CSE 6643 Homework 3

Karl Hiner

1 One-upping [25 pts]

Let $\boldsymbol{A} \in \mathbb{R}^{m \times m}$ have full rank. Assume that we have already computed the QR decomposition of $\boldsymbol{A}$. For $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^m$, we call the matrix $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{u}\boldsymbol{v}^T$ a rank-1 update of $\boldsymbol{A}$.

(a) [5 pts]

Prove that if $\boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u} \neq -1$, then $\boldsymbol{B}$ is invertible.

Let $\boldsymbol{x} \in \mathbb{R}^m$ be a non-zero vector. Then,

$$\begin{aligned}
\boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u} &\neq -1 \\
1 &\neq -\boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u} \\
\boldsymbol{v}^T\boldsymbol{x} &\neq -(\boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u})(\boldsymbol{v}^T\boldsymbol{x}) \\
\boldsymbol{x} &\neq -\boldsymbol{A}^{-1}\boldsymbol{u}(\boldsymbol{v}^T\boldsymbol{x}) \\
\boldsymbol{A}\boldsymbol{x} &\neq -\boldsymbol{u}\boldsymbol{v}^T\boldsymbol{x} \\
\big(\boldsymbol{A} + \boldsymbol{u}\boldsymbol{v}^T\big)\boldsymbol{x} &\neq \boldsymbol{0}. \\
\boldsymbol{B}\boldsymbol{x} &\neq \boldsymbol{0}
\end{aligned}$$

Thus, if $\boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u} \neq -1$, there are no nontrivial solutions for $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{0}$, and so $\boldsymbol{B}$ is invertible.

(b) [10 pts]

Design an algorithm that provably solves the system of equations $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{b}$ in $O(m^2)$ operations.

$$\begin{aligned}
\boldsymbol{x} &= \boldsymbol{B}^{-1}\boldsymbol{b} && \text{(assume } \boldsymbol{B} \text{ is invertible)} \\
&= \big(\boldsymbol{A} + \boldsymbol{u}\boldsymbol{v}^T\big)^{-1}\boldsymbol{b} && \text{(since } \boldsymbol{B} = \boldsymbol{A} + \boldsymbol{u}\boldsymbol{v}^T\text{)} \\
&= \left(\boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1}\boldsymbol{u}\boldsymbol{v}^T\boldsymbol{A}^{-1}}{1 + \boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u}}\right)\boldsymbol{b} && \text{(Sherman–Morrison formula)} \\
&= \boldsymbol{A}^{-1}\boldsymbol{b} - \frac{\boldsymbol{A}^{-1}\boldsymbol{u}\boldsymbol{v}^T}{1 + \boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{u}}\boldsymbol{A}^{-1}\boldsymbol{b} && \text{(note: (1a above), } \boldsymbol{B} \text{ invertible } \implies \text{den} \neq 0\text{)} \\
&= \tilde{\boldsymbol{x}} - \frac{\tilde{\boldsymbol{u}}\boldsymbol{v}^T}{1 + \boldsymbol{v}^T\tilde{\boldsymbol{u}}}\tilde{\boldsymbol{x}} && \text{(let } \tilde{\boldsymbol{x}} \equiv \boldsymbol{A}^{-1}\boldsymbol{b}, \tilde{\boldsymbol{u}} \equiv \boldsymbol{A}^{-1}\boldsymbol{u}\text{)} \\
&= \left[\boldsymbol{I} + \left(\frac{-1}{1 + \boldsymbol{v}^T\tilde{\boldsymbol{u}}}\right)\tilde{\boldsymbol{u}}\boldsymbol{v}^T\right]\tilde{\boldsymbol{x}} && \text{(rearrange)} \\
&= \big(\boldsymbol{I} + \alpha\tilde{\boldsymbol{u}}\boldsymbol{v}^T\big)\tilde{\boldsymbol{x}} && \text{(let } \alpha \equiv \frac{-1}{1 + \boldsymbol{v}^T\tilde{\boldsymbol{u}}}\text{)}
\end{aligned}$$

Thus, we have derived an expression for $\boldsymbol{x}$ in terms of a scalar α and vectors $\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{x}}$.

