Project 3 for Math 2605

You may use any language you like to do the programming. Java, C and MATLAB are good choices, but use whatever you like. Turn in source code with your results. You may discuss the project with others, but may not copy code or results.

This project concerns the Jacobi algoithm. The idea is to compare the theoretical bound on how fast it runs with actual practice.

---

(1) Write a program that does the following:

It should randomly generate a 5×5 symmetric matrix A, and should then run the Jacobi algorithm, producing a sequence of successively "more diagonal" matrices B, and should stop when

$$\text{Off}(B) = \sum_{i \neq j} B_{i,j}^2 \leq 10^{-9}$$

After each partial diagonalization step, record the value of $\text{Off}(B)$, generating a (decreasing!) sequence of numbers. The program should return this sequence of numbers.

(2) To analyze your data, graph it. If b_k denote the value of $\ln(\text{Off}(B))$ after the kth partial diagonalization, we will have the theoretical bound

$$b_k \leq k \ln(9/10) + \ln(\text{Off}(A)) \ .$$

(The value $9/10$ comes from $1 - 2/(n^2 - 2n)$ with $n = 5$). Plot your points (k, b_k) together with a graph of the line

$$y = x \ln(9/10) + \ln(\text{Off}(A)) \ .$$

do this for 10 randomly generated matrices. How does the actual data compare with the theoretical bound?

(3) The sorting step, in which we look for the largest off diagonal entry is time consuming for large matrices. What happens if you simply do not bother with this, but just "sweep through" the upper right entries in a systematic fashion? Implement this, and again produce graphs for 10 random 5×5 matrices comparing practice with this version of the new "no sorting" algorithm with the old theoretical bound (for the "with sorting" version). What do you conclude about the importance of sorting versus its expense?

That specifies the project. You can produce the graphs by hand if need be, but try to get your program to draw the graphs. If you have the experience with java, try making an applet that does this and produces the graph. Ideally, the applet would also show the original random matrix, and the five diagonal values at the end of the run, as well as the graph. It would also be nice to have radio butns to toggle between the "with sorting" and "no sorting" versions. A nice implementation like this will earn extra credit.

No matter how you do it, *write up a report* – a few paragraphs – describing what you did, why, and what you concluded form the comparrisons.