CS 111: Homework 7: Due by 11:59 pm Sunday, November 14, 2021

Submit your paper as one PDF file, and tell GradeScope which page(s) each problem is on. If you worked with a partner, you must each separately write up and turn in your own homework paper, and report the name of your partner. No groups of more than two.

Background. In this homework you'll learn how to solve least squares problems using a different matrix factorization instead of the SVD. Given a matrix A with m rows and n columns (where $m \geq n$), the *QR factorization* factors

$$A = QR$$

where Q is m-by-m and orthogonal, and R is m-by-n and upper triangular. When $m > n$, there is also an "economy size" QR factorization, in which Q is m-by-n with orthogonal columns, and R is square and upper triangular. Please read NCM Section 5.5 to learn more about the QR factorization.

You can use QR factorization to solve $Ax = b$ when A is square, because $QRx = b$ is the same as $Rx = Q^T b$. That's Problem 1 below. As you'll see in Problems 2 and 3, you can also use QR factorization to solve the least squares problem $\min_x ||Ax - b||_2$ when $m > n$. In practice QR is a little less expensive than SVD for dense matrices (by maybe a factor of 2), and it can be much less expensive when A is sparse. Numpy's `npla.lstsq()` uses SVD, but most large-scale least squares computations use QR.

1. Let

$$A = \begin{pmatrix} 4 & -1 & -1 \\ -1 & 4 & -1 \\ -1 & -1 & 4 \end{pmatrix}$$

and let $b = (15, -3, 12)^T$.

1.1. Use the `scipy` QR factorization routine `scipy.linalg.qr()` to compute the two matrices (orthogonal and upper triangular) that constitute the QR factorization of A. Print Q and R, and verify by inspection that R is upper triangular. Verify that Q is orthogonal by comparing $Q^T Q$ to the identity matrix. Verify that the factorization is correct by multiplying the factors and comparing the result to A.

Note: When we say to "compare" one matrix A to another matrix B, it is sufficient to compute (and print) the relative norm of their difference,

```
npla.norm(A - B) / npla.norm(B)
```

Which matrix norm to use? As you can see by saying `npla.norm?`, the default matrix norm in `numpy` is the *Frobenius norm*, which is the square root of the sum of the squares of all n^2 elements of the matrix. The Frobenius norm is a good one to use for comparing two matrices, because it's much easier to compute than the 2-norm, but it isn't as useful in the analysis of matrix algorithms.

1.2. Use `cs111.Usolve()` and/or `cs111.Lsolve()` to compute the solution x to $Ax = b$ from the QR factors, without calling any other factorization or solve routine. (You are allowed to transpose any matrix if you want.) Verify that x is correct by computing (and showing) the relative residual norm. Show all the Jupyter input and output for your computations.

2. In this problem you will delve into the similarities and differences between the "'full-size" and "economy-size" QR factorizations of a matrix with more rows than columns. Start by generating a random 9-by-5 matrix A, using `np.random.rand()`. Show all your work in Jupyter.

2.1. Use `Q1, R1 = scipy.linalg.qr(A)` to generate the full-size QR factorization of A. What are the dimensions of Q_1? Of R_1? Verify that Q_1 is orthogonal by comparison to an identity matrix. Verify by inspection that R_1 is upper triangular; note what "triangular" means for a non-square matrix. Verify that in fact $Q_1 R_1 = A$.

2.2. Use `Q2, R2 = scipy.linalg.qr(A, mode='economic')` to generate the economy-size QR factorization of A. What are the dimensions of Q_2? Of R_2? What is $Q_2^T Q_2$? Is Q_2 orthogonal? Why or why not? Verify by inspection that R_2 is upper triangular. Verify that in fact $Q_2 R_2 = A$.

In English words, what is the relationship between Q_1 and Q_2? Use python to compute the relative Frobenius norm of a difference of two matrices to demonstrate that relationship.

In English words, what is the relationship between R_1 and R_2? Use python to compute the relative Frobenius norm of a difference of two matrices to demonstrate that relationship.

3. Here you will solve a least-squares approximation problem with your 9-by-5 matrix A from Problem 2 above.

Begin by using `b = np.random.rand(9)` to generate a random right-hand side b. (It's important for this experiment that b is random.) What does it mean to look for a solution to $Ax = b$ in this case? We would need a 5-vector x that satisfies all 9 equations represented by the rows of A, but such a vector (almost certainly) doesn't exist because the linear equation system is *overdetermined.* Instead, we will seek the *least-squares* solution, which is the 5-vector x that minimizes the 2-norm of the residual $b - Ax$. The least-squares problem is

$$\min_x ||b - Ax||_2.$$

Since the system is overdetermined, we do not expect there to exist an x that makes the residual norm zero. It is a theorem, though, that for the least-squares solution the residual vector is orthogonal (perpendicular) to every column of A. That is, if x is the value that achieves the smallest possible residual $r = b - Ax$, as measured by the 2-norm, then $A^T r = 0$ is the vector of all zeros.

3.1 Use `npla.lstsq()` to compute the least-squares solution x. Print x and the relative residual norm $||b - Ax||_2/||b||_2$. Verify that the residual is orthogonal to the columns of A by computing (and printing) $||A^T r||_2$.

3.2 Use the full-size factorization $Q_1 R_1 = A$ from Problem 2 to solve for x as follows, showing your work in Jupyter. First compute $y = Q_1^T b$. Then "solve" $R_1 x = y$ for x, which will require extracting a submatrix of R_1 before calling `cs111.Usolve()`. As above, print x and the relative residual norm, and verify that the residual is orthogonal to the columns of A.

3.3 Use the economy-size factorization $Q_2R_2 = A$ from Problem 2 to solve for x as follows, showing your work in Jupyter. First compute $y = Q_2^T b$. Then solve $R_2x = y$ for x. Notice that this time R_2 is square, so you can just call `cs111.Usolve()`. Why does this give the same result? As above, print x and the relative residual norm, and verify that the residual is orthogonal to the columns of A.

3.4 Finally, change the right-hand side b to be equal to the row sums of A (by multiplying by the appropriate vector of 1s). Use any of the methods above (your choice) to compute the new least-squares solution x. What is the relative residual now? What is unusual about the new relative residual, and why did this happen? Explain in English.