CSC - IT Center for Science

CGNR with ILQ preconditioning

Author:

Kasper Rantamäki
kasper.rantamaki@csc.fi

June 2023

1 Introduction

In many engineering and scientific computing applications the problems can commonly be simplified into a system of linear equations - usually denoted as:

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

where $A \in \mathbb{R}^{m \times n}$, $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{b} \in \mathbb{R}^n$. Unfortunately, as problems increase in scale so do the dimensions of the coefficient matrix A, which quickly leads to issues with memory. Luckily, A is often very sparse and thus representable with e.g. CRS or CCS formats in a memory efficient fashion.

While A could be assumed sparse the same doesn't necessarily hold for the inverse of A and hence even if we were to ignore the time complexity of the inversion process we could very well run out of memory when solving the system in 1 directly. To combat this a considerable effort has gone into solving it iteratively. We will focus on the (probably) most well known iterative method - the conjugate gradient method.

2 CG and CGNR

2.1 Conjugate gradient method

To understand the conjugate gradient method (from here on denoted CG) it is important to note that for symmetric positive definite A the solutions $\mathbf{x}$ to problems:

$$A\mathbf{x} = \mathbf{b} \tag{2}$$

$$\min_{\mathbf{x}} . \ \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} \tag{3}$$

are equivalent. The function being minimized in 3 is generally known as the energy function. As the energy function is quadratic in nature the problem in 3 is a convex unconstrained nonlinear optimization problem, which is solvable with multitude of different algorithms, but is usually done with *gradient descent*. Furthermore, if the search directions in the gradient descent are chosen to be A-orthogonal the problem simplifies to the well known conjugate gradient method.

2.2 Conjugate gradient on the normal equations

Main limitation of CG is that the coefficient matrix A has to be s.p.d. for the equivalence of problems 2 and 3 to hold. There are multiple generalizations of CG like BiCG and CGS, but we will focus on the conjugate gradient on the normal equations (CGNR).

CGNR doesn't solve 1 per se, but a related problem:

$$A^T A\mathbf{x} = A^T\mathbf{b} \tag{4}$$

Here $A^T A$ is s.p.d independent of the form of A.

The major problem with CGNR is that the condition number of the matrix $A^T A$ is the square of the condition number of A. The convergence of CG is tied to the condition number as the associated error after i iterations is bounded by:

$$\|\mathbf{x}_i - \mathbf{x}\| = \|\mathbf{e}_i\|_A \leq 2\|\mathbf{e}_0\|_A \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i \tag{5}$$

where $\kappa = \kappa_2(A)$.

Hence, to efficiently use 4 some way for decreasing the condition number is detrimental. And preconditioners are designed for this exact task.

3 Preconditioners

Very generally speaking preconditioners are matrices M that modify 1 into form:

$$MA\mathbf{x} = M\mathbf{b} \tag{6}$$

Now if $M \approx A^{-1}$ the problem should be significantly easier to solve. Obviously, as M is only an approximation there are no guarantees that matrix MA is symmetric and thus 6 is commonly written with $M = M_1 M_2$ as:

$$M_1 A M_2 (M_2^{-1}\mathbf{x}) = M_1 \mathbf{b} \tag{7}$$

However, in the case of CGNR this might be sub-optimal. If we utilize the symmetry of matrix $A^T A$ we could formulate a preconditioned system as:

$$A^T M A\mathbf{x} = A^T M\mathbf{b} \tag{8}$$

Here the matrix M must obviously be symmetric, but if we additionally assume it to be positive definite we could rewrite 8 by taking the Cholesky decomposition of M as:

$$(LA)^T LA\mathbf{x} = A^T L^T L\mathbf{b} \tag{9}$$

where L is the Cholesky factor of M and thus lower triangular.

Ideally, we would like to find L such that $(LA)^T LA \approx I$. Matrix for which such holds is known as an orthogonal matrix and is usually found with some ortogonalization process like *Gram-Schmidt process*.

4 Gram-Schmidt and QR decomposition

Orthogonalization is commonly formulated as a QR decomposition:

$$A = QR \tag{10}$$

where Q is a matrix with orthogonal columns and R is an upper triangular matrix.

QR decomposition is very easy to arrive to by e.g. Gram-Schmidt process. In Gram-Schmidt a given set of vectors $\{\mathbf{a}_1,\ ...,\ \mathbf{a}_n\}$ is orthogonalized as:

$$\begin{aligned}
\mathbf{u}_1 &= \mathbf{a}_1 \\
\mathbf{u}_2 &= \mathbf{a}_2 - \mathrm{proj}_{\mathbf{u}_1}(\mathbf{a}_2) \\
\mathbf{u}_3 &= \mathbf{a}_1 - \mathrm{proj}_{\mathbf{u}_1}(\mathbf{a}_3) - \mathrm{proj}_{\mathbf{u}_2}(\mathbf{a}_3) \\
&\vdots \\
\mathbf{u}_n &= \mathbf{a}_n - \sum_{j=1}^{n-1} \mathrm{proj}_{\mathbf{u}_j}(\mathbf{a}_n)
\end{aligned} \tag{11}$$

where $\mathrm{proj}_{\mathbf{u}}(\mathbf{a}) = \frac{\langle \mathbf{a},\mathbf{u}\rangle}{\langle \mathbf{u},\mathbf{u}\rangle}\mathbf{u}$ is the projection operator. Orthonormal vectors $\mathbf{e}_i$ are then found by normalizing vectors $\mathbf{u}_i$.

