# CS6210 - Homework/Assignment-4

Arnab Das(u1014840)

November 6, 2016

## 1: Chapter-6, question-4

**(a)** The techniques discussed in this chapter are for polynomial data fitting and not exponential data fitting, hence cannot be applied directly to $u(t)$ .

**(b)** Given approximation of the form

$$u(t) = \gamma_1 \exp(\gamma_2 t)$$

and provided data points as $(t_1, z_1), (t_2, z_2), \ldots, (t_m, z_m)$, where $z_i > 0, i = 1, 2, \ldots, m$ and $m > 0$.
Considering instead the following approximation:

$$v(t) = lnu(t) = (ln\gamma_1) + \gamma_2 t$$

such that the data points become $(t_1, b_1), (t_2, b_2), \ldots, (t_m, b_m)$ where $b_i = lnz_i$ and

$$v(t) = x_1 + x_2 t$$

such that $x_1 = ln\gamma_1$ and $x_2 = \gamma_2$ and $v(t) = lnu(t)$
Then for solving $A^T Ax = A^T b$, we define the following matrices:
$A = \begin{bmatrix} 1 & t_1 \\ 1 & t_2 \\ \ldots & \ldots \\ 1 & t_m \end{bmatrix}$ leading to $B = A^T A = \begin{bmatrix} \sum_{i=1}^{m} 1 = m & \sum_{i=1}^{m} t_i \\ \sum_{i=1}^{m} t_i & \sum_{i=1}^{m} {t_i}^2 \end{bmatrix}$ and $b = \begin{bmatrix} t_1 \\ t_2 \\ \ldots \\ t_m \end{bmatrix}$ leading to $A^T b = \begin{bmatrix} \sum_{i=1}^{m} b_i \\ \sum_{i=1}^{m} t_i b_i \end{bmatrix}$ Thus solving

$$B = A^T Ax = A^T b$$

we get the following two equations:

$$x_1 + x_2 = 1 \tag{1}$$

and

$$3x_1 + 5x_2 = 4.9 \tag{2}$$

Solving (i) and (ii) , we get: $x_1 = ln\gamma_1 = 0.5$ and $x_2 = \gamma_2 = 0.95$
Thus, we have ,

$$v(t) = lnu(t) = ln(0.5) + 0.95t$$
$$=> u(t) = 0.5 \times exp(0.95t)$$

(Answer).

## 2: Chapter-6, question-5

**(a)** For tall and skinny matrices, A, of the system of equations , $Ax = b$, the number of rows is much larger than the number of columns. Let the number of rows be m and the number of columns n, then in such cases generally $m >> n$. These systems are overdetermined and b is generally not in the range space of A. Thus applying LU does not gives a unique solution to Ax=b. Instead, a better way to approach such problems is to minimize the residual $||b - Ax||$, such that within the tolerance of the residual we have the best solution for x. When transforming to the normal equation, $A^T Ax = A^T b$, here one can use LU decomposition since we have $n \times n$ matrix, however cholesky decomposition is a better choice here since the matrix $A^T A$ is symmetric positive definite. In case of the QR decomposition, which has an upper triangular part in R, however the Q matrix allows us to extract the upper-triangular system of equation whose solution leads to a solution of the least square problem. Here, the orthonormal behaviour of Q is used to transform into

equivalent set of equations such that the **norms are not affected**. Finally, SVD is used more in cases where A is rank deficient or nearly rank deficient, in which cases LU cannot be used. Thus, in all the cases LU directly doesn't fits the scenario for application except that LU requires to be slightly modified so that for every column it zeroes out, the vector of its remaining elements is orthonormal to the previous columns. With this introduction of orthonormality, it can be used in QR decomposition.

