Journal of Computational and Applied Mathematics 486 (2026) 117685

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Exact orthogonalization of integer matrices

Thomas R Bewley

UC San Diego, La Jolla, CA, and USAF Academy, Colorado Springs, CO, United States

ARTICLE INFO

Keywords:
Integer matrices
Fundamental matrix subspaces
QR decomposition
Modified Gram Schmidt

ABSTRACT

Define an "integer matrix" as a matrix with integer elements. This paper develops and tests an algorithm, dubbed Integer Gram-Schmidt (IGS), that for any integer matrix $A_{m,n}$ of rank r generates the exact matrix decomposition $A = QD^{-1}R$, where $\{Q, D, R\}$ themselves are also integer matrices, and where the r columns of Q orthogonally span the column space of A, D is diagonal, and R is upper triangular. IGS also, optionally, generates an integer matrix L, the $m-r$ columns of which exactly span the left nullspace of A. For matrices of sufficiently small dimension and with individual entries of sufficiently small magnitude, as characterized herein, $QD^{-1}R$ decompositions can be computed using `int64` arithmetic only; larger problems require the use of `int128`, `int256`, or variable-precision arithmetic to complete. Applying IGS to A^T, of course, generates corresponding exact integer orthogonal bases for the row space and nullspace of A. IGS is a natural modification of the Modified Gram-Schmidt (MGS) procedure, with at each step each subsequent column of Q scaled by the inverse of its greatest common factor (GCF), rather than its 2-norm, in order to minimize the magnitude of the integers in each column of Q. As its main use is to develop integer orthogonal bases for the four fundamental subspaces of integer matrices, we also develop a Pivoted IGS algorithm generating $A\Pi = QD^{-1}R$, where Π is a permutation matrix selected to further reduce the 2-norm of the resulting columns of Q.

1. Main result

Given any real matrix $A_{m,n}$ of rank r, the Modified Gram Schmidt (MGS) algorithm (Section 5.2.8 of [1]) computes a real matrix decomposition $A = QR$, where $\{Q, R\}$ are real matrices, the r columns of $Q_{m,r}$ orthogonally span the column space of A, $Q^TQ = I$, and $R_{r,m}$ is in row echelon form.

Given any *integer* matrix $A_{m,n}$ of rank r, we can instead write an exact *integer* matrix decomposition $A = QD^{-1}R$, where $\{Q, D, R\}$ are integer matrices, the r columns of $Q_{m,r}$ orthogonally span the column space of A, $D_{r,r} = Q^TQ$ is diagonal, and $R_{r,m}$ is upper triangular; we can also form an integer matrix $L_{m,m-r}$, the $m-r$ columns of which orthogonally[1] span the left nullspace of A. As just one example with $m = 5$, $n = 3$, and $r = 3$, we may write such a decomposition as

$$A = \begin{pmatrix} -3 & 3 & 1 \\ 4 & 1 & -3 \\ 4 & -2 & 1 \\ -2 & -2 & 2 \\ -2 & 2 & -3 \end{pmatrix}, \quad Q = \begin{pmatrix} -3 & 108 & 654 \\ 4 & 101 & -202 \\ 4 & -46 & 305 \\ -2 & -124 & -100 \\ -2 & 72 & -675 \end{pmatrix}, \quad L = \begin{pmatrix} 234 & 0 \\ 218 & 10 \\ 275 & -11 \\ 410 & 7 \\ 225 & -9 \end{pmatrix}, \tag{1a}$$

E-mail address: tbewley@ucsd.edu

[1] For the purpose of this paper, "orthogonality" of a set of integer vectors implies solely that the dot product between any two different vectors in the set is zero; said vectors are not (as is often customary) also assumed to be of unit length.

https://doi.org/10.1016/j.cam.2026.117685
Received 5 February 2025; Received in revised form 12 March 2026
Available online 8 April 2026

$$D = \begin{pmatrix} 49 & 0 & 0 \\ 0 & 44,541 & 0 \\ 0 & 0 & 1027170 \end{pmatrix}, \quad R = \begin{pmatrix} 49 & -13 & -9 \\ 0 & 909 & -705 \\ 0 & 0 & 3390 \end{pmatrix}. \tag{1b}$$

When such a decomposition is computed correctly, $Q^T Q$ and $L^T L$ are diagonal, $Q^T L$ and $A^T L$ are zero, and of course $A = QD^{-1}R$. The Integer Gram Schmidt (IGS) scheme developed in this paper, which is provided in executable Matlab syntax in Algorithm 1, generates such decompositions exactly, using integer arithmetic only, and is thus prone to overflow (unless variable-precision arithmetic is used), but is not prone to round-off error. As seen in Algorithm 1, IGS consists of eight main steps:

