LQ-Schur Projection on Large Sparse Matrix Equations

Daniel Boley* and Todd Goehring
Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455, USA

Abstract

A new paradigm for the solution of nonsymmetric large sparse systems of linear equations is proposed. The paradigm is based on an LQ factorization of the matrix of coefficients, i.e. factoring the matrix of coefficients into the product of a lower triangular matrix and an orthogonal matrix. We show how the system of linear equations can be decomposed into a collection of smaller independent problems which can then be used to construct an iterative method for a system of smaller dimensionality. We show that the conditioning of the reduced problem cannot be worse than that of the original, unlike Schur complement methods in the nonsymmetric case. The paradigm depends on the existence of an ordering of the rows representing the equations into blocks of rows which are mutually structurely orthogonal, except for a last block row which is coupled to all other rows in a limited way.

1 Introduction

The solution of large sparse systems of linear equations is a difficult and often time-consuming task. The focus of this paper is on a new paradigm for constructing a reduced order system from the original system, as a generalization to the methods based on the Schur complement [1]. The Schur complement method starts by partitioning a matrix $\mathbf{A}$ and applying a block Gaussian elimination process to this matrix:

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{I} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{21} \\ \mathbf{0} & \mathbf{U}_{22} \end{pmatrix}.$$

Then the Schur complement of $\mathbf{A}_{22}$ is $\mathbf{S} = \mathbf{L}_{22}\mathbf{U}_{22}$, where $\mathbf{L}_{21} = \mathbf{A}_{21}\mathbf{A}_{11}^{-1}$ and $\mathbf{U}_{22} = \mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{A}_{12}$ [6]. Typically, this method is useful when the equations are ordered so that systems of equations involving $\mathbf{A}_{11}$ are relatively easy to solve, so that $\mathbf{S}$ can be used in an iterative method without being formed explicitly. However the ordering within $\mathbf{A}_{11}$ can have a dramatic effect on the numerical quality of the resulting representation of the Schur

*e-mail: boley@cs.umn.edu

complement. This can lead to difficult trade offs between the numerical accuracy on the one hand and the sparsity or complexity of solution on the other.

In this paper, we describe a generalization of this method in which the Gaussian elimination is replaced with a triangular orthogonalization process, i.e. a QR factorization [5]. Since we are using a row based scheme instead of a column based scheme, we will use instead an LQ factorization, in which a matrix is factored into $\mathbf{A} = \mathbf{LQ}$ with $\mathbf{L}$ lower triangular and $\mathbf{Q}$ orthogonal. In this case, the numerical accuracy is independent of the ordering of the rows, so we are free to choose an order to enhance the sparsity of the $\mathbf{LQ}$ factors as much as possible.

It is well known that the LQ process results in much more fill than the Gaussian elimination process. However we make certain assumptions on the structure of the original matrix which will limit the fill and hence make a practical algorithm. The main assumption is that we have ordered the rows of the matrix $\mathbf{A}$ into a number of blocks which are mutually structurally orthogonal, except for a last block of rows which are coupled to all the other blocks. The motivation for this structure comes from the discretization of differential operators over physical domains.

Domain Decomposition addresses the topic of solving partial differential equations by partitioning the domain into smaller semi-independent subdomains. There are many variations: overlapping vs non-overlapping subdomains, iterative Schwarz methods vs Schur complement methods (where interior nodes are solved directly and the boundary nodes are solved iteratively) [1, 7], as well as preconditioners of many types (ILU, ILUT, multilevel, etc.), some of which are motivated by the underlying differential equations [6, 1]. The present paper is limited in scope to the linear algebra aspects of using an LQ decomposition on matrices with a sparsity structure commonly found among matrices derived from discrete differential operators. The relation with any underlying differential equations, including a discussion of convergence or application-specific preconditioners is beyond the scope of this short paper.

In this paper, in Sec. 2 we motivate the assumptions made regarding the structure of the matrix operator, in Sec. 3 we introduce the overall setup and notation. In Sec. 4 we present the details of the solution process, in Sec. 5 we relate the iteration operator to the Schur complement, and in Sec. 6 we sketch some preconditioners peculiar to this formulation for the reduced order system. We end with a short example in Sec. 7 and a summary in Sec. 8.

2 Motivation of Structure

The methods discussed in this paper have been developed for systems with structures similar to those derived from the finite-element or finite-difference discretization of a partial differential equation over some domain. In order to carry out the process, we make certain assumptions about the way the domain is partitioned into subdomains. These assumptions are described below.

1. The physical domain has been covered with a finite collection of nodes or vertices, each representing the unknowns at a single location or element in the domain. To each node corresponds an equation which couples the unknown values at that node with

those of its immediate neighbors. This is the typical situation that results from any finite-difference method with a usual 5 or 9 point stencil in 2D or 7 point stencil in 3D, or a finite-element method in which the values within each element are coupled only those of the immediately adjacent elements. This structure is represented by a graph with vertices corresponding to the nodes and edges corresponding to coupling between nodes in a single equation (see e.g. [6, 7]). In the following, we discuss the ordering of the matrix based on a partitioning of the graph, which is a discrete representation of the original domain. The partitioning of the graph would correspond to a partitioning of the original domain into disjoint subdomains. To emphasize that we are focusing on the discrete problem, we refer to subgraphs instead of subdomains.