Vectors $\tilde{\boldsymbol{u}}$ and $\tilde{\boldsymbol{x}}$ are both defined in terms of $\boldsymbol{A}^{-1} = \boldsymbol{R}^{-1}\boldsymbol{Q}^T$, where we assume $\boldsymbol{R}$ and $\boldsymbol{Q}$ have already been computed. Thus, both vectors can be computed using back substitution in $O(m^2)$. α can then be computed in $O(m)$, and the final expression for $\boldsymbol{x}$ involves a vector-scalar product, a vector outer product, an identity addition, and a matrix-vector multiplication, for a total of

$$O(m) + O(m^2) + O(m) + O(m^2) = O(2m) + O(m^2) = O(m^2)$$

operations.

Here is the algorithm:

1. Solve $\boldsymbol{R}\tilde{\boldsymbol{x}} = \boldsymbol{Q}^T\boldsymbol{b}$ for $\tilde{\boldsymbol{x}}$ using back substitution. ($O(m^2)$)
2. Solve $\boldsymbol{R}\tilde{\boldsymbol{u}} = \boldsymbol{Q}^T\boldsymbol{u}$ for $\tilde{\boldsymbol{u}}$ using back substitution. ($O(m^2)$)
3. Compute the scalar $\alpha = \dfrac{-1}{1 + \boldsymbol{v}^T\tilde{\boldsymbol{u}}}$. ($O(m)$)
4. Finally, compute the solution $\boldsymbol{x} = \big(\boldsymbol{I} + \big((\alpha\tilde{\boldsymbol{u}})\boldsymbol{v}^T\big)\big)\tilde{\boldsymbol{x}}$. ($O(m^2)$)

Since the highest-order term across all steps is $O(m^2)$, the total number of operations is thus $O(m^2)$.

(c) r-upping [10 pts]

Extend the algorithm from the previous exercise to the case of $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{U}\boldsymbol{V}^T$, for $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{m\times r}$ and $r \ll m$. Calculate the asymptotic complexity of the resulting algorithm.

The Sherman-Morrison formula was the key step above. Here, we will use the generalization of that formula, the Woodbury matrix identity. Given the definitions above, the Woodbury matrix identity states that if $\big(\boldsymbol{I} + \boldsymbol{V}^T\boldsymbol{A}^{-1}\boldsymbol{U}\big)$ is invertible,

$$\boldsymbol{B}^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}\big(\boldsymbol{I} + \boldsymbol{V}^T\boldsymbol{A}^{-1}\boldsymbol{U}\big)^{-1}\boldsymbol{V}^T\boldsymbol{A}^{-1}.$$