Algorithm 1 The Integer Gram-Schmidt (IGS) algorithm, in executable Matlab syntax.

```
function [Q,D,R,r,L] = IGS(A)
% Copyright 2025 by Thomas Bewley, published under BSD 3-Clause License.
[m,n]=size(A); Q=int64(A); % Convert to integers (all math below done on integers!)
for i=1:n                  % orthogonalize the columns of Q
  Q(:,i)=Q(:,i)/gcd_vec(Q(:,i)); f(i)=dot_product(Q(:,i),Q(:,i));
  if f(i)>0, for j=i+1:n;
    Q(:,j)=f(i)*Q(:,j)-Q(:,i)*dot_product(Q(:,i),Q(:,j));
  end, end
end
index=[1:n]; for i=1:n     % strip out the zero columns of Q
  if f(i)==0, l=length(index);
    for j=1:l, if index(j)==i
      index=index([1:j-1,j+1:l]); break
    end, end
  end
end, Q=Q(:,index); f=f(index); r=length(index);
L=int64(eye(m)); for j=1:r % orthogonalize columns of L against Q
  for i=1:m
    L(:,i)=f(j)*L(:,i)-Q(:,j)*dot_product(Q(:,j),L(:,i));
    L(:,i)=L(:,i)/gcd_vec(L(:,i));
  end
end
for j=1:m                  % orthogonalize the columns of L
  h(j)=dot_product(L(:,j),L(:,j));
  for i=j+1:m
    L(:,i)=h(j)*L(:,i)-L(:,j)*dot_product(L(:,j),L(:,i));
    L(:,i)=L(:,i)/gcd_vec(L(:,i));
  end
end
index=[1:m]; for i=1:m     % strip out the zero columns of L
  if dot_product(L(:,i),L(:,i))==0, l=length(index);
    for j=1:l, if index(j)==i
      index=index([1:j-1,j+1:l]); break
    end, end
  end
end, L=L(:,index);
Q=double(Q); L=double(L); % convert back to double (Matlab default)
R=Q'*A; D=Q'*Q;           % generate R and D
end % function IGS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [p]=dot_product(u,v)
p=0; for i=1:length(u), p=p+u(i)*v(i); end
end
function [g]=gcd_vec(u)
g=gcd(u(1),u(2)); for i=3:length(u), g=gcd(g,u(i)); end
end
```

'1.) initialize $Q = A$, scaling each column by the inverse of its GCF;
2.) orthogonalize the columns of Q, and again scale each column by the inverse of its GCF;
3.) strip out the resulting of zero columns of Q;
4.) initialize $L = I_{m\times m}$;
5.) orthogonalize the columns of L against the columns of Q;
6.) orthogonalize the columns of L, scaling each column by the inverse of its GCF;
7.) strip out the resulting of zero columns of L; and

8.) generate $R = Q^T A$ and $D = Q^T Q$.

Some observations related to the above steps:

(2a) Rather than normalizing the columns of Q, which would convert them to real vectors, we instead divide each column of Q by its GCF, thus minimizing the magnitudes of its integer entries.

(2b) The i'th column of Q, denoted here $\mathbf{q}^i$, is not normalized, so in general $f_i = \mathbf{q}^i \cdot \mathbf{q}^i \neq 1$. Thus, rather than updating $\mathbf{q}^j \leftarrow \mathbf{q}^j - (\mathbf{q}^i \cdot \mathbf{q}^j)\,\mathbf{q}^i$ as in MGS, we effectively scale the RHS of this update by f_i, instead performing the update $\mathbf{q}^j \leftarrow f_i\,\mathbf{q}^j - (\mathbf{q}^i \cdot \mathbf{q}^j)\,\mathbf{q}^i$, which keeps the entries of $\mathbf{q}^j$ as integers after the update, while projecting the vector $\mathbf{q}^j$ in the same direction as does the standard MGS update.

(8) Rather than computing R and D while orthogonalizing the columns of Q, it is simplest to just compute them after the fact, leveraging the identity $Q^T(A = QD^{-1}R) \;\Rightarrow (Q^T A) = (Q^T Q)D^{-1}R$, where by orthogonality $D \triangleq Q^T Q$ is diagonal, and $R \triangleq Q^T A$ is upper triangular.