2. The entire graph has been partitioned by some algorithm into a collection of subgraphs, each separated from the neighboring subgraphs by a double-layered boundary. To see why a double layered boundary is useful in this situation, consider 4 nodes in the neighborhood of a boundary:

   $$\begin{tikzcd}
   \vdots & \vdots & \vdots & \vdots \\
   \cdots A \arrow[r, dash] & B \arrow[r, dash] & C \arrow[r, dash] & D \cdots \\
   \vdots & \vdots & \vdots & \vdots
   \end{tikzcd}$$

   where A is in the interior of one subgraph, B is on the boundary of the same subgraph, C is on the boundary of a different subgraph, and D is in the interior of this latter subgraph. We say that B is part of the same subgraph as A, and C is part of the same subgraph as D, but B, C lie on the boundary of their respective subgraphs. In a parallel processing environment where each subgraph would be allocated to a different processor, A, B would be put on together on one processor, while C, D would be put together on another.

   The equation centered at node A couples A with node B and with other nodes from its own subgraph (both in the interior and possibly on the boundary). But A is not coupled with nodes from any other subgraph, whether in the interior (such as D) or on the boundary (such as C). Analogously, the equation centered at B couples B with other nodes on the boundary of its own subgraph and neighboring subgraphs (such as C), and with nodes in the interior of its own subgraph (such as A). But B is not coupled with any node in the interior of any other subgraph (such as D). In other words, there are edges within each subgraph, between the interior and the boundary of the same subgraph, and among the boundary nodes, but there are no edges between the interiors of two different subgraphs, nor between the boundary of one subgraph and the interior of a different subgraph.

   When the equations are assembled into a matrix operator, the result is that the rows centered at interior nodes (such as A) are automatically structurely orthogonal to rows centered on interior nodes of other subgraphs (such as D), as well as boundary nodes of other subgraphs (such as C). The rows centered on boundary nodes (such as B) are structurally orthogonal to rows centered on interior nodes of other subgraphs (such as D).

3. The equations are ordered in the typical way in Domain Decomposition with the boundary nodes (such as B and C above) ordered last. All the interior nodes of each subgraph are ordered together, after which come all the boundary nodes, again grouped by subgraph. The resulting matrix $\mathbf{A}$ will have the following form

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix},$$

where

- $\mathbf{A}_{11}$ is $n_1 \times n_1$ block diagonal, and each diagonal block represents the coupling among the interior nodes of a single subgraph;
- $\mathbf{A}_{12}$ is $n_1 \times n_2$ with a rectangular block "diagonal" structure, and each rectangular block represents the coupling between the interior of a single subgraph and the boundary of the same subgraph, as viewed from the interior;
- $\mathbf{A}_{21}$ is $n_2 \times n_1$ with a rectangular block "diagonal" structure, and each rectangular block represents the coupling between the interior of a single subgraph and the boundary of the same subgraph, as viewed from the boundary;
- $\mathbf{A}_{22}$ is $n_2 \times n_2$ and represents the coupling among all the boundary nodes.

Fig. 1 shows a typical structure for $\mathbf{A}$ derived from a standard 5 point stencil in 2D. We note that with this ordering, $\mathbf{A}_{11}, \mathbf{A}_{12}, \mathbf{A}_{21}$ all tend to be sparse, and within the top part, $(\mathbf{A}_{11} \quad \mathbf{A}_{12})$, the rows corresponding the each subgraph are structurally orthogonal to the rows corresponding to any other subgraph. The structural orthogonality allows one to orthogonalize the rows for each subgraph independently.

3 Problem Setup and Projectors

Let $\mathbf{A}$ be a $n \times n$ matrix and $\mathbf{b}$ be an n-vector, partitioned as

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}, \tag{1}$$

where $\mathbf{A}_{11}$ is $n_1 \times n_1$ block diagonal and hence relatively easy to "invert," and $\mathbf{A}_{22}$ is $n_2 \times n_2$. The matrix $\mathbf{A}$ would typically be constructed from a Domain Decomposition-like process, in which the block $\mathbf{A}_1 = (\mathbf{A}_{11} \quad \mathbf{A}_{12})$ represents the equations in the interior of a collection of disjoint subgraphs and $\mathbf{A}_2 = (\mathbf{A}_{21} \quad \mathbf{A}_{22})$ represents the equations on the boundaries between the subgraphs. We assume that the boundaries are "double layered" so that the rows corresponding to each subgraph's interior are structurally orthogonal to those for other subgraphs' interiors, as described in the previous section.

The goal is to solve the system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$. To this end, we decompose the solution $\mathbf{x}$ into two parts $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2$ with $\mathbf{x}_1$ lying in the space spanned by the rows of $\mathbf{A}_1$ and $\mathbf{x}_2$ lying in the space orthogonal to this row space:

$$\begin{aligned} &\mathbf{x}_1 \in \text{ROW-SPACE}\, \mathbf{A}_1 \\ &\mathbf{x}_2 \perp \text{ROW-SPACE}\, \mathbf{A}_1 \Longleftrightarrow \mathbf{x}_2 \in \text{NULLSPACE}\, \mathbf{A}_1. \end{aligned} \tag{2}$$

We denote the orthogonal projector for ROW-SPACE $\mathbf{A}_1$ as $\mathbf{P}_1$ and the orthogonal projector for the complementary space NULLSPACE $\mathbf{A}_1$ as $\mathbf{P}_2 \equiv \mathbf{I} - \mathbf{P}_1$. Then the second component of the solution can be written as $\mathbf{x}_2 = \mathbf{P}_2\mathbf{w}$ for some vector $\mathbf{w}$ to be determined.

The overall procedure we will develop can be summarized as follows.

