H es triangular superior de Hessenberg, entonces HR y RH son triangulares superiores de Hessenberg.*

Por el lema 2.2 se tiene que $Q^*AQ = H$ que es una matriz triangular superior de Hessenberg (pues R y R^{-1} son triangulares superiores y C es una matriz de Hessenberg). Esta es otra manera de obtener la iteración de Arnoldi, más inestable pero importante desde el punto de vista teórico. Además, nos permite interpretar que la iteración de Arnoldi proyecta la matriz A ortogonalmente sobre el espacio de Krylov $\mathcal{K}_m$.

3. El Método GMRES

3.1. Descripción del método GMRES

El método de los Residuos Mínimos Generalizados (GMRES) apareció en la década de los 80' y se debe a Youcef Saad y Martin H. Schultz.

Mientras que muchos de los métodos iterativos para resolver sistemas lineales suelen estar pensados para trabajar con matrices definidas positivas, o matrices *ralas*, el método GMRES puede ser usado para sistemas lineales con matrices cuadradas inversibles arbitrarias.

La idea clave de este método, consistirá en pensar en resolver un problema de *cuadrados mínimos* en cada iteración. En pocas palabras, en el paso n de la iteración, trataremos de hallar un vector $x_n \in \mathcal{K}_n$ tal que minimice $\|r_n\|_2 = \|Ax_n - b\|_2$. Veamos cómo resolver dicho problema.

Supongamos que

$$K_n = [b, Ab, A^2b, \ldots, A^{n-1}b]$$

es la n-ésima matriz de Krylov (que es de tamaño $m \times n$). El vector x_n que estamos buscando se puede pensar como $x_n = K_n c$ con $c \in \mathbb{C}^n$, de manera que x_n está en $\mathcal{K}_n$ (que es el espacio columna de K_n). Luego:

$$\min_{x_n \in \mathcal{K}_n} \|r_n\|_2 = \min_{x_n \in \mathcal{K}_n} \|Ax_n - b\|_2 = \min_{c \in \mathbb{C}^n} \|AK_n c - b\|_2.$$

Una manera obvia de encontrar c tal que este mínimo se alcance, es utilizar una factorización QR para AK_n. Sin embargo, esto es costoso, inestable e ineficiente. Es aquí donde entra en juego la iteración de Arnoldi.

Supongamos que, en el algoritmo de Arnoldi, con entradas A, b y n, se obtiene la matriz $Q_n \in \mathbb{C}^{m \times n}$ como salida. Denotemos $\{\mathbf{q}^1, \ldots, \mathbf{q}^n\}$ a las columnas de esta matriz. Es claro que este conjunto es una base ortonormal para el espacio $\mathcal{K}_n$, por lo que se dijo en la sección anterior. Luego, existe un vector $y \in \mathbb{C}^n$ tal que $x_n = Q_n y$. Luego, lo que se busca es:

$$\min_{x_n \in \mathcal{K}_n} \|r_n\|_2 = \min_{y \in \mathbb{C}^n} \|AQ_n y - b\|_2$$

Pero, por cómo desarrollamos la iteración, se cumple que $AQ_n = Q_{n+1}\overline{H}_n$. Luego, queremos:

$$\min_{y \in \mathbb{C}^n} \|Q_{n+1}\overline{H}_n y - b\|_2$$

Como Q_{n+1} es una matriz con columnas ortonormales, al multiplicar por Q_{n+1}^* no se modifican las normas euclídeas, de donde basta hallar:

$$\min_{y \in \mathbb{C}^n} \|Q_{n+1}^* Q_{n+1}\overline{H}_n y - Q_{n+1}^* b\|_2 = \min_{y \in \mathbb{C}^n} \|\overline{H}_n y - Q_{n+1}^* b\|_2.$$

Además, podemos hacer otra simplificación: como $b/\|b\| = \mathbf{q}^1$, resulta que b es ortogonal a $\mathbf{q}^2, \ldots, \mathbf{q}^{n+1}$, y además, $(\mathbf{q}^1)^* b = \|b\|$. Luego, $Q_{n+1}^* b = \|b\|\mathbf{e}_1$, y entonces, hay que encontrar:

$$\min_{y \in \mathbb{C}^n} \|\overline{H}_n y - \|b\|\mathbf{e}_1\|_2.$$

3.2. Algoritmo GMRES

3.2.1. Método GMRES

Ahora, de los últimos cálculos efectuados, debería ser evidente cómo desprender el algoritmo para resolver el sistema lineal $Ax = b$ usando el método GMRES. Notemos que se requiere una rutina efectiva para resolver el problema de cuadrados mínimos en cada paso. Una forma aconsejable es utilizar una descomposición QR (utilizando, por ejemplo, las rotaciones de Givens). Esto no es tan costoso debido a la estructura de Hessenberg de la matriz que se debe descomponer. Es vital aprovechar dicha distribución de ceros para reducir el costo del algoritmo GMRES.

