Università della Svizzera italiana | Faculty of Informatics | Institute of Computing CI

Numerical Computing **2022**

Student: Kelvin Likollari Discussed with: Gerald Prendi

Solution for Project 5

Due date: Wednesday, 7 December 2022, 11:59 PM

Numerical Computing 2022 — Submission Instructions
(Please, notice that following instructions are mandatory: submissions that don't comply with, won't be considered)

- Assignments must be submitted to iCorsi (i.e. in electronic format).
- Provide both executable package and sources (e.g. C/C++ files, Julia). If you are using libraries, please add them in the file. Sources must be organized in directories called:
  Project_number_lastname_firstname
  and the file must be called:
  project_number_lastname_firstname.zip
  project_number_lastname_firstname.pdf
- The TAs will grade your project by reviewing your project write-up, and looking at the implementation you attempted, and benchmarking your code's performance.
- You are allowed to discuss all questions with anyone you like; however: (i) your submission must list anyone you discussed problems with and (ii) you must write up your submission independently.

The purpose of this assignment is to gain insight on the theoretical and numerical properties of the Conjugate Gradient method. Here we use this method in an image processing application with the goal of deblurring an image given the exact (noise-free) blurred image and the original transformation matrix. Note that the "noise-free" simplification is essential for us to solve this problem in the scope of this assignment.

1. General Questions [10 points]

(1) In the Conjugate Gradient (CG) method, matrix A is a part of the following system of equations:

$$\mathrm{Ax} = \mathrm{b}$$

Effectively, in this equation, matrix A indicates the transformation matrix which comes from the repeated application of the image kernel, responsible for the blurring effect in the images. Matrix A is of $n^2 \times \mathrm{n}^2$ dimensions. Looking at the q1.jl file, a print statement has been added to calculate the size of the A.mat file. The results, as seen when running the code inside the Julia REPL indicate that the size of matrix A is 62500 × 62500, or else, written in proper matrix notation:

$$\mathrm{A} \in \mathrm{R}^{62500 \times 62500}$$

(2) By definition, a band matrix or banded matrix is a sparse matrix whose non-zero entries are confined to a diagonal band, comprising the main diagonal and zero or more diagonals on either side. Located in q1.jl file, a function has been implemented that takes into consideration the above definition provided by Wikipedia. Firstly it should be mentioned that the matrix has an upper and a lower triangular. After calling this function of matrix A, the result yielded is 753 diagonal bands, say for the upper diagonal part. However, it must be stated that 1 more diagonal band should be added, where this diagonal represents the main diagonal and remain with

$$753 + 1 = 754$$

diagonal bands. Last but not least, the same procedure (except than re-adding the diagonal must be applied to the other side of the diagonal. This yields the final result of 1507 diagonal bands.

(3) The first step to calculating the length of the vectorized blur image b, is importing the respective file. Having imported it and having applied vectorization to that file, the print statement used yields that the vectorized blur image b has a length of 62500. This can be also observed when taking into account the equation Ax = b. Effectively, taking into account subquestion 1, B $\in \mathrm{R}^{250\times250}$. Moreover, if vectorization is applied to matrix B, this means that the matrix B will have its entries stacked one on top of another. Thereby, if this happens, the size of the new (stacked-entry) matrix B will be 62500, as a result of multiplying 250 $\times$ 250

2. Properties of A [10 points]

(1) Looking at section 2.1 of the provided pdf, it is stated that A is symmetric, and of full rank matrix, but not positive definite. The issue of this matrix not being positive definite can be solved by solving the following system of equations:

$$\mathrm{Ax} = \mathrm{b}$$

Given this equation, and multiplying both sides with A^T, a positive-definite augmented transformation matrix, say $\bar{A}$ is yielded. Essentially,

$$A^T\mathrm{Ax} = A^T\mathrm{b}$$

will yield:

$$\bar{A}\mathrm{x} = \bar{b}$$

Given that $\bar{A}$ is symmetric, and at the same time, positive definite matrix with dimensions $n^2\times\mathrm{n}^2$, by definition, $\bar{A} = A^T\mathrm{A}$. Therefore, even if $\bar{A} = A^T\mathrm{A}$ gets transposed, that will change nothing, and the matrix will remain symmetric and of full rank. This can be proved using the following mathematical formula:

$$A^T\mathrm{A} \rightarrow (\mathrm{A}^T\mathrm{A})^T = \mathrm{A}A^T$$

which signifies that A is a symmetric matrix.