1. Solve for $\mathbf{x}_1$ directly, where $\mathbf{x}_1$ satisfies

$$\begin{array}{l} \mathbf{A}_1\mathbf{x}_1 = \mathbf{b}_1 \text{(an underdetermined system)} \\ \text{such that } \mathbf{x}_1 \in \text{ROW-SPACE}\, \mathbf{A}_1. \end{array} \tag{3}$$

   For this step, we use the LQ factorization of $\mathbf{A}_1$, because parts of these factors will then be saved for later use. The constraint makes the solution unique.

2. Form the equations for the second component of the solution:

$$\begin{array}{l} \mathbf{A}_2\mathbf{x}_2 = \mathbf{r} \overset{\triangle}{=} \mathbf{b}_2 - \mathbf{A}_2\mathbf{x}_1 \\ \text{such that } \mathbf{x}_2 = \mathbf{P}_2\mathbf{w} \text{ for some vector } \mathbf{w} \text{ to be determined.} \end{array} \tag{4}$$

3. Solve equation (4) for $\mathbf{w}$ and then for $\mathbf{x}_2$. This is typically done by an iterative method, though in some cases one could even think of using a direct method. In actual fact, we solve the equations

$$\mathbf{A}_2\mathbf{P}_2\mathbf{w} = \mathbf{r} \overset{\triangle}{=} \mathbf{b}_2 - \mathbf{A}_2\mathbf{x}_1, \tag{5}$$

   where $\mathbf{w}$ lies in a restricted space to make the solution unique, so that in effect we are solving a system of reduced dimensionality.

The key to the success of the overall method is that we can represent the operator $\mathbf{A}_2\mathbf{P}_2$ in (5) in terms of $\mathbf{A}_2$ and sparse items constructed from $\mathbf{A}_1$. We never have to form $\mathbf{P}_2$ explicitly.

LQ Factorization

In order to carry out the intended solution procedure, we use a partial LQ factorization of $\mathbf{A}$. The entire LQ factorization of $\mathbf{A}$ is defined as

$$\mathbf{A} = \mathbf{LQ} \equiv \begin{pmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{Q}_1 \\ \mathbf{Q}_2 \end{pmatrix} \equiv \begin{pmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{Q}_{11} & \mathbf{Q}_{12} \\ \mathbf{Q}_{21} & \mathbf{Q}_{22} \end{pmatrix}, \tag{6}$$

where $\mathbf{L}, \mathbf{Q}$ is partitioned in conformity to the partitionings shown in (1). Here $\mathbf{L}$ is a lower triangular matrix and $\mathbf{Q}$ is an orthogonal matrix. Even though we define the entire LQ factorization, we compute only the top n_1 rows:

$$\mathbf{A}_1 = \mathbf{L}_{11}\mathbf{Q}_1. \tag{7}$$

The rest of the LQ factorization will be represented implicitly. Furthermore, we use $\mathbf{L}_{11}$ only during the computation of $\mathbf{x}_1$, after which it may be dropped, keeping only $\mathbf{Q}_1$ in later stages.

In terms of the LQ factorization, the orthogonal projectors can be written as

$$\mathbf{P}_1 = \mathbf{Q}_1^T\mathbf{Q}_1 \quad \text{and} \quad \mathbf{P}_2 = \mathbf{Q}_2^T\mathbf{Q}_2 = \mathbf{I} - \mathbf{Q}_1^T\mathbf{Q}_1. \tag{8}$$

4 Solution Process

We now go through the steps of the solution process in some more detail.

4.1 Solve for $\mathbf{x}_1$

We compute the LQ factorization of $\mathbf{A}_1$ (7). Given the structure we have assumed for the matrix $\mathbf{A}$, the rows in $\mathbf{A}_1$ corresponding to different subgraphs are structurally orthogonal. Hence the LQ factorization of each block can be computed independently and in parallel. If one thinks of using a Gram-Schmidt orthogonalization procedure on the rows for each subgraph, the result is that both $\mathbf{L}_{11}$ and $\mathbf{Q}_{11}$ will have the same block diagonal structure as $\mathbf{A}_{11}$, but each diagonal block will be more full. One is free to order the rows within each subgraph to reduce the fill, and we usually choose to order them by graph distance to the boundary, which the nodes closest to the boundary last. The block $\mathbf{Q}_{12}$ will have also have a rectangular diagonal-like block structure inherited from that of $\mathbf{A}_{12}$. We do not compute $\mathbf{L}_{21}$ or $\mathbf{Q}_{12}$ since the latter is often very full.

Writing (3) in terms of the LQ factorization yields $\mathbf{L}_{11}\mathbf{Q}_1\mathbf{x}_1 = \mathbf{b}_1$, so that the solution satisfying (3) is

$$\mathbf{x}_1 = \mathbf{Q}_1^T \mathbf{L}_{11}^{-1} \mathbf{b}_1. \tag{9}$$

Thus the vector $\mathbf{x}_1$ can be found by solving the triangular system $\mathbf{L}_{11}\mathbf{y}_1 = \mathbf{b}_1$, and then setting $\mathbf{x}_1 = \mathbf{Q}_1^T \mathbf{y}_1$.

