ELSEVIER Linear Algebra and its Applications 310 (2000) 25–42

LINEAR ALGEBRA AND ITS APPLICATIONS

www.elsevier.com/locate/laa

A modified Gram–Schmidt algorithm with iterative orthogonalization and column pivoting

Achiya Dax

Hydrological Service, P.O. Box 6381, 91063 Jerusalem, Israel

Received 4 July 1999; accepted 27 December 1999

Submitted by B.N. Parlett

Abstract

Iterative orthogonalization is aimed to ensure small deviation from orthogonality in the Gram–Schmidt process. Former applications of this technique are restricted to classical Gram–Schmidt (CGS) and column-oriented modified Gram–Schmidt (MGS). The major aim of this paper is to explain how iterative orthogonalization is incorporated into row-oriented MGS. The interest that we have in a row-oriented iterative MGS comes from the observation that this method is capable of performing column pivoting. The use of column pivoting delays the deteriorating effects of rounding errors and helps to handle rank-deficient least-squares problems.

A second modification proposed in this paper considers the use of Gram–Schmidt *QR* factorization for solving linear least-squares problems. The standard solution method is based on one orthogonalization of the r.h.s. vector **b** against the columns of Q. The outcome of this process is the residual vector, $\mathbf{r}^*$, and the solution vector, $\mathbf{x}^*$. The modified scheme is a natural extension of the standard solution method that allows it to apply iterative orthogonalization. This feature ensures accurate computation of small residuals and helps in cases when Q has some deviation from orthogonality. © 2000 Elsevier Science Inc. All rights reserved.

Keywords: Gram–Schmidt orthogonalization; Row-oriented MGS; Reorthogonalization; Iterative orthogonalization; Column pivoting; Accurate computation of small residuals

1. Introduction

In this paper, we explain how the row-oriented modified Gram–Schmidt (MGS) algorithm is adapted to include reorthogonalization. Let A be a real $m \times n$ matrix. It

E-mail address: dax@vms.huji.ac.il (A. Dax).

is assumed throughout this paper that $m > n$ and that the columns of A are linearly independent. The Gram–Schmidt orthogonalization process is aimed at producing an $m \times n$ orthogonal matrix,

$$Q = [\mathbf{q}_1, \ldots, \mathbf{q}_n], \qquad Q^{\mathrm{T}}Q = I,$$

and an $n \times n$ upper triangular matrix, $R = (r_{ij})$, such that

$$A = QR. \tag{1.1}$$

The columns of Q are obtained by successively orthogonalizing the columns of A. The classical Gram–Schmidt (CGS) algorithm and the modified Gram–Schmidt (MGS) algorithm share the property that the computed matrices Q and R satisfy a bound of the form

$$\|A - QR\|_2 \leqslant \gamma\varepsilon\|A\|_2, \tag{1.2}$$

where ε denotes the machine precision (or unit round-off) in our computations and γ is a constant that depends on $m, n,$ and the details of the arithmetic. The difference between the two methods lies in their ability to orthogonalize the columns of A. Let

$$\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_n > 0$$

denote the singular values of A. Bjorck [2] has shown that if $\sigma_1/\sigma_n \ll 1/\varepsilon$, then the MGS algorithm yields a matrix Q that satisfies

$$\left\|I - Q^{\mathrm{T}}Q\right\|_2 \leqslant \gamma\varepsilon(\sigma_1/\sigma_n). \tag{1.3}$$

In other words, Q is guaranteed to be nearly orthogonal only when A is a well-conditioned matrix. The CGS fails to satisfy a bound of the form (1.3) and the computed columns of Q may depart from orthogonality to an almost arbitrary extent. The difference between the CGS and the MGS was observed by Rice [15]. For a detailed discussion of the bounds (1.2) and (1.3), see [2–5,11,17].