Input : Una matriz $A \in \mathbb{C}^{m\times m}$ inversible, un vector $b \in \mathbb{C}^m$ no nulo y un número $n \leq m$
Output: Solución aproximada x al sistema lineal $Ax = b$
for $k = 1, \ldots, n$ **do**
  Realizar el paso k de la iteración de Arnoldi, y obtener las matrices $\overline{H}_k$ y Q_k.
  Resolver el problema de cuadrados mínimos $\|\overline{H}_k y - \|b\|\mathbf{e}_1\|$ (por ejemplo con QR).
  $x_k = Q_k y$.
end
end

3.2.2. Conteo Operacional

Vamos a realizar el conteo operacional de cada paso del ciclo en el algoritmo GMRES. Antes, debemos hacer algunas observaciones: el algoritmo podría mejorarse para evitar realizar una factorización QR en cada paso, mediante una factorización que se vaya actualizando en cada paso. Esto permitiría ahorrar muchas operaciones. No obstante, el costo del algoritmo seguiría siendo casi el mismo.

El costo de la iteración k de Arnoldi tiene un costo de $2m^2 + 2m + 2mk^2 + 2mk - \frac{k^2+k}{2}$. Resolver el problema de cuadrados mínimos para la matriz de Hessenberg $\overline{H}_k$ (debido a su estructura) se puede llevar a cabo usando rotaciones de Givens con $3k^2$ operaciones. El producto $Q_k y$ se puede llevar a cabo en $k(2k + 1)$ operaciones, arrojando un total de:

$$\sum_{k=1}^{n} \left(2m^2 + 2m + 2mk^2 + 2mk - \frac{k^2 + k}{2} + 3k^2 + k(2k + 1) \right)$$

lo cual resulta ser $\mathcal{O}(2m^2 n + \frac{2}{3}mn^3)$, y para $n \ll m$, $\mathcal{O}(2m^2 n)$. Si se realizaran todas las iteraciones (o sea, si $m = n$), entonces el algoritmo tendría un costo de $\mathcal{O}(\frac{2}{3}m^4)$ lo cual es excesivamente costoso. Queda en evidencia que la fuerza del algoritmo depende de que la solución sea razonable para un $n \ll m$.

3.2.3. Convergencia del método GMRES

Vamos a probar un teorema que revelará bajo qué condiciones es esperable que el algoritmo sea verdaderamente útil:

Teorema 3.1 *Sean $A \in \mathbb{C}^{m\times m}$ inversible y $b \in \mathbb{C}^m$. Si el grado del polinomio minimal de A es n, entonces el algoritmo* GMRES converge a la solución de $Ax = b$ en a lo sumo n pasos.

Demostración: Supongamos que q es el polinomio minimal de A, y que q tiene grado n. Entonces, podemos escribir:

$$q(x) = c_0 + c_1 x + \ldots + c_n x^n$$

donde, ciertamente, $q(A) = 0$ y $c_n \neq 0$. Notemos que también debe ser $c_0 \neq 0$. En efecto, supongamos que $c_0 = 0$, entonces:

$$0 = q(A) = c_1 A + c_2 A^2 + \ldots + c_n A^n$$

de donde, como A es inversible, multiplicando por A^{-1} a ambos lados:

$$0 = c_1 + c_2 A + \ldots + c_n A^{n-1}$$

y entonces hay un polinomio de grado menor o igual a $n-1$ tal que A lo anula. Absurdo, pues el polinomio minimal de A es de grado n, y entonces todo polinomio de grado menor no puede ser anulado por A.

