# MAT 226B, Winter 2020

# Homework 1

(due by Thursday, January 30, 11:59 pm)

---

**General Instructions**

- You are required to submit each of your assignments via file upload to Canvas. Note that the due dates set in Canvas are hard deadlines. I will not accept any submissions outside of Canvas or after the deadline. For each assignment, upload two files: a single pdf file of your assignment and a single zip file that contains all your Matlab files. To prepare the zip file, create a directory that contains all your Matlab files and then zip that directory. For each of the subproblems that require Matlab computations include a single driver file so that I can run and check your program.
- If at all possible, use a text processing tool (such as LaTeX) for the preparation of your assignments. If you submit scanned-in hand-written assignments, make sure that you write clearly and that you present your solutions in a well-organized fashion. If I cannot read your assignment, I will not be able to grade it!
- You are required to solve the problems on the homework sets and on the final project yourself! If there are students with solutions or program codes that were obviously copied or are trivial modifications of each other, then each involved student (regardless of who copied from whom) will only get the fraction of the points corresponding to the number of involved students.
- Test cases for computational problems are often provided as binary Matlab files. For example, suppose the file "`LS.mat`" contains the coefficient matrix $A$ and the right-hand side $b$ of a system of linear equations. The Matlab command "`load('LS.mat')`" will load $A$ and $b$ into Matlab.
- When you are asked to print out numerical results, print real numbers in 15-digit floating-point format. You can use the Matlab command "`format long e`" to switch to that format from Matlab's default format. For example, the number $10\pi$ would be printed out as `3.141592653589793e+01` in 15-digit floating-point format.

---

1. Let $A = \left[a_{jk}\right] \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix, and let $L = \left[l_{jk}\right]$ denote the Cholesky factor of $A$.

   (a) Suppose the matrix $A$ is dense. Determine the exact number of flops required to compute $L$, using the Cholesky algorithm presented in class. (Each addition, subtraction, multiplication, division, or square root counts as one flop.)

(b) Suppose the matrix $A$ is banded with bandwidth $2p+1$, *i.e.*, $a_{jk} = 0$ if $|j-k| > p$. Show that the Cholesky factor $L$ has lower bandwidth $p$, *i.e.*, $l_{jk} = 0$ if $j - k > p$.

(c) For banded matrices $A$, formulate a variant of the Cholesky algorithm presented in class that fully exploits the bandedness of $A$.

(d) Determine the exact number of flops required when your algorithm from part (c) is applied to a banded matrix $A$ with bandwidth $2p + 1$.

2. Let $m \geq 2$ be an integer and set $n = m^2$. Consider the matrices

$$T_m = \begin{bmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{bmatrix} \in \mathbb{R}^{m \times m}$$

and

$$T_{m\times m} = \begin{bmatrix} T_m + 2I & -I & 0 & \cdots & 0 \\ -I & T_m + 2I & -I & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -I & T_m + 2I & -I \\ 0 & \cdots & 0 & -I & T_m + 2I \end{bmatrix} \in \mathbb{R}^{n \times n},$$

where $I$ denotes the $m \times m$ identity matrix. The matrices $T_m$ and $T_{m\times m}$ are symmetric positive definite, and thus the Cholesky factorization exists for each matrix.

(a) Determine explicit formulae for the entries of the Cholesky factor $L_m$ of $T_m$.

(b) Determine the sparsity structure of the Cholesky factor $L_{m\times m}$ of $T_{m\times m}$. How many fill-in elements appear in $L_{m\times m}$ ?

3. Let $1 \leq m < n \leq$ be integers and

$$\mathcal{I} = \left\{ 1 \leq i_1 < i_2 < \cdots < i_m \leq n \right\}$$

be a set of $m$ integers. Let $A = \left[a_{jk}\right] \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix such that all entries $a_{jk}$ with row index $j \in \mathcal{I}$ or column index $k \in \mathcal{I}$ are nonzero and that all other nondiagonal entries of $A$ are zero.

Determine a reordering of the rows and columns of the matrix $A$ such that the Cholesky factor $L$ of the reordered matrix has as few nonzero entries as possible. Give sharp upper and lower bounds for $\texttt{nnz}(L)$.