4.2 Forming Second Set of Equations

To find $\mathbf{x}_2$, we must solve (5) for $\mathbf{w}$, and then set $\mathbf{x}_2 = \mathbf{P}_2\mathbf{w}$. However, we will find a reduced representation for (5) in terms of the partial LQ factorization. The right hand side of (5) can be expanded as follows:

$$\begin{aligned}
\mathbf{b} - \mathbf{A}\mathbf{x}_1 &= \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix} - \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix} \mathbf{x}_1 \\
&= \begin{pmatrix} \mathbf{0} \\ \mathbf{b}_2 - \mathbf{A}_2 \mathbf{Q}_1^T \mathbf{L}_{11}^{-1} \mathbf{b}_1 \end{pmatrix} \\
&\triangleq \begin{pmatrix} \mathbf{0} \\ \mathbf{r}_2 \end{pmatrix}.
\end{aligned} \tag{10}$$

We can expand the left hand side of (5) as follows:

$$\begin{aligned}
\mathbf{A}\mathbf{x}_2 \;=\; \mathbf{A}\mathbf{P}_2\mathbf{w} &= \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix} \mathbf{Q}_2^T \mathbf{Q}_2 \mathbf{w} \\
&= \begin{pmatrix} \mathbf{0} \\ \mathbf{A}_2 \mathbf{Q}_2^T \mathbf{Q}_2 \end{pmatrix} \mathbf{w} \qquad \text{(because } \mathbf{A}_1\mathbf{Q}_2^T = \mathbf{0}\text{)} \\
&= \begin{pmatrix} \mathbf{0} \\ \mathbf{L}_{22} \mathbf{Q}_2 \end{pmatrix} \mathbf{w} \qquad \text{(because } \mathbf{A}_2\mathbf{Q}_2^T = \mathbf{L}_{22}\text{)}.
\end{aligned} \tag{11}$$

Thus, it is seen that set of equations (5) actually reduces to a $n_2 \times n_2$ linear system involving the matrix $\mathbf{L}_{22}$. The catch is that we never compute $\mathbf{L}_{22}$ or $\mathbf{Q}_2$. Instead, we represent them using only $\mathbf{A}_2$ and the projector derived from $\mathbf{A}_1$:

$$\mathbf{A}\mathbf{P}_2 = \begin{pmatrix} \mathbf{0} \\ \mathbf{A}_2\mathbf{P}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{A}_2(\mathbf{I} - \mathbf{P}_1) \end{pmatrix}. \tag{12}$$

Combining the above, equation (5) reduces to

$$\mathbf{A}_2(\mathbf{I} - \mathbf{P}_1)\mathbf{w} = \mathbf{r}_2, \tag{13}$$

after which we compute $\mathbf{x}_2 = (\mathbf{I} - \mathbf{P}_1)\mathbf{w}$. The advantages of (13) are

- all the matrix operators on the left hand side are sparse or have a sparse representation;
- by using the relation $\mathbf{A}_2(\mathbf{I} - \mathbf{P}_1) = \mathbf{L}_{22}\mathbf{Q}_2$ we can precondition the matrix operator by focusing on $\mathbf{Q}_2$ or $\mathbf{L}_{22}$ separately, as described below;
- we have some freedom in the representation of $\mathbf{w}$, in the sense that (13) is an under-determined system such that any solution $\mathbf{w}$ leads to the same final solution $\mathbf{x}_2$.

We discuss this last point a little further. Consider applying a Krylov space method to the square but singular system

$$\mathbf{A}(\mathbf{I} - \mathbf{P}_1)\mathbf{w} = \begin{pmatrix} \mathbf{0} \\ \mathbf{A}_2(\mathbf{I} - \mathbf{P}_1) \end{pmatrix}\mathbf{w} = \begin{pmatrix} \mathbf{0} \\ \mathbf{r}_2 \end{pmatrix} \tag{14}$$

in which (13) is embedded. If we use Krylov space methods such as GMRES with an initial guess equal to zero, then all the solutions will be taken from the Krylov space

$$\left\{ \begin{pmatrix} \mathbf{0} \\ \mathbf{r}_2 \end{pmatrix}, \begin{pmatrix} \mathbf{0} \\ \mathbf{A}_2(\mathbf{I} - \mathbf{P}_1) \end{pmatrix}\begin{pmatrix} \mathbf{0} \\ \mathbf{r}_2 \end{pmatrix}, \begin{pmatrix} \mathbf{0} \\ \mathbf{A}_2(\mathbf{I} - \mathbf{P}_1) \end{pmatrix}^2\begin{pmatrix} \mathbf{0} \\ \mathbf{r}_2 \end{pmatrix}, \cdots \right\} \tag{15}$$

which has the form

$$\begin{pmatrix} 0 & 0 & \cdots \\ \star & \star & \cdots \end{pmatrix}. \tag{16}$$

This naturally leads to restricting the solutions $\mathbf{w}$ to the form

$$\mathbf{w} \triangleq \begin{pmatrix} \mathbf{0} \\ \mathbf{w}_2 \end{pmatrix}, \tag{17}$$

leading to the square set of equations

$$\mathbf{L}_{22}\mathbf{Q}_{22}\mathbf{w}_2 = \mathbf{r}_2 \tag{18}$$

This completely specifies the solution $\mathbf{w}$, but the numerical accuracy of the solution could suffer if the condition number of $\mathbf{Q}_{22}$ is too high. We will show below how one can completely eliminate the effect of $\mathbf{Q}_{22}$, but if $\mathbf{Q}_{22}$ is very badly conditioned, then some modification to the problem should be applied, such as a simple reordering of the columns in the original problem, without affecting the structural orthogonality among the rows. For the purposes of this paper, we will assume that $\mathbf{Q}_{22}$ is not singular or almost singular.