Como $c_0 \neq 0$, se obtiene:

$$0 = I + \frac{c_1}{c_0} A + \ldots + \frac{c_n}{c_0} A^n$$

de donde resulta:

$$\begin{aligned}
I &= -\left(\frac{c_1}{c_0} A + \ldots + \frac{c_n}{c_0} A^n\right) \\
A^{-1} &= -A^{-1}\left(\frac{c_1}{c_0} A + \ldots + \frac{c_n}{c_0} A^n\right) \\
A^{-1} &= \frac{-c_1}{c_0} + \frac{-c_2}{c_0} A^1 + \ldots + \frac{-c_n}{c_0} A^{n-1} \\
A^{-1} b &= \frac{-c_1}{c_0} b + \frac{-c_2}{c_0} A^1 b + \ldots + \frac{-c_n}{c_0} A^{n-1} b \\
x &= \widetilde{c}_1 b + \widetilde{c}_2 A^1 b + \ldots + \widetilde{c}_n A^{n-1} b
\end{aligned}$$

es decir, la solución $x \in \langle b, Ab, \ldots, A^{n-1}b \rangle = \mathcal{K}_n$, y por lo tanto el método debe converger en a lo sumo n pasos. $\square$

Se ha visto, entonces, que si A es una matriz cuyo polinomio minimal tiene grado $n \ll m$, entonces el método GMRES es muy poderoso, pues en a lo sumo n iteraciones se habrá llegado a la solución al sistema.

Hay que remarcar que, en general, una matriz aleatoria compleja de tamaño $m \times m$ tiene un polinomio minimal de grado m. Esto sucede debido a que las matrices aleatorias suelen tener m autovalores distintos. Para casos en los que se trabaja con matrices especiales, en los que los autovalores tienen multiplicidades relativamente grandes, el método GMRES es totalmente eficaz.

4. Aplicación: Page Rank

4.1. Del Page Rank a un problema numérico

El Page Rank es lo que determina el valor o la *importancia* que tiene una determinada página web. Esto es algo que Google (o cualquier buscador de características similares) tiene en cuenta a la hora de exhibir los resultados a una búsqueda realizada.

Para ésto, se determina que:

- La importancia de una página web depende sólo de las páginas web que la enlazan (es decir, que contienen un enlace a dicha página).
- Una página web reparte su *importancia* entre todas las páginas web que enlaza.

Dicho esto, si hay n páginas en la web, sea P la matriz $n \times n$ donde el elemento P_{ij} es la probabilidad de moverse de la página i a la página j en un sólo click.

Vale aclarar que hay muchas páginas que no contienen *outlinks* (es decir, no enlazan a ninguna página). Antes de proseguir, daremos una pequeña definición:

Definición 4.1 *Una matriz A de tamaño $n \times n$ se dice* estocástica *por filas si la suma de los elementos de cada fila es igual a 1. Análogamente, decimos que es* estocástica *por columnas si la suma de los elementos de cada columna es igual a 1. Decimos que es* estocástica *si es estocástica por filas y por columnas.*

Supongamos que la página i no enlaza a ninguna otra página. Entonces, es claro que $P_{ij} = 0$ para cada $1 \leq j \leq n$. Es decir, la fila i-ésima de P tiene suma 0, y por lo tanto P no es estocástica por filas. Sin embargo, podemos arreglar ésto tomando $\widetilde{P} = P + dv^t$ donde:

- d es un vector de n componentes, tales que $d_i = 1$ si la i-ésima fila de P corresponde a un nodo sin outlinks y 0 en caso contrario.
- v es un vector de n componentes, que se corresponde con alguna distribución de probabilidad sobre las páginas, por ejemplo, asignando que la página i tenga enlaces a todas las n páginas, en cuyo caso la fila i quedaría $\frac{1}{n}(1, \ldots, 1)$ y como n es el total de páginas la influencia de este cambio en el resultado es mínima.

Ahora, existe un segundo obstáculo, que es el caso de un grupo de páginas cuyos enlaces van sólo entre sí, es decir, que no enlazan a ninguna página exterior. Esto se traduce en que la matriz $\widetilde{P}$ que tomamos por ahora podría no ser irreducible.

Para solucionar esto, pensemos en el caso ideal, con una matriz B, donde todas las páginas se enlazan entre todas y de la misma forma (por ejemplo, que todas tengan la misma probabilidad de enlazarse, i.e. $B_{ij} = \frac{1}{n}$ para cada i, j). A B la expresamos como $B = ev^t$ donde $e = (1, \ldots, 1)$ y v es la distribución de probabilidad que utilizamos antes.