The rest of the steps of the Integer Gram-Schmidt algorithm are self explanatory.

2. Discussion

An exact integer $QD^{-1}R$ decomposition may be rewritten a few different ways:

- as $A = Q_1 R$ where $Q_1 = QD^{-1}$ is Q with its columns scaled by the diagonal elements of D^{-1};
- as $A = Q\,R_1$ where $R_1 = D^{-1}R$ is R with its rows scaled by the diagonal elements of D^{-1};
- as $A = Q_2 R_2$ where $Q_2 = QD^{-1/2}$ and $R_2 = D^{-1/2}R$.

The rational expressions for Q_1 and R_1 above are, of course, also exact, as are the expressions for Q_2 and R_2, before the (real) divisions and square roots are calculated. Note that the third form above, with $Q_2^T Q_2 = I$, is equivalent a standard (real) QR decomposition of A.

Note also that an integer $QD^{-1}R$ decomposition of an integer matrix A can sometimes be formed by taking a standard (real) QR decomposition of A, then attempting to express the (real) elements of Q as rational expressions. This approach, however, is highly susceptible to error due to the finite-precision arithmetic involved. IGS, on the other hand, is based on integer arithmetic only, and is thus not susceptible such errors.

The complexity of the well-known Modified Gram-Schmidt (MGS) algorithm is $O(m\,n^2)$ floating point operations (flops), and is dominated by its first set of nested loops (related to the orthogonalization of Q). The new Integer Gram-Schmidt (IGS) algorithm, which follows an entirely analogous structure as the (real) MGS algorithm, but carefully modifies the updates to keep all arithmetic amongst the integers, is thus $O(m\,n^2)$ integer operations (iops).

As quantified in Section 5, the magnitudes of the elements of an integer $QD^{-1}R$ decomposition generally grow as m and n and the magnitude of the integer elements of A grow, though this dependence is not at all smooth. Variable precision integer arithmetic can easily be implemented to address this, starting with 64-bit integers and increasing to 128-bit integers, etc, as the need arises in the IGS computations.

Integer and rational matrices of the type discussed above play a valuable role in dynamical systems theory and crystallography [2,3].

Integer $QD^{-1}R$ decompositions are also quite valuable in the pedagogical setting, when introducing orthogonal bases of the four fundamental subspaces of a matrix A, as illustrated graphically by the celebrated "Strang Plot" [4], leveraging simple classroom example matrices with integer elements.

There has been significant previous work in the generation of rational orthogonal matrices (see, e.g., [5], and the references contained therein). To the best of our knowledge, the integer $QD^{-1}R$ decomposition in the unpivoted (Section 1) and pivoted (Section 3) forms, and the natural modification of the Modified Gram-Schmidt (MGS) procedure identified herein which exactly generates them, had not previous been discovered and studied, and may well be useful in such settings.

3. Pivoting the integer $QD^{-1}R$ decomposition

One of the primary intended purposes of the IGS algorithm developed in this work is to develop orthogonal integer bases of the four fundamental subspaces defined by an integer matrix A. In an effort to keep the elements of Q as small as possible, an effective approach is to modify the algorithm proposed in Section 1 by reordering the columns of Q during step 2 in order to put first those (nonzero) columns of Q with the smallest 2-norms, against which the subsequent columns of Q are orthogonalized via MGS-like updates. The column swaps involved are easily tracked in a corresponding "enumeration vector" π; that is, if the i'th element of the vector π is j, then the i'th column of Q is simply the j'th column of A orthogonalized against the earlier columns of Q and scaled by the inverse of its GCF. Defining the i'th column of the corresponding permutation matrix Π as the j'th column of the identiy matrix where $j = \pi(i)$ [in code, `Pi=zeros(n,n); for i=1:n, Pi(pi(i),i)=1; end`], the corresponding pivoted decomposition is simply $A\,\Pi = QD^{-1}R$.

4. Sensitivity of integer $QD^{-1}R$ decompositions

As in the case of the QR decomposition of real matrices, the columns of Q in the integer $QD^{-1}R$ decomposition orthogonally span the real vector space defined by the columns of A.

However, the peculiar restriction that all of the elements of Q (and, of $\{D, R\}$) must actually be integers in an integer $QD^{-1}R$ decomposition, with or without pivoting, results in a nonmonotonic and (perhaps) surprisingly strong sensitivity in the magnitude of the elements of $\{Q, D, R\}$ in response to small changes in the individual elements of A.

