Question 2

a) $x, y \in \mathbb{R}^m$

$x^Tx = y^Ty$, $v = \dfrac{y-x}{\|y-x\|}$, $H = I - 2vv^T$

Show that $Hx = (I - 2vv^T)x = y$

soln

$$Hx = (I - 2vv^T)x$$

$$= x - 2vv^Tx = x - \frac{2(y-x)(y-x)^Tx}{\|y-x\|^2}$$

$$= x - \frac{2(y-x)(y-x)^T}{\|y-x\|^2} + (y-y)$$

$$= \frac{-(y-x)}{\|y-x\|^2}\left[\|y-x\|^2 + 2(y-x)^Tx\right] + y$$

$$= \frac{-(y-x)}{\|y-x\|^2}\left[(y-x)^T(y-x) + 2(y-x)^Tx\right] + y$$

$$= \frac{-(y-x)}{\|y-x\|^2}\left[y^Ty - y^Tx - x^Ty + x^Tx + 2y^Tx - 2x^Tx\right] + y$$

\# Since $x^Tx = y^Ty$, then $y^Tx = x^Ty$

$$Hx = \frac{-(y-x)}{\|y-x\|^2}\left[2x^Tx - 2x^Tx + 2y^Tx - 2y^Tx\right] + y$$

$$= \frac{-(y-x)}{\|y-x\|^2}\cdot[0] + y$$

$$Hx = y$$

$H = I - 2vv^T$

$Tr(H) = Tr(I) - 2Tr(vv^T)$

for an identity matrix in $\mathbb{R}^{m \times m}$, $Tr(I) = m$

$$vv^T = \begin{bmatrix} v_1^2 & v_1v_2 & \cdots & v_1v_m \\ v_1v_2 & v_2^2 & & \\ \vdots & & \ddots & v_m^2 \\ v_1v_m & \cdots & & \end{bmatrix}$$

Since v is a unit vector,

$Tr(vv^T)$ is simply the inner product v^Tv.

$Tr(vv^T) = 1$

Therefore

$Tr(H) = m - 2 \times 1 = m - 2$ — ①

We know that H is both symmetric and orthogonal so the ~~magnitude~~ absolute value of its eigenvalues is 1 (i.e. $\|\lambda\| = 1$)

Also, $Tr(H) = \sum_{i=1}^{m} \lambda_i$, $i = 1, 2, \ldots, m$ — ②

where λ_i are the eigenvalues of H

Equating ① and ②

$\lambda_1 + \lambda_2 + \cdots + \lambda_m = m - 2$

Let $\lambda_1 = \lambda_2 = \lambda_3 = \cdots = \lambda_{m-1} = \lambda$

$(m-1)\lambda + \lambda_m = m - 2$

$m\lambda - \lambda + \lambda_m = m - 2$

Equating co-efficients

$m\lambda = m \Rightarrow \lambda = 1$

$\lambda_m - \lambda = -2$

but $\lambda = 1$

$\lambda_m = -2 + 1 = -1$

Thus $\lambda = 1$ with multiplicity of $m - 1$ and $\lambda = -1$ with multiplicity of 1.

2b

(ii)

$$H = I - 2vv^T$$

$$Hv = (I - 2vv^T)v$$

$$= v - 2v(v^Tv)$$

But $v^Tv = 1$

$$Hv = \cancel{v} - 2v$$

$$= -v$$

$$H = I - 2vv^T$$

$$Hu = u - 2vv^Tu$$

Since u is orthogonal to v (i.e $v^Tu = 0$) — ⊛

$$Hu = u$$

From *, $v \perp u$. Since the dim span$(v) = 1$, dim span$(v)^\perp = m-1$

Thus, $\lambda = 1$ is an eigen value with multiplicity of $m-1$.

Additionally, $Hv = -v$ shows that the remaining eigenvalue is -1.

2biii

The determinant of a matrix is the same as the product of its eigenvalues, that is;

$$\det(H) = \lambda_1 \times \lambda_2 \times \lambda_3 \times \cdots \times \lambda_m$$

where $\lambda_i, i = 1, 2, \ldots, m$ are the eigenvalues of H

Therefore, $\det(H) = -1(1)^{m-1} = -1$

QUESTION 1

```
clc;clear all;
%%%%%% Set up Vandermonde matrix %%%%%%
m=100;
n=15;
A=zeros(m,n);
A(:,1)=ones(m,1);
for j=1:m
    tj=(j-1)/(m-1);
    for i=2:n
        A(j,i)=tj^(i-1);
    end
end
```

1(a)

```
function [Q,R]=GS(A)
[m, n]=size(A);
assert(m>=n,'The row count must be greater than or equal to the column count')
Q=zeros(m,n);
R=zeros(n,n);
for j=1:n
    aj=A(:,j);
    vj=aj;
    for i=1:j-1
        qi=Q(:,i);
        R(i,j)=qi'*aj;
        vj=vj-R(i,j)*qi;
    end
    R(j,j)=norm(vj);
    Q(:,j)=vj/R(j,j);
end

end
```

