1. Let $A$ and $B$ be square matrices of order $n$. Six different ways to compute $C = A \ast B$ are obtained by considering the different orderings of a triply-nested “for-loop”, each with the single executable statement

$$C(i, j) = C(i, j) + A(i, k) \ast B(k, j).$$

What is this triply-nested for-loop? Which ordering(s) of the for-loops correspond to a row-by-row computation of $C$? Which ordering(s) correspond to a column-by-column computation of $C$? How can you describe the computation of $C$ given by the remaining orderings? How many flops are required by the different algorithms?

2. Let $x$ and $y$ be 2 $n$-vectors and define $A$ to be the matrix such that $A(i, j) = x(i) \ast y(j)$, $i, j = 1 : n$. Given that the vectors $x$ and $y$ are on hand, show that it is possible to compute the vector $w = Av$ in $3n$ flops where $v$ is a given $n$-vector.

3. Let $T$ be a triangular matrix of order $n$. Show that if $T(i, i) \neq 0$, $i = 1 : n$ then $T$ has rank $n$. Show that if $T(j, j) = 0$, for some $j$, then $\text{rank}(T) < n$.

4. Let $A$ be a square matrix of order $n$. Give a constructive argument to show that there exist a permutation matrix $P$, a unit lower-triangular matrix $L$, and an upper-triangular matrix $U$ such that

$$PA = LU.$$

Is it true that $A$ is nonsingular if and only if $U$ is nonsingular? Explain.

5. A sparse matrix is a matrix with many zeros. Often a sparse matrix is represented compactly. For example, the nonzeros of a sparse matrix may be stored as a collection of 3-tuples, $(i, j, \text{value}_{ij})$, where $A(i, j) = \text{value}_{ij} \neq 0$. The zeros are not stored; the number of 3-tuples is equal to the number of nonzeros in the matrix.
Describe an algorithm for computing $A \times x$, where $A$ is a sparse matrix represented using the 3-tuple data structure and $x$ is a given (dense) $n$-vector. What is the complexity of computing $A \times x$ (i.e., the number of flops) in terms of $n, r_i, c_j$ where $r_i$ is the number of nonzeros in row $i$, $i = 1 : n$, and $c_j$ is the number of nonzeros in column $j$, $j = 1 : n$.

6. (a) Let $H$ be a real symmetric matrix of order $n$ and let $A$ be $m$-by-$n$ with $\text{rank}(A) = m$. Assume $m \ll n$. Define

$$M = \begin{bmatrix} H & A^T \\ A & W \end{bmatrix}.$$ 

Assume $W$ is real and symmetric and $M$ is positive definite. If $H = LL^T$ is precomputed, i.e., on tap, how can a system $Mx = b$ be solved in $O(n^2)$ flops? Matrix $L$ is lower triangular and nonsingular.

(b) Experiment with your technique (in MATLAB) on different sized systems. Compare flop counts, and execution times, to the counts/times obtained using $x = M \backslash b$. (Compare solutions to verify that your technique is valid.) In addition, plot observed execution time for your technique versus $n^2$. Observations? Explain any results that appear to be inconsistent with the theory.

7. Suppose we wish to compute the solution to the square (nonsingular) linear system, $(H + A^T W^{-1} A)z = r$, where $H$ is symmetric ($n$-by-$n$), $W$ is symmetric, nonsingular, and $A$ is $m$-by-$n$ with $\text{rank}(A) = m$. Propose a direct method to solve for $z$ that does not require the explicit formulation of $(H + A^T W^{-1} A)$, $A^T W^{-1} A$, or even $W^{-1}$. Suggest why your method might be advantageous.