coursework2

Problem 1:

In Lecture we considered application of the classical and modified Gram-Schmidt algorithms to produce a QR factorisation the problem Lauchli matrix

$$A = \begin{bmatrix} 1 & 1 & 1 \\ \ell & 0 & 0 \\ 0 & \ell & 0 \\ 0 & 0 & \ell \end{bmatrix} = QR$$

In several application areas,it is important to compute the columns of Q so that their orthogonality is as strict as possible. For these cases, reorthogonalisation can vastly improve performance of the algorithm. Specically, the Gram-Schmidt orthogonalisation process is repeated p times. For the classical Gram-Schmidt algorithm, this modication results in:

Algorithm CGS

for j = 1 **to** n **do**
    $u_j = a_j$
    for i = 1 **to** j - 1 **do**
        $r_{ij} = q_i^* a_j$
        $u_j = u_j - r_{ij} q_i$
    end for
    $r_{jj} = ||u_j||_2$
    $q_j = u_j / r_{jj}$
end for

$\Rightarrow$

Algorithm CGS with reorthogonalisation

for j = 1 **to** n **do**
    $u_j = a_j$
    for k = 1 **to** p **do**
        for i = 1 **to** j - 1 **do**
            $u_j = u_j - q_i^* u_j q_i$
        end for
    end for
    $q_j = u_j / ||u_j||_2$
end for

Notice that Algorithm 2 does not include the computation of the elements of R, since the projection step is repeated p times Base on the above

(a)

write the corresponding algorithm for the modified Gram-Schmidt (MGS) processs with reorthogonalisation, including with the computation of the elements of R

the original modified Gram Schmidt using orthogonal projectors is

---

Algorithm original MGS

---

for j = 1 **to** n **do**
    $u_j = a_j$
end for
for j = 1 **to** n **do**
    $r_jj = ||u_j||_2$
    $q_j = u_j/r_{jj}$
    for k = j+1 **to** n **do**
        $r_{jk} = q_j^* u_k$
        $u_k = u_k - r_{jk} q_j$
    end for
end for

---

with reorthogonalisation

Algorithm MGS with Reorthogonalisation and Computation of R

---

for j = 1 **to** n **do**
    $u_j = a_j$
end for
for j = 1 **to** n **do**
    $r_{jj} = ||u_j||_2$
    $q_j = u_j/r_{jj}$
    for i = 1 **to** p **do**
        for k = j + 1 **to** n **do**
            $r_{jk} = q_j^* u_k$
            $u_k = u_k - r_{jk} q_j$
        end for
    end for
end for

(b)

Apply CGS and MGS with one reorthogonalisation step (p = 2) to the Lauchi matrix A

we apply the conditions in the lecture, assuming ℓ is small or

$$\begin{cases} 1+\ell=1 \\ 1+\ell^2=1 \end{cases}$$

and from $1+\ell^2=1$, $\ell^2=0$, we can purge any ℓ^2

In CGS with reorthogonalization

first we initialize U with A, and R with an empty matrix the size of U, in the Lauchi matrix, n = 3

- $j=1: u_1 = \begin{bmatrix}1\\ \ell\\ 0\\ 0\end{bmatrix}$
    - since j - 1 = 0 < 1, the inner two loop do not execute
    - $q_1 = \frac{u_1}{||u_1||_2} = \frac{u_1}{\sqrt{1+\ell^2}} = u_1 = \begin{bmatrix}1\\ \ell\\ 0\\ 0\end{bmatrix}$
- $j=2: u_2 = \begin{bmatrix}1\\ 0\\ \ell\\ 0\end{bmatrix}$
    - k = 1
        - i = 1
        - $$\begin{aligned} u_2 = u_1 - q_1^* u_2 q_1 &= \begin{bmatrix}1\\ 0\\ \ell\\ 0\end{bmatrix} - \begin{bmatrix}1 & \ell & 0 & 0\end{bmatrix}\begin{bmatrix}1\\ 0\\ \ell\\ 0\end{bmatrix}\begin{bmatrix}1\\ \ell\\ 0\\ 0\end{bmatrix} \\ &= \begin{bmatrix}1\\ 0\\ \ell\\ 0\end{bmatrix} - \begin{bmatrix}1\\ \ell\\ 0\\ 0\end{bmatrix} \\ &= \begin{bmatrix}0\\ -\ell\\ \ell\\ 0\end{bmatrix} \end{aligned}$$
    - k = 2
        - i = 1

$$u_2 = u_2 - q_1^* u_2 q_1 = \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix} - [1 \quad \ell \quad 0 \quad 0] \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix} - (-\ell^2) \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}$$

- $$= \begin{bmatrix} \ell^2 \\ -\ell + \ell^3 \\ \ell \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix}$$