(2) For this section, p.183 of the textbook will be used. It is stated that $A^T\mathrm{Ax} = A^T\mathrm{b}$ can be also written as the problem of finding a vector, say x, which minimizes:

$$\phi(\mathrm{x}) = \tfrac{1}{2}\ \mathrm{Ax}\ \text{-}\ b^T\mathrm{x}$$

To prove why those two formulas are equivalent, the derivatives must be employed. Essentially, to obtain a critical point, the vector of the first derivatives of ϕ must be set equal to 0. This will yield a gradient of $\nabla\phi(x)$ = Ax - b = -r, and setting it to 0 will give back the original systems-of-equations problem:

$$\mathrm{Ax} = \mathrm{b}$$

Applying the derivatives into ϕ(x), the original function ϕ gets simplified to

$$\mathrm{Ax} - \mathrm{b} = 0 \rightarrow \mathrm{Ax} = \mathrm{b}$$

Generally, given that matrix A is a symmetric positive definite, finding a minimum of ϕ will be equivalent to Ax = b, as in that case, the Hessian is H_f, so only a minimum can be achieved, and if Ax = b, the gradient is no more.

3. Conjugate Gradient [30 points]

(1) The code for the conjugate gradient solver has been written after translating the given pseudocode into equivalent Julia code with proper modifications where applicable. The respective snippet of code is located in the Julia file.

(2) After having written the Julia code for the CG method, follows the convergence plotting of the system is defined by the 2 mat-test files. The results are depicted in the following images:

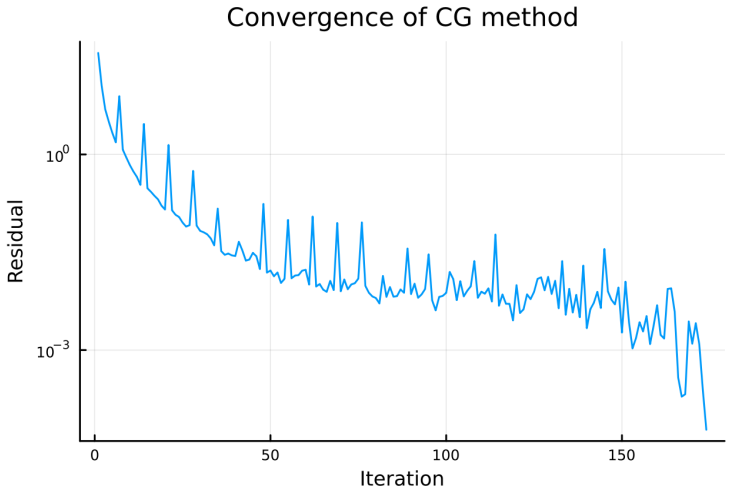

Figure 1: Convergence of the CG method

(3) The graph for the eigenvalues of the test matrix A is visualized below:

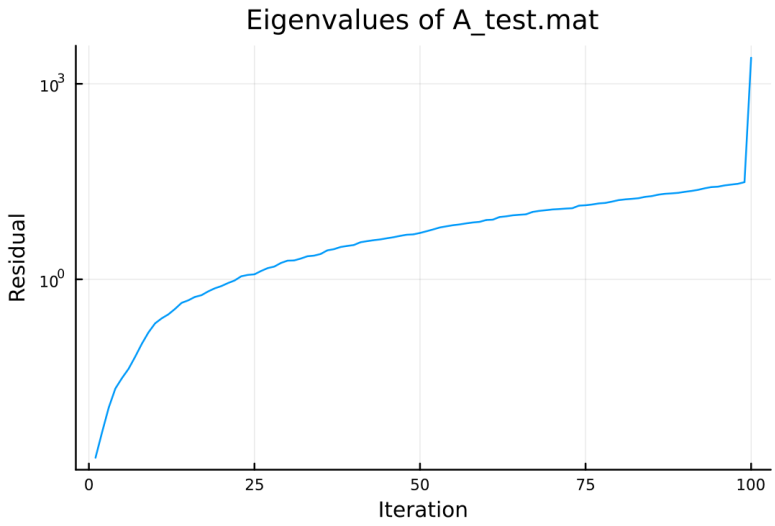

Figure 2: Eigenvalues of test mat file A

In general, the condition number describes the relation of the solution x, to changes in b, in the equation Ax = b. As seen from the print statement in the corresponding Julia file, the condition number is 1.3699623789128773e6, and it is pretty obvious that it is enormously large. Because the condition number is extremely large, this means at the same time that small changes in b shall result in large changes in x. By definition, this is called an ill-conditioned system and it can be claimed that the same definition holds for this exercise's mat file. In addition, the convergence rate of CG is hindered as a consequence of the large condition number.