4. You are given an $9 \times 9$ matrix $A \succ 0$ with the following sparsity structure:

$$A = \begin{bmatrix} * & 0 & * & 0 & 0 & 0 & * & * & * \\ 0 & * & 0 & * & 0 & 0 & 0 & 0 & 0 \\ * & 0 & * & * & 0 & 0 & * & 0 & 0 \\ 0 & * & * & * & * & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & * & * & 0 & * & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & * & 0 & * & * \\ * & 0 & * & 0 & * & 0 & * & * & 0 \\ * & 0 & 0 & 0 & 0 & * & * & * & 0 \\ * & 0 & 0 & 0 & 0 & * & 0 & 0 & * \end{bmatrix}.$$

   (a) Draw the graph $G(A)$ associated with $A$.

   (b) Apply the minimum degree algorithm (as presented in class) to $G(A)$ to obtain a reordering of the rows and columns of $A$.

   (c) Show the sparsity structure of the sparse Cholesky factor $L$ of $P^T AP$, where $P$ is the permutation matrix corresponding to the minimum-degree ordering obtained in part (b).

5. The Cholesky algorithm presented in class generates the entries of the Cholesky factor of $A \succ 0$ in a columnwise fashion. For a Matlab implementation, this algorithm is thus well suited for adaptation to sparse matrices. Since there are sparse matrices $A \succ 0$ with nonsparse Cholesky factors $L$, such a Cholesky algorithm will not always run to completion. It is good practice to monitor runtime and amount of fill-in elements and to terminate the algorithm early when either one exceeds 'reasonable' limits.

   (a) Write an efficient Matlab function

   ```
   [L,flag,k] = spcholesky(A,nnzmax,tlimit)
   ```

   that implements the Cholesky algorithm presented in class for sparse matrices.

   The inputs and outputs of `spcholesky` are

   `A:` a matrix $A \succ 0$, stored in Matlab's sparse storage format,

   `nnzmax:` an integer that limits the number of nonzero entries of $L$,

   `tlimit:` a limit (in seconds) of the allowed runtime,

   `L:` the sparse lower-triangular matrix generated by the algorithm,

   `flag:` a flag that communicates how the algorithm terminated:
   `flag` $= 0$ if the algorithm ran to completion,
   `flag` $= 1$ if the limit `nnzmax` was exceeded,
   `flag` $= 2$ if the limit `tlimit` was exceeded,

   `k:` the index `k` of the outer loop of the algorithm at termination.

**Hints:** Use the Matlab function "`tril`" to efficiently initialize $L$. Use the Matlab functions "`tic`" and "`toc`" to monitor the runtime of your algorithm. Check the total elapsed runtime at the end of each $k$-th execution of the outer loop and stop if the limit `tlimit` is exceeded. Check the total number of nonzeros of $L$ at the end of each $k$-th execution of the outer loop and stop if the limit `nnzmax` is exceeded. Make sure that the inner loop (over row indices $j$) is only executed for the indices $j$ with $l_{jk} \neq 0$.

(b) Run your function "`spcholesky`" on the $16 \times 16$ matrix $A$ given in the binary Matlab file "`small_ex.mat`". Choose `nnzmax` and `tlimit` so that your function terminates with the Cholesky factor $L$ of $A$. Print out $\texttt{nnz}(L)$, the nonzero entries $l_{jk}$ of $L$ in columns $k = 4, 7, 10, 13, 16$, together with their corresponding row indices $j$, and the relative error

$$\frac{\left\|A - LL^T\right\|_2}{\|L\|_2}$$

of your computed Cholesky factorization.

(c) Matlab provides the functions "`symamd`", "`colamd`", "`symrcm`", and "`colperm`" that can be employed to generate reorderings of the rows and columns of $A$. The output of these routines is a compact representation of the permutation matrix $P$ in the sparse Cholesky factorization

$$P^T AP = LL^T. \tag{1}$$

The use of this compact representation (in the case of "`symamd`") for the computation of (1) by means of "`spcholesky`" is as follows:

```
p = symamd(A);
[L,flag,k] = spcholesky(A(p,p),nnzmax,tlimit);
```

For each of the two matrices $A$ given in the binary Matlab files "`medium_ex1.mat`" and "`medium_ex1.mat`", compute the Cholesky factor $L$ of $A$ for the following 5 cases:

(i) No reordering of the rows and columns of $A$;
(ii) Reordering with `symamd`;
(iii) Reordering with `colamd`;
(iv) Reordering with `symrcm`;
(v) Reordering with `colperm`.

For each of these 10 runs, use $\texttt{nnzmax} = 30 * \texttt{nnz}(A)$ and $\texttt{tlimit} = 200$ and print out $\texttt{nnz}(L)$, `flag`, `k`, and the 10 largest (in absolute value) entries $l_{j,1000}$ in column $k = 1000$ of $L$, together with their corresponding row indices $j$.