Thus, we can express the solution $\boldsymbol{x}$ as

$$\begin{aligned}
\boldsymbol{x} &= \Big(\boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}\big(\boldsymbol{I} + \boldsymbol{V}^T\boldsymbol{A}^{-1}\boldsymbol{U}\big)^{-1}\boldsymbol{V}^T\boldsymbol{A}^{-1}\Big)\boldsymbol{b} \\
&= \boldsymbol{A}^{-1}\boldsymbol{b} - \boldsymbol{A}^{-1}\boldsymbol{U}\big(\boldsymbol{I} + \boldsymbol{V}^T\boldsymbol{A}^{-1}\boldsymbol{U}\big)^{-1}\boldsymbol{V}^T\boldsymbol{A}^{-1}\boldsymbol{b} \\
&= \tilde{\boldsymbol{x}} - \tilde{\boldsymbol{U}}\big(\boldsymbol{I} + \boldsymbol{V}^T\tilde{\boldsymbol{U}}\big)^{-1}\boldsymbol{V}^T\tilde{\boldsymbol{x}} && (\text{let } \tilde{\boldsymbol{x}} \equiv \boldsymbol{A}^{-1}\boldsymbol{b}, \tilde{\boldsymbol{U}} \equiv \boldsymbol{A}^{-1}\boldsymbol{U}) \\
&= \Big[\boldsymbol{I} - \tilde{\boldsymbol{U}}\big(\boldsymbol{I} + \boldsymbol{V}^T\tilde{\boldsymbol{U}}\big)^{-1}\boldsymbol{V}^T\Big]\tilde{\boldsymbol{x}} && (\text{rearrange}) \\
&= \Big[\boldsymbol{I} - \big(\boldsymbol{I} + \tilde{\boldsymbol{U}}\boldsymbol{V}^T\big)^{-1}\tilde{\boldsymbol{U}}\boldsymbol{V}^T\Big]\tilde{\boldsymbol{x}} && (\text{"push-through identity"[1]}) \\
&= \big(\boldsymbol{I} + \boldsymbol{\alpha}\tilde{\boldsymbol{U}}\boldsymbol{V}^T\big)\tilde{\boldsymbol{x}} && (\text{let } \boldsymbol{\alpha} = -\big(\boldsymbol{I} + \tilde{\boldsymbol{U}}\boldsymbol{V}^T\big)^{-1})
\end{aligned}$$

Note that, since $\tilde{\boldsymbol{U}} \in \mathbb{R}^{m\times r}$, and $\boldsymbol{V}^T \in \mathbb{R}^{r\times m}$, we can compute the matrix product $\tilde{\boldsymbol{U}}\boldsymbol{V}^T$ using $O(m^2 r)$ operations.

TODO need to finish after step 2. https://people.clas.ufl.edu/hager/files/update-1.pdf p224 is close.

Here is the algorithm:

1. Solve $\boldsymbol{R}\tilde{\boldsymbol{x}} = \boldsymbol{Q}^T\boldsymbol{b}$ for $\tilde{\boldsymbol{x}}$ using back substitution (same as step 1 in (b)). ($O(m^2)$)

2. Compute $\tilde{\boldsymbol{U}} = \boldsymbol{A}^{-1}\boldsymbol{U}$ by solving each of the linear systems $\boldsymbol{R}\tilde{\boldsymbol{u}}_i = \boldsymbol{Q}^T\boldsymbol{u}_i$ for $\tilde{\boldsymbol{u}}_i$, where $\tilde{\boldsymbol{u}}_i$ is the ith column of $\tilde{\boldsymbol{U}}$ and $\boldsymbol{u}_i$ is the ith column of $\boldsymbol{U}$. This can be done using back substitution (as in step 2 in (b)) for each of the r columns. $(O(m^2r))$
3. Compute the scalar $\alpha = \dfrac{-1}{1+\boldsymbol{v}^T\tilde{\boldsymbol{u}}}$. $(O(m))$
4. Finally, compute the solution $\boldsymbol{x} = \big(\boldsymbol{I} + \big((\alpha\tilde{\boldsymbol{u}})\boldsymbol{v}^T\big)\big)\tilde{\boldsymbol{x}}$. $(O(m^2))$

2 You Factor [25 pts]

In class we have seen that if $\tilde{\boldsymbol{x}} \in \mathbb{R}^m$ is the solution to the system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, as computed by unpivoted LU factorization, we have

$$(\boldsymbol{A} + \boldsymbol{E})\tilde{\boldsymbol{x}} = \boldsymbol{b}. \tag{1}$$

For u the unit roundoff error and $\tilde{\boldsymbol{L}}, \tilde{\boldsymbol{U}}$ the LU factors computed in finite precision, we have

$$|\boldsymbol{E}| \leq mu\big(2|\boldsymbol{A}| + 4\big|\tilde{\boldsymbol{L}}\big|\big|\tilde{\boldsymbol{U}}\big|\big) + O(u^2). \tag{2}$$

Here, $|\cdot|$ signifies the element-wise absolute values and $\leq$ is interpreted element-wise, as well. In this problem, we investigate the conclusions from this bound in the case of row-pivoted LU factorization.