**(b)** In the normal equation: the way condition number came to be $K(B) = K^2(A)$. This was because of the folliwing derivation:
In normal equation we were solving:

$$A^T Ax = A^T b$$

and

$$(A^T A)^{-1} A^T = V S^{-2} V^T V \sum^T U^T = V(S^{-1}0)U^T$$

where A = $U \sum V^T$ , V and U are orthogonal matrices and $\sum$ is a diagonal matrix with singular values of A along the diagonal, and hence $A^T A = V \sum U^T U \sum V^T = V \sum^2 V^T$
Thus $||(A^T A)^{-1} A^T|| = ||(S^{-1}0)||$ and $||((A^T A)^{-1} A^T)^{-1}|| = ||V(S0)U^T||$
Thus the condition number becomes:

$$K(B = A^T A) = \frac{\lambda_1}{\lambda_n} = \frac{\sigma_1^2}{\sigma_n^2} = K^2(A)$$

Now, for QR, we write $x = (A^T A)^{-1} A^T b$, and we have represented A=QR by a QR decomposition where Q is orthonormal and of same dimension as A. The following translation results in the final form:

$$x = (A^T A)^{-1} A^T b = (R^T Q^T QR)^{-1} R^T Q^T b = (R^T R)^{-1} R^T Q^T b) = R^{-1} Q^T b$$

Hence, we get the following relation,

$$x = R^{-1} Q^T b$$

Notice, that multiplication by orthogonal matrices do not affect the norms, thus,

$$||R^{-1} Q^T|| = ||R^{-1}|| = ||A^{-1}||$$

and also,

$$||(Q^T)^{-1} R|| = ||R|| = ||A||$$

Thus the condition number in this case comes to be $||A||||A^{-1}|| = K(A)$.
The main saving comes due to usage of the orthonormal decomposition for the transformations, since the transformations using q only transofrms them in space without affecting the norm values, while for the previous normal equation, the singular values were getting multiplied during creation $A^T A$, introducing change in norm values during the transformations.

---

**3: Chapter-8, question-8**

---

**(a)** Given a rank deficient matrix, we analyse here its effect on the Gra,-Schmidt process. Consider the span of three vectors $v_1, v_2 and v_3$, such that $v_3$ is linearly dependent on $v_1$ and $v_2$. We can write $v_3$ as a linear combination of $v_1$ and $v_2$, as:

$$\mathbf{v_3 = av_1 + bv_2}$$

Let $\mathbf{u_i}$ denote the orthonormal unit vectors we generate along the Gram-Scmidt process. Then, for the first vector, $\mathbf{v_1}$, we have

$$\mathbf{u_1} = \frac{\mathbf{v_1}}{|\mathbf{v_1}|}$$

And the orthogonal transformation of $\mathbf{v_2}$ that is orthogonal to $\mathbf{v_1}$ is

$$\mathbf{y_2} = \mathbf{v_2} - (\mathbf{v_2}.\mathbf{u_1})\mathbf{u_1}$$

Then, the next orthonormal vector, $\mathbf{u_2}$ will be along $\mathbf{y_2}$, such that $\mathbf{u_2} = \frac{\mathbf{y_2}}{|\mathbf{y_2}|}$.
Now the orthogonal transformation of $\mathbf{v_3}$, say $\mathbf{y_3}$, that is orthogonal to the span$(\mathbf{u_1}, \mathbf{u_2})$, will be,

$$\mathbf{y_3} = \mathbf{v_3} - \Big((\mathbf{v_3}.\mathbf{u_1})\mathbf{u_1} + (\mathbf{v_3}.\mathbf{u_2})\mathbf{u_2}\Big)$$

$$\mathbf{y_3} = \mathbf{v_3} - \Big(((\mathbf{av_1} + \mathbf{bv_2}).\mathbf{u_1}).\mathbf{u_1} + ((\mathbf{av_1} + \mathbf{bv_2}).\mathbf{u_2})\mathbf{u_2}\Big)$$

$$\mathbf{y_3} = \mathbf{v_3} - \Big(\mathbf{av_1} + (\mathbf{bv_2}.\mathbf{u_1})\mathbf{u_1} + (\mathbf{av_1}.\mathbf{u_2})\mathbf{u_2} + (\mathbf{bv_2}.\mathbf{u_2})\mathbf{u_2}\Big)$$