Consideramos una matriz A que sea combinación entre la matriz $\widetilde{P}$ considerada hasta ahora y la matriz ideal B, y transponemos para lograr que la entrada A_{ij} represente la probabilidad de ir de la página j a la i por medio de un sólo enlace. Tenemos, entonces:

$$\begin{aligned} A &= [\alpha\widetilde{P} + (1-\alpha)B]^t \qquad 0 < \alpha < 1 \\ &= \alpha P^t + v[\alpha d^t + (1-\alpha)e^t] \\ &= \alpha P^t + vw^t \end{aligned}$$

donde $0 < \alpha < 1$ se llama factor *damping* y $w = \alpha d^t + (1-\alpha)e^t$. Notemos que A^t es estocástica por filas, por lo que A es estocástica por columnas.

Teorema 4.1 (Teorema de Perron) *Sea A una matriz $n \times n$ de entradas positivas. Entonces:*

- *Existe un número real $r > 0$ tal que r es autovalor de A y $\rho(A) = r$. Además, los demás autovalores de A tienen módulo menor que r.*
- *El autoespacio generado por los autovectores asociados a r es unidimensional.*
- *Existe un vector de entradas positivas v, asociado al autovalor r. Además, cualquier otro autovector asociado a otro autovalor no puede tener todas sus entradas positivas.*

Como A tiene entradas positivas, entonces A^t también. Por el Teorema 4.1, como $A^t e = e$, resulta que 1 es un autovalor de A^t y tiene asociado un autovector positivo. Entonces 1 es el autovalor de Perron de la matriz A^t, y por lo tanto es el autovalor dominante de dicha matriz. Como los autovalores de A y A^t son los mismos, entonces se tiene que 1 es el autovalor dominante de A, aunque ahora desconocemos el autovector dominante. Ahora buscamos el autovector Page Rank que satisface:

$$Ax = x.$$

Notemos que este vector x es una asignación válida de PageRank, pues, si x_i es el valor de PageRank de la página i, $1 \leq i \leq n$, queremos que este valor sea igual a la suma de todos los PageRanks

que le entregan las páginas que la enlazan, es decir $x_i = \sum_{j=1}^{n} A_{ij}x_j$, luego, el vector que buscamos, efectivamente debe satisfacer $Ax = x$.

Debemos notar que un vector Page Rank derivado de un α muy grande (aunque no tan cerca de 1), como podría ser 0.99, quizás sea un vector Page Rank más verdadero. Por un lado, mientras más chico es el número α, es más fácil de computar el vector Page Rank*; pero, por otro, mientras más cercano sea α a 1, la matriz de Google estará más cerca de la matriz original de los links de la web.

Si el autovalor más grande no está bien separado del siguiente, que es lo que sucede cuando α está suficientemente cerca de 1, entonces es necesario utilizar métodos más sofisticados.

El álgebra lineal numérica moderna utiliza los métodos de subespacios de Krylov en diferentes procedimientos iterativos para resolver problemas como el de Page Rank.

4.2. Arnoldi y GMRES para resolver el problema de Page Rank

Golub y Greif, en 2006, utilizaron diferentes versiones del algoritmo de Arnoldi. En especial, uno basado en una combinación del proceso de Arnoldi y en la descomposición en valores singulares, y reside en el conocimiento del autovalor más grande de la matriz de Google.

Por otro lado, Page Rank puede ser formulado como un sistema lineal, y para este tipo de problemas, GMRES es uno de los algoritmos más populares para resolver sistemas lineales grandes y no simétricos.

Además, los métodos de subespacios de Krylov pueden ser combinados con otras estrategias aceleradoras de PageRank, por ejemplo una combinación del método de las potencias con el método de Arnoldi puede mejorar mucho la convergencia del primero, especialmente cuando α es muy cercano a 1.

Esencialmente, el algoritmo de Arnoldi trata con el problema de Page Rank desde un punto de vista de autovalores, mientras el algoritmo GMRES lo hace desde el punto de vista de un sistema lineal. No se conocen muchas cosas acerca de la relación de los dos enfoques. Sin embargo, el documento: "Arnoldi vs. GMRES for computing Page Rank: a theoretical contribution to Google's Page Rank problem" de G. Wu y Y. Wei hace un aporte sobre esto.