Generally useful things:

- p174: Because each pivot selection involves maximization over a column, this algorithm produces a matrix L with entries of absolute value ≤ 1 everywhere below the diagonal. This implies $\|L\| = O(1)$ in any norm. ... so the algorithm is backward stable if $\|U\| = O(\|A\|)$.

(a) [7.5 pts]

Deduce that under row-pivoted LU factorization and taking $\|\cdot\|_\infty$ to signify the vector-infinity norm, we have

$$\|\boldsymbol{E}\|_\infty \leq mu\big(2\|\boldsymbol{A}\|_\infty + 4m\big\|\tilde{\boldsymbol{U}}\big\|_\infty\big) + O(u^2). \tag{3}$$

This prompts us to investigate the growth factor $\rho := \frac{\|\boldsymbol{U}\|_\infty}{\|\boldsymbol{A}\|_\infty}$ of row-pivoted LU factorization.

(b) [5 pts]

Verify that the rows $\boldsymbol{u}_i^T, \boldsymbol{a}_i^T$ of $\boldsymbol{U}, \boldsymbol{A}$ satisfy

$$\boldsymbol{u}_i^T = \boldsymbol{a}_i^T - \sum_{j=1}^{i-1} \boldsymbol{L}_{ij}\boldsymbol{u}_j^T. \tag{4}$$

Given the definition of row-pivoted LU factorization, we can write the system of linear equations as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \implies \boldsymbol{P}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{P}\boldsymbol{b} \implies \boldsymbol{L}\boldsymbol{U}\boldsymbol{x} = \boldsymbol{b}$ Since $\boldsymbol{L}$ is a lower triangular matrix, $\boldsymbol{U}$ is an upper triangular matrix, with the diagonal elements equal to 1. So, we can write the i-th row of $\boldsymbol{U}$ as $\boldsymbol{u}_i^T = \boldsymbol{a}_i^T - \sum\limits_{j=1}^{i-1} \boldsymbol{L}_{ij}\boldsymbol{u}_j^T$ which satisfies the equation (3).

(c) [5 pts]

Use part (b) to show that $\big\|\tilde{\boldsymbol{U}}\big\|_\infty \leq 2^{m-1}\|\boldsymbol{A}\|_\infty$.

This is the same as asking to prove that $\max \rho = 2^{m-1}$ for an mxm matrix $\boldsymbol{A}$ (Exercise 22.1 in the book). Note that (d) below is an example of this worst case. Useful here: p174: Because

each pivot selection involves maximization over a column, this algorithm produces a matrix L with entries of absolute value ≤ 1 everywhere below the diagonal. Probably want in our proof that there are no pivots in this worst case (as in the example below).

From canvas: We can first bound each element of u_1^T using (b), and then bound each element of u_2^T using (b) and the bound for each element of u_1^T, and then bound each element of u_3^T using (b) and the bounds for u_1^T, u_2^T, and so on...

(d) [7.5 pts]

Consider matrices of the form

$$\boldsymbol{A} = \begin{pmatrix} 1 & & & & 1 \\ -1 & 1 & & & 1 \\ -1 & -1 & 1 & & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{pmatrix}. \tag{5}$$

Derive the growth factor in this case as a function of m.

The $\boldsymbol{PA} = \boldsymbol{LU}$ factorization for $\boldsymbol{A}$ is:

$$\begin{pmatrix} 1 & & & & 1 \\ -1 & 1 & & & 1 \\ -1 & -1 & 1 & & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & & & & \\ -1 & 1 & & & \\ -1 & -1 & 1 & & \\ -1 & -1 & -1 & 1 & \\ -1 & -1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & & & & 1 \\ & 1 & & & 2 \\ & & 1 & & 4 \\ & & & 1 & 8 \\ & & & & 16 \end{pmatrix} \tag{6}$$

The growth factor is then

$$\rho = \frac{\|\boldsymbol{U}\|_\infty}{\|\boldsymbol{A}\|_\infty} = \frac{16}{1} = 16 = 2^4 = 2^{m-1}.$$

How does this relate to part (c)?