(4) To comment on the residual monotonicity, sections 2 and 3 of the provided PDF will be used as a reference. A starting comment that can be made on the residual monotonicity is that when working with finite numbers, the residual is not always monotonically decreasing. There might be small perturbations (i.e alternations between the monotonicity). The only thing that can be claimed to be monotonically decreasing is the distance from the solution. This will yield at the same time the fact that f is also monotonically decreasing.

4. Deblurring problem [35 points]

(1) The blurred image looks as follows:

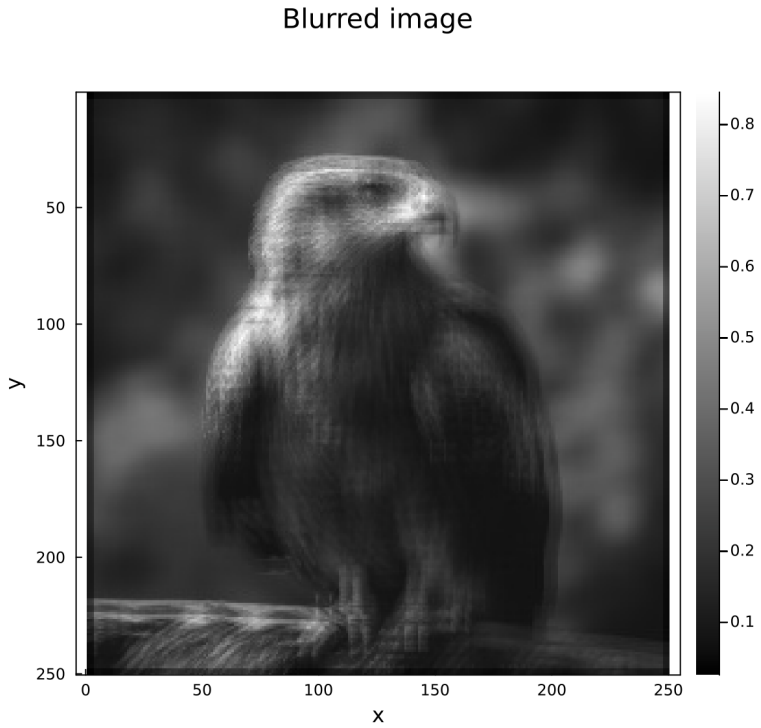

Figure 3: Blurred image

The unblurred image, using the Julia-implemented myCG function is depicted below:

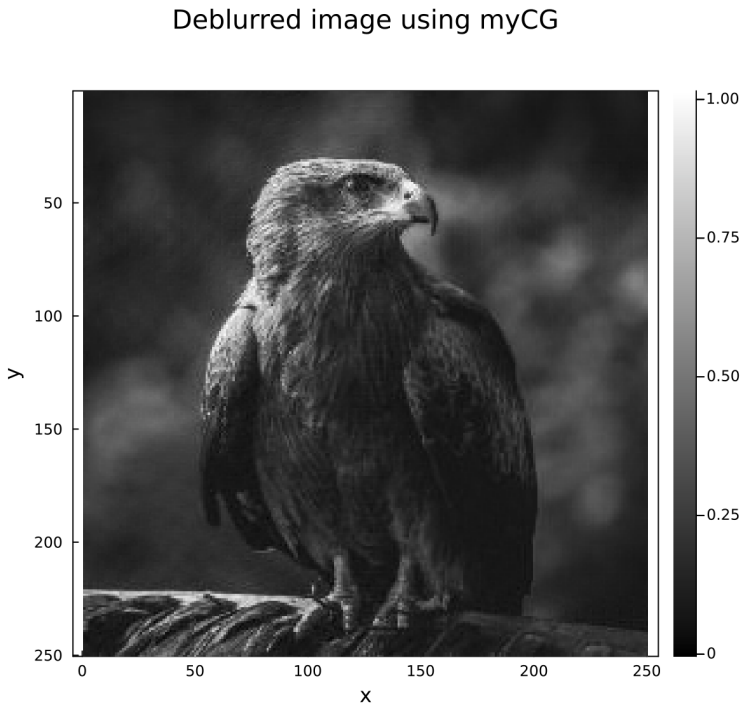

Figure 4: Deblurred image using myCG

The unblurred image, using the Julia-function named cg is depicted below:

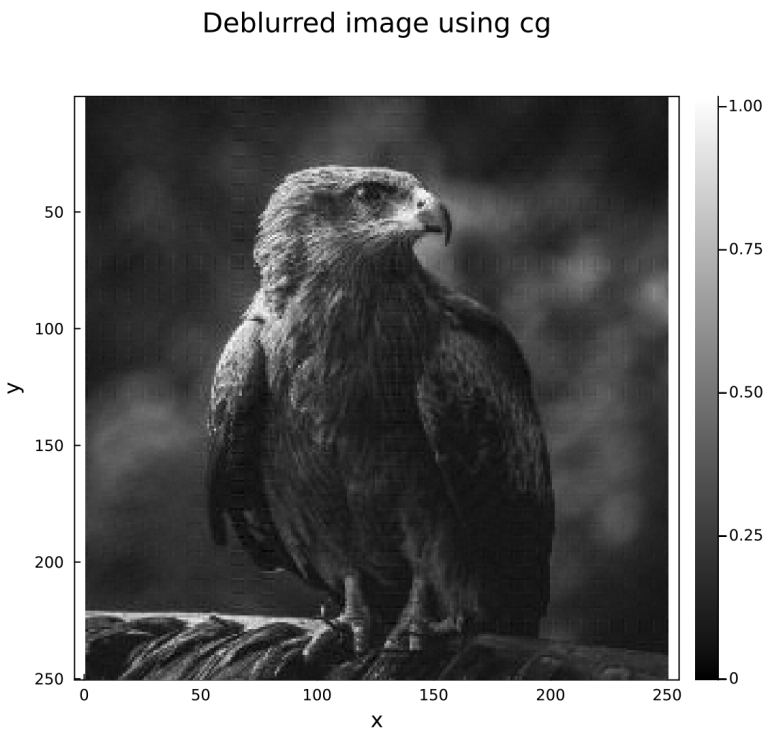

Figure 5: Deblurred image using cg

The Conjugate Gradient method was utilized to deblur the original blurred image to a significant extent. The Julia function cg performs the same task in terms of deblurring the original image. When comparing which of the aforementioned methods performs (deblurs) better, the results demonstrate that both photos are the same, which is correct, as this is how both algorithms used for deblurring should operate.

The residual and iteration convergence of both cg and myCG are visualized below:

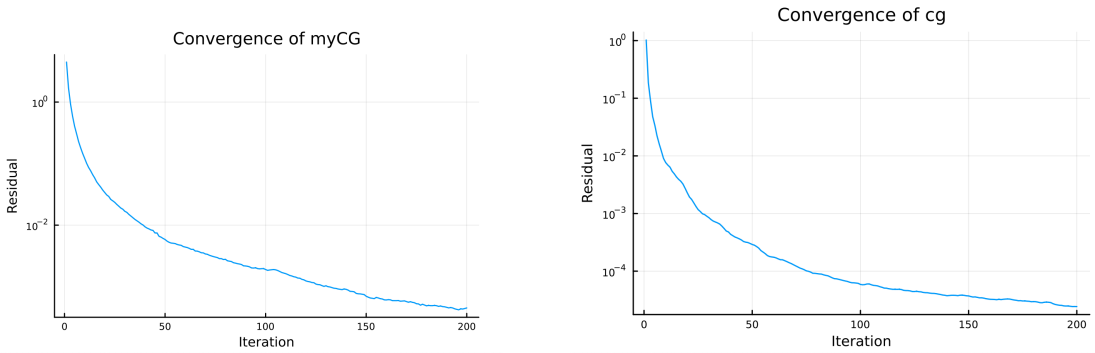

Figure 6: Convergence: myCG-cg

Since the results obtained from deblurring the original image using the myCG and cg methods are the same, this, at the same time, will imply that their convergences are also identical.

(2) The preconditioned CG would be worth the added computational cost when the system is ill-conditioned.

If we are deblurring a lot of images with the same blur operator, this means that in order to solve the deblurring problem, we have to solve the same linear system many times. In this

case, it is more efficient to compute the incomplete Cholesky factorization once and then use it to solve the linear system many times.

1. Wikipedia
2. https://math.stackexchange.com/questions/396844/non-monotonic-decrease-of-residuals-in-conjugate-gradients
3. https://see.stanford.edu/materials/lsocoee364b/11-conj$_g rad_s lides.pdf$
4. https://www.scirp.org/html/2644.html