Here:
$(b\mathbf{v_2}.\mathbf{u_1})\mathbf{u_1} = \mathbf{b}(\mathbf{v_2} - \mathbf{y_2})$
$(a\mathbf{v_1}.\mathbf{u_2})\mathbf{u_2} = \mathbf{0}$
$(b\mathbf{v_2}.\mathbf{u_2})\mathbf{u_2} = \mathbf{by_2}$

Replacing them and the value of $\mathbf{v_3}$ in the above equation we get:

$$\mathbf{y_3} = \mathbf{av_1} + \mathbf{bv_2} - (\mathbf{av_1} + \mathbf{bv_2} - \mathbf{by_2} + \mathbf{0} + \mathbf{by_2}) = \mathbf{0}$$

Thus $\mathbf{y_3}$, comes out to be zero, when we encounter the linearly dependent vector. (Answer).

**(b)** The classical pseudoCode of Gram-Scmidt is as below:

**Classical-GS**

**for** k=1:n
  w = $a_k$
  **for** j=1:k-1
    $r_{jk} = q_j^T w$
  end
  **for** j=1:k-1
    w = w - $r_{jk} q_j$
  end
  $r_{kk} = ||w||$
  $q_k = w/r_{kk}$
end

**Modified-GS**

**for** k=1:n
  w = $a_k$
  **for** j=1:k-1
    $r_{jk} = q_j^T w$
    $w = w - r_{jk} q_j$
  end
  $r_{kk} = ||w||$
  $q_k = w/r_{kk}$
end

Suppose at the k'th iteration the orthonormal q's already calculated are $Q_{k-1} = [q_1, q_2, \ldots, q_{k-1}]$. In the **classical** case, we first calculate the projections, so suppose the calculated values of the projections in the

k'th iteration are

$$[r_{ik}, r_{2k}, \ldots, r_{k-1,k}]$$

where, $r_{jk} = q_j^T w$, where w in the k'th iteration is initialised to $a_k$, the vector whose orthogonal transformation is done in the k'th step. Then, these projections along the corresponding orthonormal directions are subtracted from w, in the second inner loop, resulting in:

$$w = w - r_{1k}\mathbf{q_1} - \mathbf{r_{2k}q_2} - \cdots - \mathbf{r_{k-1,k}q_{k-1}} \tag{3}$$

In the modified version, instead of precomputing all the $r_{jk}$'s at a time, we compute $r_{jk}$ and substract from w, thus always *orthogonalizing* against the currently computed version. Suppose, we are in the k'th iteration, and j=1 results in the following computation

$$w_1 = w - r_{1k}q_1 = w - (q_1^T w)q_1$$

Then for j=2 in the k'th iteration:

$$w_2 = w_1 - r_{2k}q_2 = w - (q_1^T w)q_1 - q_2^T(w - (q_1^T w)q_1)q_2$$
$$w_2 = w_1 - r_{2k}q_2 = w - (q_1^T w)q_1 - (q_2^T.w)q_2 - 0$$
$$w_2 = w_1 - r_{2k}q_2 = w - r_{1k}\mathbf{q_1} - \mathbf{r_{2k}q_2} - \mathbf{0}$$

And similarly for increasing j's(j limits to k) it holds true since $q_i and q_j$ are orthonromal for $i \neq j$. Thus , in exact arithmatic the modified and classical version are numerically the same. (Proved).

**(c)**

---

**4: Chapter-7, Question-9**

---

For the iterative scheme, we have $x_{k+1} = x_k + \alpha_k p_k$, where, $p_k$ is the search direction and $\alpha_k$ is the step size. This includes the basic statioanry methods as well of the form, $x_{k+1} = x_k + M^{-1}r_k$.
Now, consider the given iterative scheme:

$$x_{k+1} = x_k + \alpha(b - Ax_k)$$

Since, $r_k = b - Ax_k$ is the residual at the k'th step, and in gradient descent , the search direction is in the reverse direction of the residual, Hence, $p_k = r_k$. Hence, for a fixed $\alpha$ we get:

$$M^{-1} = \alpha I$$

$$M = (\alpha I)^{-1}$$