4.3. El algoritmo aplicado al problema

El problema de Page Rank puede ser reformulado como un gran sistema lineal. Con las ecuaciones anteriores, tenemos:

$$(\alpha P^t + vw^t)x = x \iff (I - \alpha P^t)x = vw^t x = \delta v$$

donde $\delta = w^t x$ es un escalar que en principio desconocemos, pero un valor particular de δ sólo significa reescalar x, que no tiene efecto en el Page Rank, así que tomamos $\delta = 1$. Queremos resolver:

$$(I - \alpha P^t)x = b, \qquad b = v.$$

Para esto, usamos el algoritmo GMRES, que como dijimos antes, en el paso k busca x_k que minimiza:

$$\min_{x \in \mathcal{K}_n} \|(I - \alpha P^t)x - b\|_2$$

que vimos que equivale a:

$$\min_{y \in \mathbb{C}^k} \|\widetilde{H}_k y - \|b\|\mathbf{e}_1\|, \qquad x_k = Q_k y.$$

La aproximación de GMRES es $x_G = Q_k y_G$ donde y_G es la solución al problema de cuadrados mínimos anterior.

Dado un vector $\mathbf{q}^1$ unitario, el paso $k \leq m$ de Arnoldi genera sucesivamente una base ortonormal $Q_k = [\mathbf{q}^1, \ldots, \mathbf{q}^k]$ para el subespacio de Krylov $\mathcal{K}_k$. Como dijimos antes, $AQ_k = Q_{k+1}\overline{H}_k$.

*Por métodos más simples como el de las potencias

Se sabe que el autovalor más grande de la matriz de Google es 1. Para este autovalor conocido, el método de Arnoldi busca un vector unitario $x_A \in \mathcal{K}_k$ que satisface:

$$\begin{aligned} \|(A-I)x_A\|_2 &= \min_{u \in \mathcal{K}_k} \|(A-I)u\|_2 \qquad (\star) \\ &= \min_{y \in \mathbb{C}^m} \|(A-I)Q_k y\|_2 \\ &= \min_{y \in \mathbb{C}^m} \|(\overline{H}_k - \widetilde{I})y\|_2 \end{aligned}$$

donde $\widetilde{I} = [I|0]^t$ es $(n+1) \times n$.

Entonces, resolver $(\star)$ resulta de resolver la descomposición en valores singulares de $\overline{H}_k - \widetilde{I}$ y $x_A = Q_k y_A$ donde y_A es el autovector correspondiente al menor valor singular de $\overline{H}_k - \widetilde{I}$.

Podríamos usar cualquiera de los algoritmos hasta el paso k, para un k pequeño, lo que implica que el vector de PageRank será calculado con cierto error. Sin embargo, en este problema el ranking es lo más importante (más que la precisión).

4.3.1. Relaciones entre ambos algoritmos

El artículo señalado anteriormente prueba y desarrolla (de forma extensa y no trivial) las siguientes relaciones entre ambos algoritmos, que en definitiva están más allá de los alcances de este informe. Dada su importancia, haremos un intento de exponerlas aquí, bajo el costo de omitir las demostraciones.

1. Notar que el subespacio de búsqueda es el mismo si $\mathbf{q}^1 = \frac{v}{\|v\|_2}$.

2. Como dijimos anteriormente, el algoritmo de Arnoldi puede ser entendido como un método de proyección ortogonal, en cambio GMRES es una técnica de proyección oblicua. Por lo tanto, los algoritmos de Arnoldi y GMRES no son matemáticamente equivalentes para el problema de PageRank.

3. La rapidez de convergencia para x_A (solución de Arnoldi) depende de $\sigma_{\text{mín}}(I - A_2)$ que señala la separación del autovalor 1 de los otros autovalores de A, y la rapidez de x_G (solución de GMRES) depende de $\kappa_2(I - \alpha P^t)$.

Por lo tanto, no se puede determinar cuál es definitivamente mejor que el otro, y la respuesta dependerá del problema en cuestión.

Referencias

[1] Greg Fasshauser, *Numerical Linear Algebra (Class Notes)*, Illinois Institute of Technology, 2006.

[2] David S. Watkins, *Fundamentals of Matrix Computations*, John Wiley & Sons, 1991.

[3] Wikipedia (english), Arnoldi Iteration.

[4] G H. Golub & C.F. Van Loan, *Matrix Computations*, Johns Hopkins Univ. Press, Baltimore, 1989.

[5] Wikipedia (english), Generalized minimal residual method.

[6] G. Wu & Y. Wei, Arnoldi vs. GMRES for Computing PageRank: A Theoretical Contribution to Google's PageRank Problem.