CSCI 5646 **Problem Set 5**

This assignment is due by 11:55pm on Saturday, October 1st. Minimal credit will be given for incomplete solutions or solutions that do not provide details on how the solution is found. You are encouraged to discuss the problems with your classmates, but all work (analysis and code) must be your own.

1. In class we derived the column oriented version of forward substitution by partitioning the lower triangular system $G\mathbf{x} = \mathbf{b}$. Different forms of the partition lead to different versions of forward substitution. For example, consider the following partition

$$\begin{bmatrix} \hat{G} & 0 \\ \hat{\mathbf{h}}^T & g_{nn} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}} \\ x_n \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{b}} \\ b_n \end{bmatrix}$$

   where $\hat{G}$ is $(n-1) \times (n-1)$ lower triangular.

   (a) Derive a recursive version of forward substitution based on this partition.

   (b) Write pseudocode for a non-recursive version of the algorithm. (**Hint**: Think about how your recursive algorithm would compute x_i given $x_1, x_2, \ldots, x_{i-1}$).

   (c) Is the algorithm you developed more suited for a language which uses column-oriented or row-oriented storage of G? Justify your response.

2. (a) Consider using forward substitution to solve the $n \times n$ lower triangular system $G\mathbf{x} = \mathbf{b}$ where the first k leading entries of $\mathbf{b}$ are zero. Come up with a simple modification of the forward substitution algorithm that dramatically reduces the computational cost of the solve. What is the leading order term in the cost of your method in terms of n and k?

   (b) Let A be $n \times n$ and nonsingular and suppose that (for whatever reason) you need to compute A^{-1}. If you have an LU factorization of A then you can compute A^{-1} by computing each of its columns individually. We'll show this in detail in class, but you should be able to convince yourself that the k^{th} column of A^{-1} is the solution to the linear system $A\mathbf{x}_k = LU\mathbf{x}_k = \mathbf{e}_k$ where $\mathbf{e}_k$ is the k^{th} canonical basis vector. Thus, naively, each column of A^{-1} can be computed via one forward and one backward substitution (at a cost of roughly $2n^2$ flops), for a total cost (including the LU factorization) of $8n^3/3$ flops. The cost of the computation can be significantly reduced if you leverage the fact that $\mathbf{e}_k$ has $k-1$ leading zeros. Use your strategy from part (a) to show that A^{-1} can be computed in roughly $2n^3$ flops (making the inverse cheaper but still totally not worth ever computing).