Of course, we remark that we do not form (18) explicitly, but we apply the operator of (18) in an iterative method by actually iterating with (14), restricting the iteration vectors to lie in a space of the form (16). When so restricted, (14) reduces to the iteration:

$$\begin{aligned} \mathbf{v}_2 \mapsto \mathbf{A}_2\mathbf{P}_2 \begin{pmatrix} \mathbf{0} \\ \mathbf{v}_2 \end{pmatrix} &= \mathbf{A}_2(\mathbf{I} - \mathbf{Q}_1^T\mathbf{Q}_1) \begin{pmatrix} \mathbf{0} \\ \mathbf{v}_2 \end{pmatrix} \\ &= \mathbf{A}_2 \begin{pmatrix} \mathbf{0} \\ \mathbf{v}_2 \end{pmatrix} - \mathbf{A}_2\mathbf{Q}_1^T\mathbf{Q}_1 \begin{pmatrix} \mathbf{0} \\ \mathbf{v}_2 \end{pmatrix} \\ &= \mathbf{A}_{22}\mathbf{v}_2 - \mathbf{A}_2\mathbf{Q}_1^T\mathbf{Q}_{12}\mathbf{v}_2 \\ &= (\mathbf{A}_{22} - \mathbf{A}_2\mathbf{Q}_1^T\mathbf{Q}_{12})\mathbf{v}_2. \end{aligned} \tag{19}$$

The last expression involves only the original data $\mathbf{A}_2$ and the sparse orthogonal factor $\mathbf{Q}_1$ derived from $\mathbf{A}_1$. If (19) is used in an iterative method, then it can be left in separate parts $\mathbf{v}_2 \mapsto \mathbf{A}_{22}\mathbf{v}_2 - \mathbf{A}_2\mathbf{Q}_1^T\mathbf{Q}_{12}\mathbf{v}_2$, but one could also explicitly form the matrix

$$\mathbf{A}_P \triangleq \mathbf{A}_{22} - \mathbf{A}_2\mathbf{Q}_1^T\mathbf{Q}_{12} \tag{20}$$

if one intends to use it in a direct method or apply an ILU-type of preconditioner. By a simple manipulation, one can verify that $\mathbf{A}_P = \mathbf{L}_{22}\mathbf{Q}_{22}$, so that we have indeed found a representation for (18) without computing $\mathbf{L}_{22}, \mathbf{Q}_{22}$ explicitly.

5 Relation to Schur Complement

The reader will notice the obvious similarity of the above development with the Schur complement. We write the inverse of $\mathbf{A}$ in terms of the LQ factorization (6):

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{Q}_{11}^T & \mathbf{Q}_{21}^T \\ \mathbf{Q}_{12}^T & \mathbf{Q}_{22}^T \end{pmatrix} \begin{pmatrix} \mathbf{L}_{11}^{-1} & \mathbf{0} \\ \times & \mathbf{L}_{22}^{-1} \end{pmatrix}, \tag{21}$$

where the "$\times$" block is irrelevant to this discussion. Letting $\mathbf{S}$ denote the Schur complement of $\mathbf{A}_{22}$, $\mathbf{S}^{-1}$ is the 2-2 block of $\mathbf{A}^{-1}$, and thus

$$\mathbf{S} = \mathbf{L}_{22}\mathbf{Q}_{22}^{-T} \tag{22}$$

This is in contrast to the operator $\mathbf{L}_{22}\mathbf{Q}_{22}$ in (18). For nonsymmetric operators, it is possible for $\mathbf{S}$ to have a higher condition number compared to the original matrix $\mathbf{A}$, and the same is true of the operator $\mathbf{L}_{22}\mathbf{Q}_{22}$. However, we will show below how to find a sparse right preconditioner to completely eliminate the effect of $\mathbf{Q}_{22}$. The remaining triangular part, $\mathbf{L}_{22}$, must have a condition number no larger than that of the original matrix $\mathbf{A}$, as can be seen by applying the eigenvalue interlacing theorem [5] to the symmetric matrix

$$\mathbf{Q}\mathbf{A}^T\mathbf{A}\mathbf{Q}^T = \begin{pmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{pmatrix}^T \begin{pmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{pmatrix} = \begin{pmatrix} \times & \times \\ \times & \mathbf{L}_{22}^T\mathbf{L}_{22} \end{pmatrix}. \tag{23}$$

A sparse left preconditioner can be used to help reduce the conditioning due to $\mathbf{L}_{22}$.

6 Preconditioning

We can completely eliminate the effect of $\mathbf{Q}_{22}$ from (18) (and hence from the iteration matrix $\mathbf{AP}_2$). We form a triangular neutralizing matrix $\mathbf{N}$ which is applied to the right of the operator $\mathbf{A}_P = \mathbf{L}_{22}\mathbf{Q}_{22}$; hence it is called a right preconditioner. The preconditioner is the Cholesky factor of a symmetric positive definite matrix derived from the partial LQ factorization:

$$\mathbf{N}^T\mathbf{N} = \mathbf{Q}_{22}^T\mathbf{Q}_{22} = \mathbf{I} - \mathbf{Q}_{12}^T\mathbf{Q}_{12}. \tag{24}$$

Then a simple computation:

$$(\mathbf{Q}_{22}\mathbf{N}^{-1})^T \cdot \mathbf{Q}_{22}\mathbf{N}^{-1} = \mathbf{N}^{-T}\mathbf{Q}_{22}^T\mathbf{Q}_{22}\mathbf{N}^{-1} = \mathbf{I}$$

shows that $\mathbf{Q}_{22}\mathbf{N}^{-1}$ is an orthogonal matrix. Hence the condition number of $\mathbf{L}_{22}\mathbf{Q}_{22}\mathbf{N}^{-1}$ is exactly the same as the condition number of $\mathbf{L}_{22}$. Furthermore, $\mathbf{N}$ is an upper triangular matrix which also inherits the block diagonal structure present in $\mathbf{Q}_{12}^T\mathbf{Q}_{12}$.