1(b)

```
function [Q,R]=MGS(A)
[m, n]=size(A);
assert(m>=n,'The row count must be greater than or equal to the column count')
Q=zeros(m,n);
R=zeros(n,n);
for i=1:n
    vi=A(:,i);
    R(i,i)=norm(vi);
    Q(:,i)=vi/R(i,i);
```

```
        for j=i+1:n
            qi=Q(:,i);
            R(i,j)=qi'*A(:,j);
            A(:,j)=A(:,j)-R(i,j)*qi;

        end
    end
    end
```

1(C)

```
[Q,R]=GS(A);
[q1,r1]=MGS(A);
```

```
norm(A-Q*R,'inf')
```

```
ans = 1.1657e-15
```

```
norm(A-q1*r1,'inf')
```

```
ans = 7.7716e-16
```

```
norm(eye(n)-Q'*Q)  % substantial accumulation of roundoff errors
```

```
ans = 4.9871
```

```
norm(eye(n)-q1'*q1) % minimal accumulation of roundoff errors
```

```
ans = 2.1379e-07
```

COMMENT

Each vector in the classical Gram-Schmidt (CGS) method is taken one at a time and made orthogonal to all previous vectors. However, in modified Gram-Schmidt (MGS), each vector is adjusted to be orthogonal to all subsequent vectors. As a result, the CGS is more susceptible to numerical roundoff errors, thus leading to a loss of orthogonality.

Question 3

(a)

```
function [V,R]=house(A)
[m, n]=size(A);
assert(m>=n,'The row count must be greater than or equal to the column count')
R=A;
V=zeros(m,n);
for k=1:n
    x=R(k:end,k);
    I=eye(m-(k-1));
    vk=x;
    vk(1)=vk(1)+sign(x(1))*norm(x);
    vk=vk/norm(vk);
    V(k:end,k)=vk;
    P=I-2*(vk*vk');
    R(k:end,k:n)=P*R(k:end,k:n);
end
end
```

(b)

```
function [Q]=house2q(V)
[m, n]=size(V);
assert(m>=n,'The row count must be greater than or equal to the column count')
Q=eye(m);
for k=n:-1:1
    Q(k:m,:) = Q(k:m,:) - 2*V(k:m,k)*(V(k:m,k)'*Q(k:m,:));
end
end
```

(c)

```
b=[1 2 3;4 5 6; 7 8 7; 4 2 3; 4 2 2];

[V,R]=house(b);
```

```
[Q]=house2q(V);

[Q_m,R_m]=qr(b);  % MATLAB qr routine
```

```
error_q = norm(Q_m-Q) % norm of difference
```

```
error_q = 7.7441e-16
```

```
error_r=norm(R_m-R) % norm of difference
```

```
error_r = 1.7014e-15
```

```
Q*R
```

```
ans = 5×3
    1.0000    2.0000    3.0000
    4.0000    5.0000    6.0000
    7.0000    8.0000    7.0000
    4.0000    2.0000    3.0000
    4.0000    2.0000    2.0000
```

```
function [V,R]=house(A)
[m, n]=size(A);
assert(m>=n,'The row count must be greater than or equal to the column count')
R=A;
V=zeros(m,n);
for k=1:n
    x=R(k:end,k);
    I=eye(m-(k-1));
    vk=x;
    vk(1)=vk(1)+sign(x(1))*norm(x);
    vk=vk/norm(vk);
    V(k:end,k)=vk;
    P=I-2*(vk*vk');
    R(k:end,k:n)=P*R(k:end,k:n);
end
end


function [Q]=house2q(V)
[m, n]=size(V);
assert(m>=n,'The row count must be greater than or equal to the column count')
Q=eye(m);
for k=n:-1:1
    Q(k:m,:) = Q(k:m,:) - 2*V(k:m,k)*(V(k:m,k)'*Q(k:m,:));
end
end




function [Q,R]=MGS(A)
[m, n]=size(A);
```

```
assert(m>=n,'The row count must be greater than or equal to the column count')
Q=zeros(m,n);
R=zeros(n,n);
for i=1:n
    vi=A(:,i);
    R(i,i)=norm(vi);
    Q(:,i)=vi/R(i,i);
    for j=i+1:n
        qi=Q(:,i);
        R(i,j)=qi'*A(:,j);
        A(:,j)=A(:,j)-R(i,j)*qi;

    end
end
end

function [Q,R]=GS(A)
[m, n]=size(A);
assert(m>=n,'The row count must be greater than or equal to the column count')
Q=zeros(m,n);
R=zeros(n,n);
for j=1:n
    aj=A(:,j);
    vj=aj;
    for i=1:j-1
        qi=Q(:,i);
        R(i,j)=qi'*aj;
        vj=vj-R(i,j)*qi;
    end
    R(j,j)=norm(vj);
    Q(:,j)=vj/R(j,j);
end
end
```