- $q_2 = \frac{u_2}{||u_2||_2} = \frac{u_2}{\sqrt{2\ell^2}} = \frac{1}{\sqrt{2}\ell} u_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}$

- $j = 3 : u_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \ell \end{bmatrix}$

- k = 1
- i = 1
-

$$u_3 = u_3 - q_1^* u_3 q_1 = \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix} - [1 \quad \ell \quad 0 \quad 0] \begin{bmatrix} 1 \\ 0 \\ 0 \\ \ell \end{bmatrix} \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 0 \\ 0 \\ \ell \end{bmatrix} - \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix}$$

- i = 2

-

$$u_3 = u_3 - q_2^* u_3 q_2 = \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix} - \frac{1}{2}\begin{bmatrix} 0 & -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix} - \frac{1}{2}\ell \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix}$$

- k = 2
- i = 1
-

$$u_3 = u_3 - q_1^* u_3 q_1 = \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix} - \begin{bmatrix} 1 & \ell & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix} \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix} - \frac{1}{2}\ell^2 \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}$$
$$= \begin{bmatrix} -\frac{1}{2}\ell^2 \\ -\frac{1}{2}(\ell + \ell^3) \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix}$$
$$= \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix}$$

- i = 2

-

$$u_3 = u_3 - q_2^* u_3 q_2 = \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix} - \frac{1}{2}\begin{bmatrix} 0 & -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix} - 0$$
$$= \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix}$$

- $q_3 = \frac{u_3}{||u_3||_2} = \frac{u_3}{\sqrt{\frac{1}{4}\ell^2 + \frac{1}{4}\ell^2 + \ell^2}} = \frac{u_3}{\sqrt{\frac{3}{2}}\ell} = \frac{1}{\sqrt{\frac{3}{2}}} \begin{bmatrix} 0 \\ -\frac{1}{2} \\ -\frac{1}{2} \\ 1 \end{bmatrix}$

we can get the corresponding Q U, this algorithm does not calculate R

In MGS with reorthogonalization:

first we initialize U with A, and R with an empty matrix the size of U, in the Lauchi matrix, n = 3

- $j = 1 : u_1 = \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}, r_{11} = \frac{1}{\sqrt{1+\ell^2}} = 1, q_1 = u_1 / r_{11} = \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix}$
  - when i = 1,
  - k = 2
  - $r_{12} = q_1^* u_2 = \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ 0 \\ \ell \\ 0 \end{bmatrix} = 1$
  - $u_2 = u_2 - 1 * q_1 = \begin{bmatrix} 1 \\ 0 \\ \ell \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix}$
  - k = 3
  - $r_{13} = q_1^* u_3 = \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ 0 \\ 0 \\ \ell \end{bmatrix} = 1$

- $u_3 = u_3 - 1 * q_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \ell \end{bmatrix} - \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix}$
- i = 2
- k = 2
- $r_{12} = q_1^* u_2 = \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix} = -\ell^2$
- $u_2 = u_2 + \ell^2 * q_1 = \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix} + \ell^2 \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \ell^2 \\ \ell(-1+\ell^2) \\ \ell \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix}$
- k = 3
- $r_{13} = q_1^* u_3 = \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ -\ell \\ 0 \\ \ell \end{bmatrix} = -\ell^2$
- $u_3 = u_3 + \ell^2 * q_1 = \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix} + \ell^2 \begin{bmatrix} 1 \\ \ell \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \ell^2 \\ -\ell(-1+\ell^2) \\ 0 \\ \ell \end{bmatrix} = \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix}$

$j = 2 : u_2 = \begin{bmatrix} 0 \\ -\ell \\ \ell \\ 0 \end{bmatrix}, r_{22} = \sqrt{2\ell^2} = \sqrt{2}\ell, q_2 = u_2/r_{22} = \frac{1}{\sqrt{2}} \begin{bmatrix} \ell \\ -1 \\ 1 \\ 0 \end{bmatrix}$

- when i = 1,
- k = 3
- $r_{23} = q_2^* u_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} \ell \\ -1 \\ 1 \\ 0 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix} = \frac{1}{\sqrt{2}}\ell$
- $u_3 = u_3 - \frac{1}{\sqrt{2}}\ell * q_2 = \begin{bmatrix} 0 \\ -\ell \\ 0 \\ \ell \end{bmatrix} - \frac{1}{2}\ell \begin{bmatrix} \ell \\ -1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix}$
- i = 2
- k = 3
- $r_{23} = q_2^* u_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} \ell \\ -1 \\ 1 \\ 0 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix} = 0$

- $u_3 = u_3 = \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix}$

  $j = 3, u_3 = \begin{bmatrix} 0 \\ -\frac{1}{2}\ell \\ -\frac{1}{2}\ell \\ \ell \end{bmatrix}, r_{33} = \frac{1}{\sqrt{\frac{3}{2}\ell}},$

