LINEAR ALGEBRA: ANSWER TO HOMEWORK 9

VISHVAS VASUKI

1. 33.2

During some step n of Arnoldi iteration, $h_{n+1,n}$ is encountered.

1.1. a.

Theorem 1.1.1. *$AQ_n = Q_{n+1}\hat{H}_n$ can be simplified to $AQ_n = Q_n H_n$.*

Proof. As the last row of $\hat{H}_n$ is 0, no vector Aq_i has a component along q_{n+1}. Therefore the simplification holds. □

Remark 1.1.2. Implications on the structure of the $A = QHQ^*$: The submatrix $H_{n+1:m,1:n}$ is 0.

1.2. b.

Theorem 1.2.1. *K_n is an invariant subspace of A: $AK_n \subseteq K_n$.*

Proof. $K_n = \langle b, ..A_{n-1}b\rangle = \langle q_1, ..q_n\rangle$. So, any $v \in K_n$ can be written as $v = Q_n x$. So, $Av = AQ_n x = Q_n H_n x$. So, Av is in the column space of Q_n. So, Av is still in K_n.

Thus, $AK_n \subseteq K_n$. □

1.3. c.

Theorem 1.3.1. *$K_n = \langle b, ..A^{n-1}b\rangle$. Then, $K_n = K_{n+1} = K_{n+2}...$*

Proof. $K_{n+1} = \langle K_n, A^n b\rangle$. But, by the previous theorem, as $A^{n-1}b \in K_n$, $A^n b = AA^{n-1}b \in K_n$. Thus, $K_{n+1} \subseteq K_n$. But trivially, $K_n \subseteq K_{n+1}$. Thus, $K_n = K_{n+1}$.

Using the same argument inductively, we see that $K_n = K_{n+1} = K_{n+2}...$ □

1.4. d.

Theorem 1.4.1. *Each ew of H_n is an ew of A.*

Proof. As we had previously remarked, $H_{n+1:m,1:n}$ is 0. So, the characteristic polynomial of A, $det(A-lI)$ can be written as $det(H_{1:n,1:n}-lI_n)det(H_{n+1:m,n+1:m}-lI_{m-n})$.

So, any value of l which causes $det(H_{1:n,1:n}-lI_n) = 0$ also causes $det(A-lI) = 0$. Thence the result. □

1.5. e.

Theorem 1.5.1. *If A is non singular, then solution x to Ax=b lies in K_n.*

Proof. K_n is spanned by columns of Q_n. Let $Q_{n\perp}$ be a matrix whose columns form an orthonormal basis for the subspace of Range(A) orthogonal to K_n.

$$\begin{aligned}
\texttt{Let:} Q_n y + Q_{n\perp} y' &= x \\
Q_n y + Q_{n\perp} y' &= x \\
Q_n y - x &= -Q_{n\perp} y' \\
AQ_n y - Ax &= -AQ_{n\perp} y'
\end{aligned}$$

$Q_n y$ is in K_n; and K_n being invariant, $AQ_n y$ is also in K_n. Also, $Ax = b = \|b\| q_1$ is also in K_n. So, $AQ_n y - Ax$ is also in K_n.

[**Incomplete**].

□

2. 36.1

Theorem 2.0.2. *A is real and symmetric. $r(x) = \frac{x^T Ax}{x^T x}$. Stationary values of r(x) are the ew of A. Then, Ritz values at step n of the Lancoz iteration are the stationary values of r(x) if x is restricted to K_n.*

Proof. If x is restricted to K_n, it can be written as $Q_n y$. Then:

$r(x) = r(Q_n y) = \frac{y^T Q_n^T A Q_n y}{y^T y} = \frac{y^T T_n y}{y^T y}$. Let us denote this by $r'(y)$. We note that this is the Rayleigh quotient for the matrix T_n, and that, whenever y is an ev this quantity is the corresponding ew.

Following the analysis in the Rayleigh quotients chapter, we see that $\nabla r'(y) = \frac{2}{y^T y}(T_n y - r'(y)y)$. This is 0 exactly when y is an ev of T_n and $r'(y)$ is the corresponding ew of T_n.

So, stationery values of r'(y) and r(x) restricted to x of the form $x = Q_n y$ are exactly the same: $r'(y) = 0 \equiv r(Q_n y) = 0 \equiv r(x) = 0$. □

3. 38.5

Minimizing $f(x) = 2^{-1} x^T Ax - x^T b$ using steepest descent: $p_n = r_n$.

3.1. a.

Theorem 3.1.1. $\nabla f(x) = -r$.

Proof.

$$\begin{aligned}
f(x) &= 2^{-1} x^T Ax - x^T b \\
\nabla f(x) &= \nabla 2^{-1} x^T Ax - \nabla x^T b \\
&= Ax - b \\
&= -r
\end{aligned}$$

□

3.2. **b.**

Theorem 3.2.1. *Optimal step* $a_n = \frac{r_{n-1}^T r_{n-1}}{r_{n-1}^T A r_{n-1}}$.

Proof. We want to find an optimal a_n which minimize $f(x_n)$. Our search direction is r_{n-1}; so we want to find $x_n = x_{n-1} + a_n r_{n-1}$.

$$\begin{aligned} \nabla f(x_n) &= 0 \\ Ax_n - b &= 0 \\ A(x_{n-1} + a_n r_{n-1}) - b &= 0 \\ b - Ax_{n-1} &= a_n A r_{n-1} \\ r_{n-1} &= a_n A r_{n-1} \\ a_n &= \frac{r_{n-1}^T r_{n-1}}{r_{n-1}^T A r_{n-1}} \end{aligned}$$

□

3.3. **c.** The full steepest descent iteration:

$x_0 = 0, r_0 = b.$
foreach n = *1,2 ..* **do**
  $a_n = \frac{r_{n-1}^T r_{n-1}}{r_{n-1}^T A r_{n-1}}$
  $x_n = x_{n-1} + a_n r_{n-1}$
  $r_n = b - Ax_n$
end

4. 38.6

[**Incomplete**].

5

Following link provides a data structure to store sparse matrices:
`www.cs.utexas.edu/~inderjit/courses/cs383c/sparse_matrices.txt`
Write a matlab code using the above specified data structure to compute the matrix-vector product $y = Ax$ in O(nz), where nz is the number of non-zeros in the sparse matrix A. Also write a matlab code to compute $y = A^T x$ in O(nz). Note that you are not allowed to store A^T into a new matrix.

[**Incomplete**].