3 [25 pts]

Suppose that $\boldsymbol{A} \in \mathbb{K}^{m \times m}$ is strictly column diagonally dominant, meaning that for all $1 \leq k \leq m$,

$$|\boldsymbol{A}_{kk}| > \sum_{j \neq k} |\boldsymbol{A}_{jk}|. \tag{7}$$

Show that if LU factorization with row pivoting is applied to $\boldsymbol{A}$, no row interchange takes place.

To show that no row interchange occurs, we'll show that $P = I$.

Assume $P \neq I$. Then either $P_{ii} \neq 1$, or $P_{kj} \neq 0, k \neq j$.

- Assume $P_{ii} \neq 1$, then the pivot for row i is not A_{ii}. This contradicts A being strictly column diagonally dominant.
- assume $\exists k \neq j$ such that $P_{kj} \neq 0$. This implies that during some step k, there was a $j \neq k$ s.t. $|A_{kj}| < |A_{jj}|$. Thus, A_{jj} cannot be chosen as the pivot element in column j, which again contradicts the fact that A is strictly column diagonally dominant.

Therefore, we must have $P = I$, and no row interchange takes place during the factorization.

$$
\begin{aligned}
|A_{kk}| &> \sum_{j \neq k} |A_{jk}| \\
|A_{kk}| &\geq |A_{kj}| \\
|A_{kj}| &\geq |A_{ij}| \\
|A_{ij}| &\geq |A_{jj}|
\end{aligned}
$$

Combining these inequalities, we get $|A_{kk}| > \sum_{j \neq k} |A_{jk}| + |A_{jj}|$, which contradicts the assumption that A is strictly column diagonally dominant. Therefore, we must have $P = I$, and no row interchange takes place during the factorization.

4 Pivoting [25 pts]

(a) [5 pts]

Go to section (a) of the file `HW3_your_code.jl` and implement a function that takes in a matrix `LU` $\in \mathbb{K}^{m \times m}$ containing the upper triangular part of $\boldsymbol{U}$ as well as the strict lower triangular part of $\boldsymbol{L}$, as well as an array `P` $\in \{1, \dots m\}^m$ that encodes the permutation matrix $\boldsymbol{P}$ by `P`$[j] = i \Leftrightarrow \boldsymbol{P_{ij}} = 1$. Your function should not allocate any memory.

(b) [5 pts]

Go to section (b) of the file `HW3_your_code.jl` and implement the unpivoted LU factorization. Check your code by ensuring that the assertions in section a + b of `HW3_driver.jl` do not produce any errors.

(c) [5 pts]

Generate families of random $m \times m$ matrices and vectors of length m. Plot as a function of the size m, the relative error of the solution obtained from your code in parts (a,b) and the growth factor introduced in problem 2. Report the floating point type used by your program. You can use the code provided in the second homework as a starting point for creating and saving plots.

(d) [5 pts]

Go to section (d) of the file `HW3_your_code.jl` and implement the unpivoted LU factorization. Your code should pass the assertions in section (c) of `HW3_driver.jl`. Your function should take the matrix A as an input to modify in place, and return an integer array P according to the specifications of (a). Repeat the experiment of (c) using the pivoted LU factorization.

(e) [5 pts]

Go to section (e) of the file `HW3_driver.jl` and implement a function that takes an integer m as an input and returns an $m \times m$ matrix as introduced in problem 2 (d). Plot the error of the solution when solving equations in this matrix as a function of m. Compare the error to the built-in solution (the \ operator). Draw your conclusions from this comparison.

References

[1] H. V. Henderson and S. R. Searle. On Deriving the Inverse of a Sum of Matrices. *SIAM Review*, 23(1):53–60, 1981. Publisher: Society for Industrial and Applied Mathematics.