The operator used in actual computation is $\mathbf{A}_P$ (20). The preconditioned operator is therefore:

$$\mathbf{v}_2 \mapsto \mathbf{A}_p\mathbf{N}^{-1}\mathbf{v}_2 = (\mathbf{A}_{22} - \mathbf{A}_2\mathbf{Q}_1^T\mathbf{Q}_{12})\mathbf{N}^{-1}\mathbf{v}_2 \triangleq \mathbf{A}_{PN}\mathbf{v}_2. \tag{25}$$

where it is implicit that the vector $\mathbf{N}^{-1}\mathbf{v}_2$ is computed by solving the triangular system involving $\mathbf{N}$. Since $\textsc{cond}\,\mathbf{A}_{PN} = \textsc{cond}\,\mathbf{L}_{22}$, (23) leads to the following simple result:

Theorem. *Given the LQ factorization (6) of an arbitrary square matrix* $\mathbf{A}$ *and the operator* $\mathbf{A}_{PN}$ *defined in (25) with* $\mathbf{N}$ *defined in (24), the 2-norm condition number of* $\mathbf{A}_{PN}$ *is guaranteed to be less than or equal to the 2-norm condition number of the original matrix* $\mathbf{A}$.

One may apply a left preconditioner of the user's choice to $\mathbf{A}_{PN}$. The rows in this operator represent the boundary nodes in the subgraphs on the original grid. One way to construct a left preconditioner is to extract the rows (equations) corresponding to the boundary nodes in each subgraph separately, and to compute an LQ factorization of these rows alone, repeating this process for each subgraph. The resulting "L" factors for each subgraph are then assembled into a block diagonal left preconditioner, which we label $\mathbf{M}_1$. This preconditioner corresponds to a block Jacobi-type preconditioner, and is a natural choice if the equations for each subgraph were distributed to different processors, as each subgraph can be handled independently.

One can refine this preconditioner by first orthogonalizing the boundary equations within each subgraph against the interior equations for the same subgraph and then carrying out the above local LQ factorization. This preconditioner, which we label $\mathbf{M}_2$, can be motivated as follows. Let $\tilde{\mathbf{A}}_1^g$ represent the rows from the interior of a given subgraph g, and $\tilde{\mathbf{A}}_2^g$ represent the rows from the boundary of the same subgraph, and consider the local LQ factorization corresponding to this subgraph:

$$\tilde{\mathbf{A}}^g \equiv \begin{pmatrix} \tilde{\mathbf{A}}_1^g \\ \tilde{\mathbf{A}}_2^g \end{pmatrix} \quad = \quad \tilde{\mathbf{L}}^g\tilde{\mathbf{Q}}^g \equiv \begin{pmatrix} \tilde{\mathbf{L}}_{11}^g & \mathbf{0} \\ \tilde{\mathbf{L}}_{21}^g & \tilde{\mathbf{L}}_{22}^g \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{Q}}_1^g \\ \tilde{\mathbf{Q}}_2^g \end{pmatrix}.$$

The block $\tilde{\mathbf{L}}_{22}^g$ is used as a preconditioner for the rows $\tilde{\mathbf{A}}_2^g$. All the $\tilde{\mathbf{L}}_{22}^g$ blocks are then assembled into a block diagonal preconditioner

$$\mathbf{M}_2 = \textsc{diag}\{\tilde{\mathbf{L}}_{22}^g\}_{g=1,\ldots,p}.$$

Note that the blocks $\tilde{\mathbf{L}}_{11}^g$ and $\tilde{\mathbf{Q}}_1^g$ corresponding to the interior nodes have already been computed as part of (7), so here it is necessary only to extend that LQ factorization. This preconditioner will also be block diagonal, but will have more fill than $\mathbf{M}_1$. Further study of the theoretical behavior of these preconditioners is needed and will be reported in a future paper, but we do give an indication of how they affected the conditioning on some specific numerical examples given in Sec. 7.

An alternate choice of left preconditioner could be obtained by assembling the operator $\mathbf{A}_{PN}$, which is often itself sparse, and then applying an off-the-shelf preconditioner such as Incomplete LU [6]. The pros and cons of different left preconditioners, which can have a dramatic effect on the convergence of the overall algorithm, is beyond the scope of this short paper, and is the subject of further study.

7 Example

We illustrate how the sparse structure in $\mathbf{A}$ carries over the the computed $\mathbf{L}_{11}$ and $\mathbf{Q}_1$. As a first example, we take a matrix arising from a simple 5 point finite-difference stencil on a 2D grid in Fig. 1, after reordering according to Sec. 2, using 4 subgraphs. Fig. 2 shows the structure of the resulting LQ factors, which preserve the block diagonal structure. We remark that the entire $\mathbf{Q}_1$ is needed for the LQ factorization, for the computation of $\mathbf{x}_1$ and for the computation of $\mathbf{N}$. After that, $\mathbf{Q}_1$ appears only in the operator $\mathbf{A}_P$ buried within the matrix product $\mathbf{A}_2\mathbf{Q}_1^T\mathbf{Q}_{12}$. Because of the sparsity structure of $\mathbf{A}_2$ and of $\mathbf{Q}_{12}$, many nonzero entries in $\mathbf{Q}_1$ are multiplied by only zero entries in $\mathbf{A}_2, \mathbf{Q}_{12}$ and hence can be dropped. The elements of $\mathbf{L}_{11}$ are needed to compute the LQ factorization, to solve for $\mathbf{x}_1$, and to form the preconditioner $\mathbf{M}_2$, but can be dropped after that. All these entries that are dropped after $\mathbf{x}_1, \mathbf{N}, \mathbf{M}_2$ have been computed are colored gray in Fig. 2b. So during the iterative solution of preconditioned system (28), only the black elements of Fig. 2b must be retained. A detailed analysis of the exact memory requirements depends critically on the underlying structure of the equations being solved as well as on the specific ordering used.