To illustrate, consider the two matrices

$$A_1 = \begin{pmatrix} 1 & 2 & 2 \\ -2 & 2 & 2 \\ 1 & -2 & -2 \\ -2 & 2 & -2 \\ 2 & 1 & -2 \end{pmatrix}, \qquad A_2 = \begin{pmatrix} 1 & 2 & 2 \\ -2 & 2 & 2 \\ 1 & -2 & -1 \\ -2 & 2 & -2 \\ 2 & 1 & -2 \end{pmatrix}. \tag{2a}$$

Note that A_1 and A_2 differ in only a single element, as highlighted in red. Defining the permutation matrix $\Pi(\pi)$ as in Section 3, the optimized $A_1\,\Pi = QD^{-1}R$ decomposition is given by

$$Q = \begin{pmatrix} 1 & 7 & 21 \\ 1 & -8 & 9 \\ -1 & 3 & -13 \\ -1 & -12 & 19 \\ -1 & 8 & 24 \end{pmatrix}, \quad D = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 330 & 0 \\ 0 & 0 & 1628 \end{pmatrix}, \quad R = \begin{pmatrix} 10 & -2 & 3 \\ 0 & 66 & -24 \\ 0 & 0 & 148 \end{pmatrix}, \quad \pi = \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix}. \tag{2b}$$

whereas the optimized $A_2\,\Pi = QD^{-1}R$ decomposition is given by

$$Q = \begin{pmatrix} 1 & 17 & 355 \\ -2 & 8 & 274 \\ 1 & -11 & -99 \\ -2 & 8 & -534 \\ 2 & 13 & -388 \end{pmatrix}, \quad D = \begin{pmatrix} 14 & 0 & 0 \\ 0 & 707 & 0 \\ 0 & 0 & 646602 \end{pmatrix}, \quad R = \begin{pmatrix} 14 & -6 & -3 \\ 0 & 101 & 19 \\ 0 & 0 & 3201 \end{pmatrix}, \quad \pi = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}. \tag{2c}$$

That is, in response to a small change to a single element of A_1, all three columns of the corresponding Q have changed, and the 2-norm of the largest column of Q has increased by a factor of over 20.

This peculiar nonsmooth behavior precludes the use of traditional numerical analysis techniques to bound the magnitude of the individual integer elements of Q as the magnitude of the elements of A, and the dimension of A, are changed; any such analysical bounds that might be attempted would be so conservative that they would be of quite limited value in practice. We thus resort, in Section 5, to examining the $QD^{-1}R$ decompositions of tens of thousands of example (randomly-generated) integer matrices.

5. Examples of pivoted integer $QD^{-1}R$ decompositions

If the dimensions and magnitudes of the individual elements of A are all sufficiently small, the entire Integer Gram-Schmidt calculation proposed in this work can be completed reliably using `int64` arithmetic. Increasing the dimensions and/or magnitudes of the individual elements of the matrices of interest ultimately causes integer overflows, and necessitates the use of `int128` or variable-precision artithemetic (automatically switching integer representations as necessary during the IGS computation). [Note that, for the applications of interest (see Section 2), such higher-dimensional cases might actually be of somewhat reduced interest.]

For example, selecting tall matrices A with $m = 5$ rows and $n = 3$ columns and individual integer elements randomly generated on the $[-2, 2]$ interval, as illustrated in (2) above, the IGS algoroithm developed here has been run tens of thousands of times without overflowing `int64` representations.

Similarly, selecting tall matrices A with $m = 7$ rows and $n = 3$ columns and individual integer elements randomly generated on the $[-1, 1]$ interval, as illustrated in Eq. (3) below, the IGS algoroithm has again been run tens of thousands of times without overflowing `int64` representations:

$$A = \begin{pmatrix} -1 & -1 & 0 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & -1 \end{pmatrix}, \quad Q = \begin{pmatrix} 0 & -4 & -9 \\ 1 & -3 & -21 \\ -1 & -5 & 3 \\ -1 & 3 & -17 \\ 0 & 0 & -19 \\ 0 & 4 & -10 \\ -1 & -1 & -7 \end{pmatrix}, \quad L = \begin{pmatrix} 0 & 0 & 0 & 51 \\ 0 & 0 & 3 & -21 \\ 1 & 0 & 0 & -17 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & -6 & -9 \\ 1 & -1 & 1 & 10 \\ -1 & -1 & 1 & -7 \end{pmatrix},$$
$$D = \begin{pmatrix} 4 & 0 & 0 \\ 0 & 76 & 0 \\ 0 & 0 & 1330 \end{pmatrix}, \quad R = \begin{pmatrix} 4 & -1 & 2 \\ 0 & 19 & 10 \\ 0 & 0 & 70 \end{pmatrix}, \quad \pi = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}. \tag{3}$$