The MGS algorithm is implemented in two ways: A “row-oriented” version and a “column-oriented” version. The origin of these names lies in the way R is constructed. In the row-oriented version R is built row after row. In the column-oriented version R is built column after column, as in the CGS algorithm. (The details are specified in the following sections.) Nevertheless, as the columns of Q are generated in the same way, the two variants are numerically equivalent. They perform the same operations and produce identical numerical results. The difference between the two versions lies in the fact that only the row-oriented version is capable of performing column pivoting. The use of column pivoting brings a number of important advantages. First, it delays the deteriorating effects of rounding errors to the last columns of Q and the last rows of R. Second, as explained in Section 2, R satisfies (2.8) and (2.10). This feature improves the accuracy of the back substitution process for solving a linear system of the form (7.6), e.g. [11, pp. 155, 156, 161]. The solution of (7.6) is part of the standard method for solving linear least-squares problems. Further benefits come from the information which is exposed by the *QR* decomposition. For example, the pivoted *QR* decomposition has a good reputation as a gap-revealing algorithm, e.g. [17, pp. 373–375]. Similarly, QR with column pivoting may reveal that A is numerically rank deficient. In this case, it provides a simple method for calculating pseudo-inverse solutions of the relevant least-squares problem. See, for example, [4, pp. 103–117], [8, pp. 130–132] or [10, pp. 162–166].

The loss of orthogonality in the columns of Q can be avoided by repeated use of the orthogonalization process. This idea has been discussed by a number of authors under the names "reorthogonalization" and "iterative orthogonalization". See Refs. [1,3,4,6,12,14–17]. The iterative CGS algorithm is due to Daniel et al. [6], while Ruhe [16] was the first to propose a column-oriented iterative MGS algorithm. A further discussion of these techniques is given by Hoffmann [12] who compares the performance of the two methods. So far it was generally accepted that a row-oriented version of the iterative MGS is not possible [12, p. 338]. Nevertheless, as the next section shows, there is an elegant way to implement such an algorithm. This breakthrough paves the way for iterated MGS with column pivoting.

The outline of our paper is as follows. In Section 2, we present the new algorithm, while Section 3 compares it with the column-oriented iterative MGS. The need for reorthogonalization is explained in Section 4. Yet, as Section 5 shows, in some situations, two orthogonalizations are not enough to ensure small deviation from orthogonality. A further insight into the nature of the iterative orthogonalization process is gained in Section 6, in which we utilize its links with the Gauss–Seidel iteration. The use of Gram–Schmidt QR factorization for solving linear least-squares problems is discussed in Section 7. The standard solution method is based on orthogonalization of the r.h.s. vector $\mathbf{b}$ against $\mathbf{q}_1, \ldots, \mathbf{q}_n$, which results in the residual vector $\mathbf{r}^*$. This process loses accuracy when $\|\mathbf{r}^*\|_2$ is small with respect to $\|\mathbf{b}\|_2$, so iterative orthogonalization is essential to ensure accurate computation of small residuals.

We shall complete this section by introducing some necessary notations. The Gram–Schmidt orthogonalization process is an iterative method that consists of n iterations. Starting with $Q_0 = A$ and $R_0 = 0 \in \mathbb{R}^{n\times n}$ it generates two sequences of matrices, Q_k and $R_k, k = 1, \ldots, n$, such that $Q = Q_n$ and $R = R_n$. The kth iteration starts with Q_{k-1} and ends with Q_k. In practice Q_k is overwritten on Q_{k-1}. The same is true for R_k and R_{k-1}. At the beginning of the kth iteration we have already computed $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$, while the other columns of Q_{k-1} are denoted as $\mathbf{a}_j^{(k-1)}$, $j = k, \ldots, n$. That is,

$$Q_{k-1} = \left[\mathbf{q}_1, \ldots, \mathbf{q}_{k-1},\ \mathbf{a}_k^{(k-1)}, \mathbf{a}_{k+1}^{(k-1)}, \ldots, \mathbf{a}_n^{(k-1)}\right],$$
$$Q_k = \left[\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}, \mathbf{q}_k,\ \mathbf{a}_{k+1}^{(k)}, \ldots, \mathbf{a}_n^{(k)}\right],$$

and so forth. The kth iteration is composed of a number of steps and $\mathbf{a}_j^{(k-1)}$, $j = k, \ldots, n$, denotes the current value of the jth column of Q_{k-1} during these steps. In the next iteration, $\mathbf{a}_j^{(k-1)}$ automatically turns into $\mathbf{a}_j^{(k)}$. The entries of R_{k-1} are used without noting the iteration index. This way r_{ij} denotes the current value of the (i, j) element of R_{k-1} during the kth iteration. The notation $:=$ is used to denote arithmetic assignment. Thus, e.g., $r_{ik} := r_{ik} + \alpha_i$ means “set the new value of r_{ik} to be $r_{ik} + \alpha_i$”.

