Computational Linear Algebra 2021/22: Mastery Coursework

Prof. Colin Cotter, Department of Mathematics, Imperial College London

This coursework is the mastery component for MSc/MRes/4th year MSci students only. Do not attempt it if you are a 3rd year student. Your submission, which must arrive before the deadline specified on Blackboard, will consist of three components for the submission.

1. A pdf submitted to the Mastery dropbox, containing your article of 10 pages or less.
2. A SHA tagging the revision of your repository, containing your code used to produce the numerical examples presented in your article.

If you have any questions about this please ask them in the Ed Discussions forum.

Please note that it is very important to write your own text for this report, and it will be checked for plagiarism. My advice is to read the sources, take shorthand notes, put them to one side and then write your own text.

Mastery exercise This exercise is about the paper by Bai et al. (2003), which provided a new iterative technique for a particular class of matrix vector problems. In 10 pages or less (including references), describe the basic idea of the technique, as well as that of additional references that develop the idea, such as those cited in this document. You should quote theoretical results where appropriate for the exposition (i.e., where they help explain the usage, advantages and disadvantages of the method), but should not repeat detailed proofs (although briefly explaining some basic ideas behind a proof may be appropriate). You should also conduct your own computational explorations of the methods described in some simple examples that illustrate your points (especially if they confirm theoretical results). You should also provide references to relevant related work. If your document exceeds 10 pages then only the first 10 pages will be marked.

Marking guidance

1. 90-100 Excellent explanation of the fundamental idea and its extensions, with creative and novel investigation through numerical examples that are highly relevant to applications, and relevant references to published research. Clear, complete code using Numpy vector operations appropriately, with well-designed tests and making use of programming exercises from the course where appropriate. Excellent presentation.
2. 80-89 Excellent explanation of the fundamental idea and its extensions, with good investigation through numerical examples that are highly relevant to applications, and relevant references to published research. Clear, complete code using Numpy vector operations appropriately, with well-designed tests and making use of programming exercises from the course where appropriate. Very good presentation.
3. 70-79 Very good explanation of the fundamental idea and its extensions, with some investigation through numerical examples, and relevant references to published research. Well-written code using Numpy vector operations appropriately, with testing, making use of programming exercises from the course where appropriate. Very good presentation.
4. 65-69 Good explanation of the fundamental idea, with investigation through numerical examples, and references to published research. Readable code using Numpy vector operations in most appropriate places, with testing, making use of programming exercises from the course where appropriate. Very good presentation.
5. 60-64 Good explanation of the fundamental idea, with some good ideas for investigation through numerical examples but not developed far enough to draw useful conclusions, some account of references to published research. Readable code using Numpy vector operations in places, with partial testing, making use of some programming exercises from the course. Reasonable presentation.

6. 55-59 An explanation of the problem and the work of others on it, but without much independent implementation of the candidate's own. Some code is present with partial testing, making use of some programming exercises from the course, but also relying on Numpy/Scipy implementations where exercises were incomplete.
7. 50-54 As above but in some way defective. For example, one of: few references, some unclear text, poorly presented, code untested, Numpy vector operations not used; however still showing some understanding.
8. 40-49 Poor understanding; several of - few references, some unclear text, poorly presented, code untested, Numpy vector operations not used.
9. 20-39 Very poor understanding; no code present, few references, unclear text, poorly presented.
10. 0-19 No evidence of understanding; no code present, scant references, unclear text, very poor presentation.

References

Bai, Z.Z., Golub, G.H., Li, C.K., 2006. Optimal parameter in Hermitian and skew-Hermitian splitting method for certain two-by-two block matrices. SIAM Journal on Scientific Computing 28, 583–603.

Bai, Z.Z., Golub, G.H., Lu, L.Z., Yin, J.F., 2005. Block triangular and skew-Hermitian splitting methods for positive-definite linear systems. SIAM Journal on Scientific Computing 26, 844–863.

Bai, Z.Z., Golub, G.H., Ng, M.K., 2003. Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. SIAM Journal on Matrix Analysis and Applications 24, 603–626.

Bai, Z.Z., Golub, G.H., Ng, M.K., 2007. On successive-overrelaxation acceleration of the Hermitian and skew-Hermitian splitting iterations. Numerical Linear Algebra with Applications 14, 319–335.

Bai, Z.Z., Golub, G.H., Pan, J.Y., 2004. Preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite linear systems. Numerische Mathematik 98, 1–32.

Benzi, M., 2009. A generalization of the Hermitian and skew-Hermitian splitting iteration. SIAM Journal on Matrix Analysis and Applications 31, 360–374.