Considering equations in 11 we can redefine vectors $\mathbf{a}_i$ as:

$$\begin{aligned}
\mathbf{a}_1 &= \langle \mathbf{e}_1, \mathbf{a}_1\rangle \mathbf{e}_1 \\
\mathbf{a}_2 &= \langle \mathbf{e}_1, \mathbf{a}_2\rangle \mathbf{e}_1 + \langle \mathbf{e}_2, \mathbf{a}_2\rangle \mathbf{e}_2 \\
&\vdots \\
\mathbf{a}_n &= \sum_{j=1}^{n} \langle \mathbf{e}_j, \mathbf{a}_n\rangle \mathbf{e}_j
\end{aligned} \tag{12}$$

This clearly leads to the formulation of matrices Q and R as:

$$Q = [\mathbf{e}_1 \quad \mathbf{e}_2 \quad \dots \quad \mathbf{e}_n] \tag{13}$$

and

$$R = \begin{bmatrix}
\langle \mathbf{e}_1, \mathbf{a}_1\rangle & \langle \mathbf{e}_1, \mathbf{a}_2\rangle & \langle \mathbf{e}_1, \mathbf{a}_3\rangle & \dots & \langle \mathbf{e}_1, \mathbf{a}_n\rangle \\
 & \langle \mathbf{e}_2, \mathbf{a}_2\rangle & \langle \mathbf{e}_2, \mathbf{a}_3\rangle & \dots & \langle \mathbf{e}_2, \mathbf{a}_n\rangle \\
 & & \langle \mathbf{e}_3, \mathbf{a}_3\rangle & \dots & \langle \mathbf{e}_3, \mathbf{a}_n\rangle \\
 & & & \ddots & \vdots \\
 & & & & \langle \mathbf{e}_n, \mathbf{a}_n\rangle
\end{bmatrix} \tag{14}$$

Similarly, a LQ decomposition, where L is lower triangular and Q orthogonal can be found for a given matrix. Indeed, the LQ decomposition of A would be the QR decomposition of A^T as:

$$A^T = QR \;\rightarrow\; A = (QR)^T = R^T Q^T \tag{15}$$

where R^T is lower triangular.

5 Incomplete LQ decomposition

As outlined in equations 11 and 12 the orthogonalization is done iteratively, by projecting the current vector against all prior orthogonalized vectors. If fully done this would be a $\mathcal{O}(n^3)$ operation. However, if computational efficiency is preferred over the accuracy of the result then the orthogonalization could be done only against the p preceeding vectors. This wouldn't lead to a fully orthogonal matrix, but with large enough value of p the orthogonality would hopefully be significantly better.

This incomplete formulation of the LQ decomposition could be computed in $\mathcal{O}(pn^2)$ operations and would result in a lower triangular matrix L with the main diagonal and the p subdiagonals filled. Inversion of such a matrix can be efficiently done with e.g. Gaussian elimination in $\mathcal{O}(pn^2)$ operations.

Overall the matrix L should be solvable in a very efficient manor and result in a preconditioner that can potentially work well, but of course that is something that cannot be theoretically verified, but would require the implementation and testing of the aforementioned algorithm.

6 Discussion

Interestingly, solving the normal equations problem in 4 is equivalent to minimizing the least squares problem:

$$\min_{\mathbf{x}}.\; \|\mathbf{b} - A\mathbf{x}\|_2^2 \tag{16}$$

In the context of least squares problems the use of incomplete QR decompositions has been hypothesized in e.g. [1] and [2], but in neither is the formulation exactly as outlined here. Due to the simplicity of the outlined method this leaves two possibilities: either the method is mentioned in some paper that is lost to time or there is some glaring error that would be realized when the method is implemented. Which one is more likely is up for the reader to decide.

References

[1] Li, N., Saad, Y. (2006) MIQR: A Multilevel Incomplete QR Preconditioner for Large Sparse Least-Squares Problems, SIAM Journal on Matrix Analysis and Applications

[2] Lipitakis, A. D., Filelis-Papadopoulos C. K., Gravvanis G. A., Anagnostopoulos D. (2020) A class of Generic Approximate Sparse Pseudoinverse Matrix Techniques based on incomplete QR factorization, 2020 International Conference on Computational Science and Computational Intelligence (CSCI)