- $q_3 = u_3/||u_3||_2 = \frac{1}{\sqrt{\frac{3}{2}}} \begin{bmatrix} 0 \\ -\frac{1}{2} \\ -\frac{1}{2} \\ 1 \end{bmatrix}$

  we can get the corresponding Q U R

(c)

in CGS

Algorithm CGS with reorthogonalisation

1: **for** j = 1 **to** n **do** $//n\ cycles$
2: $\quad u_j = a_j$
3: $\quad$ **for** k = 1 **to** p **do** $//p\ cycles$
4: $\quad\quad$ **for** i = 1 **to** j - 1 **do** $//1 + 2 + \cdots + n - 1$
5: $\quad\quad\quad u_j = u_j - q_i^* u_j q_i$
6: $\quad\quad$ **end for**
7: $\quad$ **end for**
8: $\quad q_j = u_j/||u_j||_2$
9: **end for**

assume $U \in \mathbb{R}^{m \times n}$

in line 5, since $q_i, u_j, q_i \in \mathbb{R}^m$ and $q_i^* u_j$ is a scalar

therefore the flop operation in this line is m + m + m

the first for vector minus, the second for vector dot product and the last of multiplying a scalar with a vector

in line 8, the ℓ_2 norm of u_j, first calculates the sqaure of all elements, and the adds them together, do square root then do the division, this takes m + (m-1) + m + 1 = 3m

therefore the total flop operations is
$n * (3m) + (\sum_{i=1}^{n-1} i * p * 3m) = 3mn(\frac{p(n-1)}{2} + 1) = 3 * 4 * 3 * 3 = 108$
in MGS, this is similar,

Algorithm MGS with Reorthogonalisation and Computation of R

1: **for** j = 1 **to** n **do**
2: $\quad u_j = a_j$
3: **end for**
4: **for** j = 1 **to** n **do**
5: $\quad r_{jj} = ||u_j||_2$
6: $\quad q_j = u_j / r_{jj}$
7: $\quad$ **for** i = 1 **to** p **do**
8: $\quad\quad$ **for** k = j + 1 **to** n **do**
9: $\quad\quad\quad r_{jk} = q_j^* u_k$
10: $\quad\quad\quad u_k = u_k - r_{jk} q_j$
11: $\quad\quad$ **end for**
12: $\quad$ **end for**
13: **end for**

line 2 takes no flops

line 5 and line 6 take 3m (see CGS line 8)

line 9 and line 10 is similar to CGS line 5

so the flop in the is just the same as CGS, 108

Problem 2:

Given the following matrix A, its inverse and a vector b:

$$A = \begin{bmatrix} 5 & -1 & 1 \\ -1 & 6 & -2 \\ 1 & -2 & 6 \end{bmatrix} \text{ with } A^{-1} = \frac{1}{152} \begin{bmatrix} 32 & 4 & -4 \\ 4 & 29 & 9 \\ -4 & 9 & 29 \end{bmatrix}; b = \begin{bmatrix} 5 \\ -1 \\ 2 \end{bmatrix}$$

(a)

Find the conditional number of A using both the ℓ_1 norm and the ℓ_∞ norms

$$\begin{aligned} \kappa(A) &= ||A||\,||A^{-1}|| \\ &= ||\begin{bmatrix} 5 & -1 & 1 \\ -1 & 6 & -2 \\ 1 & -2 & 6 \end{bmatrix}||\,||\frac{1}{152} \begin{bmatrix} 32 & 4 & -4 \\ 4 & 29 & 9 \\ -4 & 9 & 29 \end{bmatrix}|| \end{aligned}$$

if using the ℓ_1 norm, or the maximum absolute column sum

then $\kappa(A) = 9 * \frac{1}{152} 42 = \frac{210}{152} = \frac{189}{76}$

if using the ℓ_∞, or the maximum absolute row sum

then $\kappa(A) = 9 * \frac{1}{152} 42 = \frac{189}{76}$

(b)

Solve the equation $Ax = b$ using an analytical method

since in edstem *Any non-iterative method can be considered analytical.*

$$\begin{aligned} x &= A^{-1}b \\ &= \frac{1}{152}\begin{bmatrix} 32 & 4 & -4 \\ 4 & 29 & 9 \\ -4 & 9 & 29 \end{bmatrix}\begin{bmatrix} 5 \\ -1 \\ 2 \end{bmatrix} \\ &= \frac{1}{152}\begin{bmatrix} 148 \\ 9 \\ 29 \end{bmatrix} \end{aligned}$$

Problem3

In lecture, we examined the stability of multiplying vectors with vectors and triangular matrices. We now consider the problem of computing the general matrix product $Ax = b$, where $A \in \mathbb{C}^{n\times n}$ and $x, b \in \mathbb{C}^n$. We use the following simple algorithm to compute b for given A, x

