Exercise 1 (4 points)

(a) Implement the CG method (Algorithm 7.2) in MATLAB as a function

\[
\text{function } X = \text{cg}(A,b,\text{tol})
\]

where \( A \) and \( b \) are the matrix \( A \in \mathbb{C}^{m \times m} \) and the vector \( b \in \mathbb{C}^m \) of the linear system \( Ax = b \). The number \( \text{tol} \) specifies the tolerance of the method. The return value \( X \) is to be a matrix containing all iterates (including \( x_0 \)) of CG as columns, i.e., \( X(:,k) \) is to be the CG iterate \( x_{k-1} \). For the initial iterate use \( x_0 = 0 \). The “for \( k = 0,1,\ldots \)” loop should run at most \( m \) times, even if no convergence was reached. After computing a residual \( r_k \) your code has to check for \( \|r_k\|/\|r_0\| < \text{tol} \). If this condition is met the algorithm is finished. The algorithm should also finish if \( \|r_0\| < \text{tol} \), i.e., if the initial residual is less than \( \text{tol} \).

(b) Let \( A \) be the tridiagonal matrix

\[
A = \begin{pmatrix}
2 & -1 & & & \\
-1 & \ddots & \ddots & & \\
& \ddots & \ddots & -1 & \\
& & -1 & 2 & \\
\end{pmatrix} \in \mathbb{C}^{n \times n}.
\]

Plot the convergence of the CG method. For the right hand side use \( b = (1,\ldots,1)^T \). Test \( n = 10, 100, 1000 \) and \( \text{tol} = 10^{-6} \). Plot the iteration number on the \( x \)-axis and the \( A \)-norm of the error \( (\|x_k - x^*\|_A) \) on the \( y \)-axis. The exact solution \( x^* \) can be computed in MATLAB as

\[
x_{\text{star}} = [1:n] .* [n:-1:1] / 2;
\]

Compare these plots of the \( A \)-norms of the error with your plots of the relative residual norms from Exercise 2 on Exercise Sheet 5. What do you observe?

Note: For this particular example CG finds the exact solution after \( \lceil n/2 \rceil \) iterations, i.e., \( x_{\lceil n/2 \rceil} = x^* \).

Hint: You can use the function \( \text{cg2} \) from the MATLAB file \( \text{cg2.m} \) to compute the CG iterates. The function has the same input and output arguments as described for the \( \text{cg} \) function in part (a).

– Continues on next page –
Exercise 2 (4 points)
Suppose the $m \times n$ matrix $A$ with $m > n$ has the form

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix},$$

where $A_1 \in \mathbb{C}^{n \times n}$ is a nonsingular matrix and $A_2 \in \mathbb{C}^{(m-n) \times n}$ is an arbitrary matrix. Prove that $\|A^+\|_2 \leq \|A_1^{-1}\|_2$.

Exercise 3 (4 points)
(a) Write a MATLAB function $[W,R] = \text{housei}(A)$ that computes an implicit representation of a full QR factorization $A = QR$ of an $m \times n$ matrix $A$ with $m \geq n$ using Householder reflections. The output variables are a lower-triangular matrix $W \in \mathbb{C}^{m \times n}$ whose columns are the vectors $v_k$ defining the successive Householder reflections, and a triangular matrix $R \in \mathbb{C}^{n \times n}$.

(b) Write a MATLAB function $Q = \text{form}_q(W)$ that takes the matrix $W \in \mathbb{C}^{m \times n}$ produced by $\text{housei}$ as input and generates a corresponding $m \times m$ orthogonal matrix $Q$.

Exercise 4 (2 points)
(a) Write a MATLAB function $[Q,R] = \text{house}_qr(A)$ that computes the full QR factorization $A = QR$. Use the functions $\text{housei}$ and $\text{form}_q$ from Exercise 3.

(b) Run the following MATLAB code:

```matlab
n = 80;
[U,X] = qr(randn(n));
[V,X] = qr(randn(n));
S = diag(3.^(-1:-10:-10*n));
A = U*S*V;

[Qc, Rc] = clgs(A);
[Qm, Rm] = mgs(A);
[Qh, Rh] = house_qr(A);
I = eye(n);

norm_c = norm(Qc'*Qc-I, 'fro')
norm_m = norm(Qm'*Qm-I, 'fro')
norm_h = norm(Qh'*Qh-I, 'fro')
```

Hand in the output you get when you run this code.
For the $\text{clgs}$ and $\text{mgs}$ functions use your implementations of the classical and modified Gram-Schmidt algorithms; see Exercise 4 of Exercise Sheet 4.