In the above example, Q has $r = 3$ columns where r is the rank of A, and L has $m - r = 4$ columns. Similarly, selecting square matrices A with $m = 4$ rows and $n = 4$ columns and individual integer elements randomly generated on the $[-2, 2]$ interval, as illustrated in

Eq. (4) below, the IGS algoroithm has again been run tens of thousands of times without overflowing `int64` representations:

$$A=\begin{pmatrix}0&0&2&-2\\-2&-2&-1&-2\\-1&0&1&2\\-2&0&1&-2\end{pmatrix},\quad Q=\begin{pmatrix}0&-1&1&3\\-1&0&0&0\\0&1&-4&2\\0&-1&-5&-1\end{pmatrix},\quad L=[],\quad r=4,$$
$$D=\begin{pmatrix}1&0&0&0\\0&3&0&0\\0&0&42&0\\0&0&0&14\end{pmatrix},\quad R=\begin{pmatrix}2&2&2&1\\0&6&1&-2\\0&0&14&-7\\0&0&0&7\end{pmatrix},\quad \pi=\begin{pmatrix}2\\4\\1\\3\end{pmatrix}. \tag{4}$$

Similarly, selecting fat matrices A with $m=5$ rows and $n=10$ columns and individual integer elements randomly generated on the $[-6,6]$ interval, as illustrated below (with $L=[]$ and $r=5$), the IGS algoroithm has again been run tens of thousands of times without overflowing `int64` representations:

$$A=\begin{pmatrix}-6&4&2&-3&-3&4&-3&3&-6&5\\1&1&-6&4&-4&1&-5&3&-5&-4\\-1&3&-6&-5&1&-3&-2&-1&2&3\\-5&-3&6&-2&-1&6&1&-5&5&-5\\-6&-6&-6&-6&-4&-3&0&6&5&-1\end{pmatrix},\quad Q=\begin{pmatrix}1&3&-801&-3783&-105\\-3&4&-150&3438&6\\-3&0&-187&-3321&191\\3&3&1001&-801&97\\-3&0&1071&-2179&-135\end{pmatrix},$$
$$R=\begin{pmatrix}74&37&21&12&15&-3&1&-36&3&-4\\0&34&-26&1&-28&-29&7&6&-23&-16\\0&0&4528&-5690&-2469&-6588&-13344&-1245&15,542&-10042\\0&0&0&56,382&3793&46,536&-6180&-6783&-16034&-36446\\0&0&0&0&925&770&678&-1783&792&-326\end{pmatrix}, \tag{5}$$
$$D=\operatorname{diag}(\begin{bmatrix}37&34&2,848,112&42,549,616&75176\end{bmatrix}),\quad \pi=\begin{pmatrix}3&6&7&4&5&1&2&8&9&10\end{pmatrix}'.$$

The validity of the IGS algorithm is confirmed in the unpivoted example (1) [with $\Pi=I$], and the pivoted examples Eq. (3) through Eq. (5) [with Π as given in Section 3], and in tens of thousands of similar randomly-generated test cases of the same size, simply by verifying in each such test that Q^TQ and L^TL are diagonal, that Q^TL and A^TL are zero, and of course that $A\Pi=QD^{-1}R$. All such randomly-generated examples complete without overflow using `int64` arithmetic, and pass such tests for correctness; increasing the magnitude of the entries of A or the dimension of A sometimes requires the use of `int128` or variable-precision integer representations to achieve such correct results.

CRediT authorship contribution statement

Thomas R Bewley: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] G.H. Golub, C.F. van Loan, Matrix Computations, Third Edition, Johns Hopkins, 1996.
[2] T. Kaczorek, Polynomial and Rational Matrices: Applications in Dynamical Systems Theory, Springer, London, 2007.
[3] M.A. Rodrıguez-Andrade, G. Aragón-González, J.L. Aragón, A. Gómez-Rodrıguez, Coincidence lattices in the hyperbolic plane, Acta Crystallogr. A 67 (2011) 35–44.
[4] G. Strang, Linear Algebra and its Applications Cengage, 4th edition, 2006.
[5] M.A. Rodrıguez-Andrade, G. Aragón-González, JL Aragón The generation of all rational orthogonal matrices in Rp,q, Linear Algebra Appl. 496 (2016) 101–113.