Algorithm Matrix-Vector Multiplication

```
1: b = 0
2: for i = 1to n do
3:     for j = 1to n do
4:         b_i = b_i + A_ij x_j
5:     end for
6: end for
```

Perform a backward stability analysis of this algorithm. You may assume the given data are floating-point numbers (errors are only introduce by arithmetic operations)

we assume the floating point precise of the machine that this algorithm will run on is $\epsilon_{\text{machine}}$, and the floating point operations to be $\oplus, \otimes$ for add ans multiply, and fl() for convert the numerical values to the real value used in the program

according to the question

- input data can be seen as x
- output is Ax

in order to check an algorithm is backwards stable, we need to check

- $\tilde{f}(x) = f(\tilde{x})$
- $\frac{||\tilde{x}-x||}{||x||} = \mathcal{O}(\epsilon_{\text{machine}})$

for clarity

we first analyse the inner cycle, or line 3 to 5,

then the inner loop will iteratively increment b_i by $A_{ij}x_j$

therefore, the inner loop does $b_i = b_i \oplus (A_{ij} \otimes x_j)$ iteratively

Since

$$\begin{aligned} fl(A_{ij}) \otimes fl(x_j) &= A_{ij} \otimes x_j \\ &= (1+\delta_1)A_{ij}x_j \end{aligned}$$

where, as the question states, we assume everything is floating point, or fl is the identity function. δ_1 is the error caused during multiplying floating point numbers and $|\delta| \leq \epsilon_{\text{machine}}$

therefore we can show that, the inner cycle will turn b_i to

$b_i + \sum_{j=1}^{n}(1+\delta_1)A_{ij}x_j$

combined with the outer loop, which does the same operation for each i, independently

the program turns b (initially 0), to $(1+\delta_1)Ax$

therefore $\tilde{f}(x) = (1+\delta_1)(1+\delta_2)Ax$

where δ_2 is the error caused by $\oplus$

to satisfy $f(\tilde{x})$ which $f(x) = Ax$, we let $\tilde{x} = (1+\delta_1)(1+\delta_2)x$

then

$$\begin{aligned} \frac{||\tilde{x} - x||}{||x||} &= \frac{||(1+\delta_1)(1+\delta_2)x - x||}{||x||} \\ &= \frac{||(\delta_1 + \delta_2 + \delta_1\delta_2)x||}{||x||} \\ &= \mathcal{O}(\epsilon_{\text{machine}}) \end{aligned}$$

therefore, we can safely conclude that this algorithm is backward stable

Problem 4

We defined the growth factor for Gaussian elimination of a matrix $A \in \mathcal{L}^{m\times m}$ to be

$$\rho = \frac{\max_{i,j}|u_{i,j}|}{\max_{i,j}|a_{i,j}|}$$

where $A = LU$, $a_{i,j}$ is the element $A[i,j]$ and $u_{i,j}$ is the element $U[i,j]$. Show that with partial pivoting this growth factor is bounded by $\rho \le 2^{m-1}$

we use induction to prove this

base case:

m = 1, A is an $1 * 1$ matrix say $[a]$

then we dont need elimination for solving $A = LU$

A = U, or $\rho = 1 = 2^0 = 2^{1-1} = 2^{m-1}$

therefore when m = 1, the growth factor is bounded by 2^{m-1}

Induction hypothesis

$\forall N \in \mathbb{R}^{(m-1)\times(m-1)}, \rho_{(m-1)} \le 2^{(m-1)-1} = 2^{m-2}$

Inductive Step:

take arbitrary matrix $A \in \mathbb{C}^{m\times m}$

During the first step of Gaussian elimination, we look for the largest absolute value in the first column of the matrix to serve as the pivot.

Let $M = \max_{1\le i\le m} |a_{i1}|$ be the largest absolute value in the first column

Assume we a swap the first row containing M (the pivot row).

then we do the following operation on the remaining rows in Gaussian Elimination

$u_{ij} := u_{ij} - \frac{a_{i1}}{a_{11}} u_{ij}$

(reminder: we assume the pivot is in the first row for clarity)

since a_{11} is the pivot, this limits $|\frac{a_{i1}}{a_{11}}| \le 1$ since $a_{11} \ge a_{i1} \forall i$

therefore, after performing the elimination, the maximum absolute value of the entries in U can at most be doubled

this can happen when $\frac{a_{i1}}{a_{11}} = -1$

and since, by inductive hypothesis, $\rho_{m-1} \le 2^{m-2}$

therefore $\rho_m \le 2 * 2^{m-2} = 2^{m-1}$

So in conclusion, we have shown that $\rho \leq 2^{m-1}$ if $A \in \mathbb{C}^{m\times m}$