F.M. Faulstich **Math 6800: Homework assignment 2** **Due:** 09/19

Exercise 1 *Consider the matrix*

$$A = \begin{bmatrix} 16 & 4 & 4 & -4 \\ 4 & 10 & 4 & 2 \\ 4 & 4 & 6 & -2 \\ -4 & 2 & -2 & 4 \end{bmatrix}$$

Compute its Cholesky decomposition by hand.

Exercise 2 *Let $x \in \mathbb{K}^m$. Prove that there exists a Householder transformation P such that*

$$Px = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

where $\alpha = ||\mathbf{x}||$.

Exercise 3 *Consider the Cauchy/Laplace kernel, i.e.,*

$$D_{pq} = \frac{1}{\omega_p + \omega_q}, \qquad \omega_p > 0, \tag{1}$$

Prove that this is symmetric and positive definite.
HINT*: Use Laplace transform*

Programming Assignment

To receive full credit for the following programming exercises, please ensure that your code correctly handles any relevant input.

Exercise 4 *(Pivoted Cholesky Algorithm) Write a function pivoted_cholesky. Follow the pseudo-code given in class. The function takes A and a tolerance ϵ as input and returns L such that $L^*L = A$.*

Exercise 5 *(Matrix-free Pivoted Cholesky) In special cases, we are able to improve on the algorithms by making problem-specific adjustments.*
Consider the matrix D in Eq. (1), which is defined via a provided vector $\omega = [\omega_p]_{p=1}^K$. We seek to compute a low-rank approximation of D without ever holding D in memory, i.e., "matrix-free"! To that end, write a function MFPC(ω, max_rank=None, tol=None) where ω is the input vector defining D, max_rank and tol are parameters that can be chosen by the user, terminating the pivoted Cholesky algorithm either after max_rank or tol is reached.
HINT*: Write two helper functions e.g. diag_fn and col_fn that return diagonal of the Cauchy kernel, and the k^{th} column of the Cauchy kernel, respectively.*

Exercise 6 *In the following exercises, you will implement a memory-improved version of Householder QR and use it to solve a linear system $Ax = b$.*

1. *Write a function backward_substitution that takes a matrix U (upper triangular) and a vector b as input that returns the solution obtained via backward substitution.*

2. *Write a function householder_qr_opt that is storage improved, i.e., you are supposed to perform all computations in place. You may only define additional variables of the types integer and one additional vector.*

3. *Write a function apply_Q_opt that computes*

$$y = Q'b,$$

   provided the output from householder_qr_opt. Again, perform all computations in place, no additional variables are to be defined.

4. *Write a linear solver (named solve_LS) that only uses the above routines householder_qr_opt, apply_Q_opt, and backward_substitution. The function takes a matrix A and vector b as input, and returns the solution to the system $Ax = b$.*