In Table 1, we summarize the effect of the LQ factorization approach on the condition number for two typical examples. The FIDAP004 matrix is a symmetric indefinite matrix taken from the Harwell-Boeing matrix collection [3] and the RAEFSKY1 matrix is a nonsymmetric matrix taken from [2]. Using a partitioner based on the multi-node level-set expansion algorithm [4, 6], the results of Table 1 are enough to show certain properties enjoyed by the LQ-Schur method.

- The memory requirements during the LQ factorization stage can be considerable, but once one reaches the iteration stage, one can drop many of the elements substantially reducing the memory requirements. But the memory requirements during the LQ factorization stage can be mitigated by using the fact that the LQ factorization for each subgraph can be computed separately.

- The conditioning of the reduced order operator $\mathbf{A}_{PN}$ can be noticeably less than that of the entire matrix operator, and in some cases less than that of the Schur complement. One can also note that the left preconditioners can be effective in further reducing the

matrix name	FIDAP004	RAEFSKY1
size	1601	3242
number of partitions	4	2
size of $\mathbf{L}_{22}$ block	416	954
nonzeros in $\mathbf{A}$	1.2%	2.8%
nonzeros in $\mathbf{L}_{11}$	9.2%	20.7%
nonzeros in $\mathbf{Q}_1$	13.4%	31.0%
in last stage	2.9%	8.7%
nonzeros in $\mathbf{N}$	4.8%	24.3%
nonzeros in $\mathbf{M}_1$	4.7%	15.1%
nonzeros in $\mathbf{M}_2$	12.8%	25.0%
$\textsc{cond}\,\mathbf{A}$	2.39×10^3	1.29×10^4
$\textsc{cond}\,\mathbf{S}$	6.75×10^3	4.27×10^3
$\textsc{cond}\,\mathbf{A}_P$	9.58×10^4	2.04×10^4
$\textsc{cond}\,\mathbf{A}_{PN}$	1.05×10^3	6.47×10^3
$\textsc{cond}\,\mathbf{M}_1^{-1}\mathbf{A}_{PN}$	3.14×10^2	1.47×10^3
$\textsc{cond}\,\mathbf{M}_2^{-1}\mathbf{A}_{PN}$	2.23×10^2	1.08×10^3

Table 1: Summary of method results on two examples, showing how the condition number can be reduced in some cases, while maintaining the sparse representation of the operators.

condition number. On the other hand, it is clear that $\mathbf{A}_P$ can have a higher condition number than even the original matrix, so it is essential to use the right preconditioner $\mathbf{N}$ to eliminate the effect of $\mathbf{Q}_{22}$ in (18).

Both of these properties are critically dependent on the specific choice of partitioner and on the success of finding a good ordering that exposes a sequence of mutually structurally orthogonal rows in the matrix operator. In fact, the partitioner we used was not chosen with these specific examples in mind, as this was not the focus of this paper. The partitioning shown here, however, is sufficient to show some of the properties of the LQ-Schur based projection.

8 Summary

We summarize the process one would use to solve a given system $\mathbf{Ax} = \mathbf{b}$.

1. Compute a partitioning and an associated ordering of the matrix $\mathbf{A}$ using any method so that the result satisfies the assumptions of Sec. 2. The result for a matrix derived from a 2D 5-point finite-difference stencil is illustrated in Fig. 1.

2. Compute a partial LQ factorization of $\mathbf{A}$. The result is illustrated in Fig. 2. Solve for partial solution $\mathbf{x}_1$ via (9) and residual $\mathbf{r}_2 \triangleq \mathbf{b}_2 - \mathbf{A}_2\mathbf{x}_1$ (10).

3. Collect the quantities $\mathbf{A}_2$, $\mathbf{Q}_1$ needed to form or apply the operator $\mathbf{A}_{PN}$ and right preconditioner $\mathbf{N}$.

Then the unpreconditioned system to be solved is

$$\mathbf{A}_P\mathbf{w}_2 = \mathbf{r}_2, \tag{26}$$

where the solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ is then

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 = \mathbf{x}_1 + (\mathbf{I} - \mathbf{Q}_1^T\mathbf{Q}_1)\begin{pmatrix}\mathbf{0}\\ \mathbf{w}_2\end{pmatrix} \tag{27}$$

4. Compute the right preconditioner $\mathbf{N}$ via (24). Form the operator $\mathbf{A}_{PN}$ (25) either implicitly or explicitly, and optionally compute a suitable left preconditioner $\mathbf{M}$,

5. Solve the preconditioned system

$$\mathbf{M}^{-1}\mathbf{A}_{PN}\mathbf{y}_2 = \mathbf{M}^{-1}\mathbf{r}_2 \tag{28}$$

using an iterative method, where the solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ is then

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 = \mathbf{x}_1 + (\mathbf{I} - \mathbf{Q}_1^T\mathbf{Q}_1)\begin{pmatrix}\mathbf{0}\\ \mathbf{N}^{-1}\mathbf{y}_2\end{pmatrix} \tag{29}$$

From this summary, it is clear that there is a high degree of overlap with Schur complement methods. But Schur complement depends on a partial Gaussian elimination of the original operator in which one must trade off pivoting for numerical stability against ordering for maximum sparsity. For nonsymmetric operators, it can also happen that the condition number of the reduced order operator will exceed that of the original. On the other hand, using an LQ factorization combined with the right preconditioner, we can bound the condition of the resulting operator to be at most that of the original matrix, and often it is much less. Though it has not been discussed here, one is also free to order the columns of the original matrix to lower the condition number of the intermediate operators $\mathbf{N}$, etc. In this sense, the method based on the LQ factorization enjoys many of the advantages of solving a nonsymmetric system via the normal equations, but without suffering the squaring of the condition number.

