PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
Departamento de Matemáticas y Programa de Ingeniería Matemática y Computacional
IMT2111 Algebra Lineal Numérica

Tarea 4

Luis Felipe Silva De Vidts

Parte Teórica

Pregunta 6

Deduzca el Proceso de Lanczos a partir del Proceso de Arnoldi aplicado a una matriz A simétrica.

Como Arnoldi genera una matriz Hessenberg superior a partir de A, en el caso de que esta sea simétrica, cuando se aplica el algoritmo, las misma transformaciones que ocurren sobre las columnas son equivalentes en sus filas, por lo que se forma una matriz tridiagonal, que es lo mismo que realiza Lanczos.

Parte Práctica

1. Implemente *Conjugate Gradient (CG) y Conjugate Residual (CR).*[1]
   Ambas para resolver $Ax = b$ con A SPD.
   - He leído que, dado que CR hace dos productos matriz-vector por iteración, es común preferir CG sobre CR. Me gustaría ver algunos experimentos numéricos que afirmen o refuten esa aseveración.
   - Para que sus rutinas sean comparables en cuanto a eficiencia es necesario que ustedes realicen ambas implementaciones. Es decir, no usen librerías.
   - Comparen en diversos escenarios: Matrices con número de condición 2 alto, matrices con autovalores *cluster* bien y mal condicionadas.
   - Es recomendable que usen matrices *sparse* y $n \geq 1000$ (En general si la matrices son muy chicas no se observa nada de interés!).
   - En las páginas 178 (Algoritmo 6.17) y 182 (Algoritmo 6.19) del libro de Y. Saad tiene los pseudos códigos de CG y CR respectivamente.

Sobre la implementación

Cada función debe tener el siguiente encabezado

$$[x, flag, relres, iter, resvec] = METODO(A, b, tol, maxit, x_0)$$

donde los parámetros de entrada son los usuales y los de salida son

- x: Aproximación
- $flag$: variable que indica el estatus del método:
  0 indica que el método convergió con la tolerancia especificada(tol)
  1 Alcanzó el máximo de iteraciones SIN convergencia
  2 El método se estancó

[1] Escribo todo esto para ordenarme al hacer la tarea

- *relres*: valor del residual relativo al final del proceso $\frac{||b-Ax||}{||b||}$
- *iter*: número de iteraciones realizadas
- *resvec*: vector con el residual relativo por iteración

Al correr los algoritmos, use matrices muy densas de $n = 2000$, tolerancia de $1 * 10^{-15}$ y $maxiter = n$, dentro de la implementación determine que el algoritmo retornará $flag = 2$ en el caso en que lo que disminuia el error fuese menor que el nivel de tolerancia dado, lo que no significa que el método no convergío, sino que está avanzando muy lento o se quedo realmente estancado.

Caso	Método	$flag$	Iter	$\frac{\|\|b-A\tilde{x}\|\|}{\|\|b\|\|}$
A random	Conjugate Gradient	0	276	2.0874e-17
$K_2(A)$ =868.24	Conjugate Residual	2	271	3.7720e-17
A random	Conjugate Gradient	0	438	1.8809e-17
$K_2(A)$ =8.2e+09	Conjugate Residual	2	405	1.1249e-15
A autovalores cluster	Conjugate Gradient	0	16	1.4232e-18
$K_2(A)$ =1.0e+06	Conjugate Residual	0	16	1.1207e-18
A autovalores cluster	Conjugate Gradient	0	6	8.8442e-18
$K_2(A)$ =1.4	Conjugate Residual	0	6	9.1642e-18

En general notamos que los algoritmos se comportan de la misma manera en todos los casos, con pequeñas variaciones de cantidad de iteraciones y de errores relativos, sin embargo Conjugate Residuals realiza el doble de operaciones matriz vector por iteracion que Conjugate Gradients, lo que hace más conveniente usar Conjugate Gradients que Conjugate Residuals, ya que esa cantidad de operaciones extra no da una mejor solución, ni lo hace de forma más rápida.< >