The main limitations of the LQ factorization approach are the increased memory requirements in order to compute even the partial factors. In a practical implementation, one can mitigate this problem to some extent by computing the LQ factorization and partial solution $\mathbf{x}_1$ for each subgraph separately, saving only the items needed for the later iterative process. But the necessary bookkeeping can become rather complicated. We have not investigated the possibility of keeping $\mathbf{Q}_1$ in some sort of factored form.

References

[1] T. F. Chan and T. P. Mathew. Domain decomposition algorithms. In *Acta Numerica*, pages 61–143. Cambridge Univ. Press, 1994.

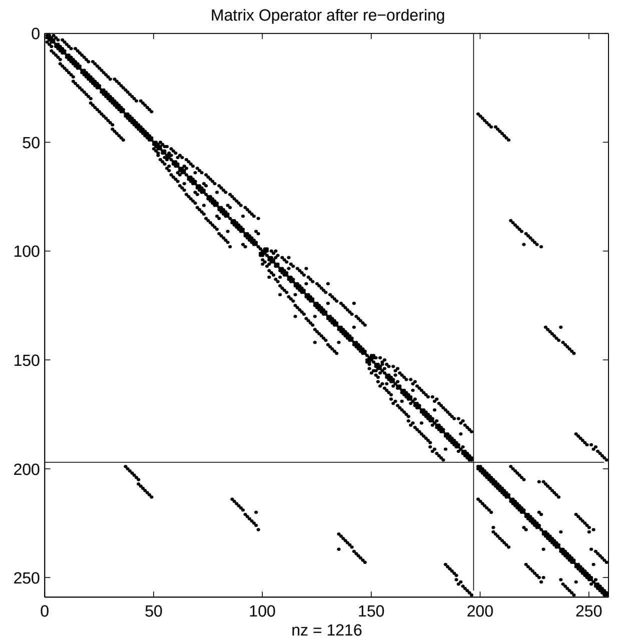
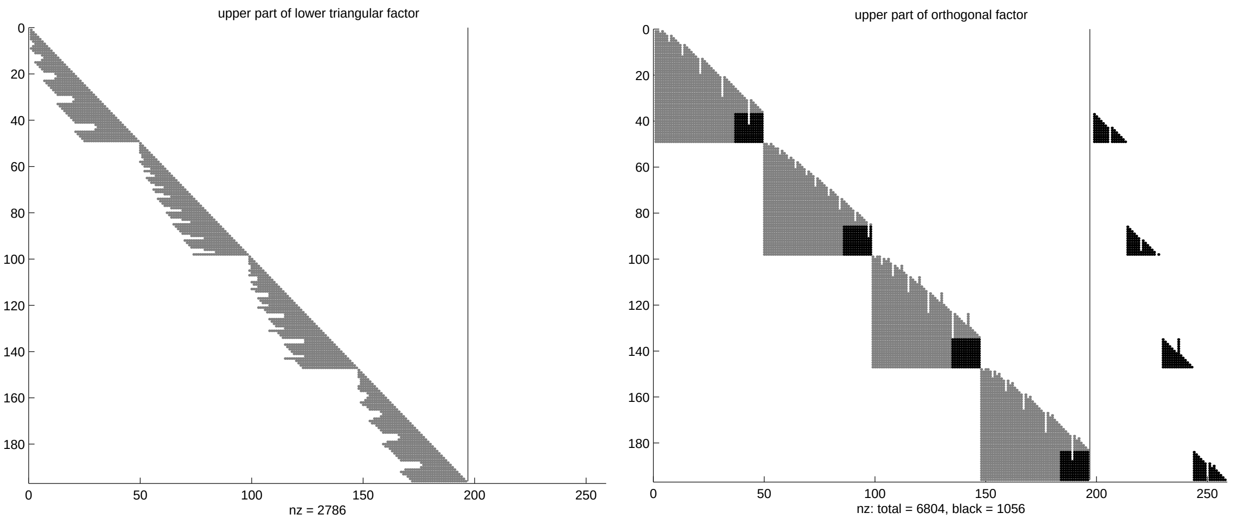


Figure 1: Matrix example based on a 2D 5-point finite-difference stencil after reordering into 4 subgraphs.



(a) L from LQ factor

(b) Q from LQ factor

Figure 2: LQ factors for the matrix in Fig. 1. The gray entries can be dropped during the final stage while computing $\mathbf{x}_2$

[2] T. Davis. University of Florida sparse matrix collection. http://www.cise.ufl.edu/∼davis /sparse/, ftp://ftp.cise.ufl.edu/pub/faculty/davis/matrices. See NA Digest vol. 97, no. 23, June 7, 1997.

[3] I. Duff, R. Grimes, and J. Lewis. Harwell-Boeing sparse matrix collection. http: //math.nist.gov/MatrixMarket/collections/hb.html.

[4] T. Goehring and Y. Saad. Heuristic algorithms for automatic graph partitioning. tech. report UMSI 94-29, Univ. of Minn. Supercomputer Inst., 1994.

[5] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins Univ. Press, 3rd edition, 1996.

[6] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS, 1996.

[7] B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition, Paralllel Methods for Elliptic Partial Differential Equations.* Cambridge Univ. Press, 1996.