2. The new algorithm

In this section, we describe a row-oriented MGS algorithm with reorthogonalization. We start with a simple version that illustrates how row-oriented and column-oriented orthogonalizations are combined together. Later, we shall show how the algorithm is modified to include column pivoting and iterative orthogonalization. The kth iteration of the proposed method, $k = 1, 2, \ldots, n$, is composed of the following three steps.

```
Step 1: (Reorthogonalization of $\mathbf{a}_k^{(k-1)}$ with respect to $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$)
    If $k = 1$ skip to Step 2. Otherwise, for $i = 1, \ldots, k-1$, do as follows:
    Set $\alpha_i = \mathbf{q}_i^{\mathrm{T}} \mathbf{a}_k^{(k-1)}$, $r_{ik} := r_{ik} + \alpha_i$ and $\mathbf{a}_k^{(k-1)} := \mathbf{a}_k^{(k-1)} - \alpha_i \mathbf{q}_i$.
Step 2: (Normalization of $\mathbf{a}_k^{(k-1)}$)
    Set $r_{kk} = \|\mathbf{a}_k^{(k-1)}\|_2$ and $\mathbf{q}_k = \mathbf{a}_k^{(k-1)} / r_{kk}$.
Step 3: (Orthogonalization of $\mathbf{a}_{k+1}^{(k-1)}, \ldots, \mathbf{a}_n^{(k-1)}$ with respect to $\mathbf{q}_k$)
    If $k = n$ terminate. Otherwise, for $j = k+1, \ldots, n$ do as follows:
    Set $r_{kj} = \mathbf{q}_k^{\mathrm{T}} \mathbf{a}_j^{(k-1)}$ and $\mathbf{a}_j^{(k-1)} := \mathbf{a}_j^{(k-1)} - r_{kj} \mathbf{q}_k$.
```

Observe that without Step 1, the new algorithm is exactly the row-oriented MGS algorithm (see [3] or [4]). The justification of the way we build R is based on the following arguments. Using matrix notations the kth iteration is summarized by the equality

$$Q_{k-1} E_1^{(k)} \cdots E_{k-1}^{(k)} E_k^{(k)} E_{k+1}^{(k)} \cdots E_n^{(k)} = Q_k,$$

where $E_j^{(k)}$, $j = 1, \ldots, n$, are elementary matrices: For $j = 1, \ldots, k-1$, post-multiplication by $E_j^{(k)}$ subtracts the jth column, multiplied by α_j, from the kth column. Similarly, post-multiplication by $E_k^{(k)}$ divides the kth column by r_{kk}; while for $j = k+1, \ldots, n$, post-multiplication by $E_j^{(k)}$ subtracts the kth column, multiplied by r_{kj}, from the jth column. Define

$$E^{(k)} = E_1^{(k)} \cdots E_{k-1}^{(k)} \cdot E_k^{(k)} \cdot E_{k+1}^{(k)} \cdots E_n^{(k)}.$$

Then

$$A E^{(1)} \cdots E^{(n)} = Q_n = Q$$

and

$$R = (E^{(n)})^{-1} \cdots (E^{(1)})^{-1},$$

where

$$(E^{(k)})^{-1} = (E_n^{(k)})^{-1} \cdots (E_1^{(k)})^{-1} \quad \text{for } k = 1, \ldots, n.$$

The matrices $(E_j^{(k)})^{-1}$ have a simple structure: For $j = 1, \ldots, k-1$, post-multiplication by $(E_j^{(k)})^{-1}$ adds the jth column, multiplied by α_j, to the kth column. Similarly, post-multiplication by $(E_k^{(k)})^{-1}$ multiplies the kth column by r_{kk}; while for $j = k+1, \ldots, n$, post-multiplication by $(E_j^{(k)})^{-1}$ adds the kth column, multiplied by r_{kj}, to the jth column. These relations explain the rules for constructing R.

The incorporation of column pivoting in our algorithm is done in the same way as in the row-oriented MGS algorithm. A common pivoting strategy is to start the kth iteration by interchanging columns such that

$$\left\| \mathbf{a}_k^{(k-1)} \right\|_2^2 = \max_{j=k,\ldots,n} \left\| \mathbf{a}_j^{(k-1)} \right\|_2^2. \tag{2.1}$$

See e.g. [4,13,15]. This rule is similar to the standard pivoting strategy of the Householder *QR* factorization (e.g. [4,9,10,17]). The practical implementation of column pivoting is achieved by adding the following step at the start of the kth iteration.

Step 0: (Column pivoting) Compute a column index j^* such that

$$\left\| \mathbf{a}_{j^*}^{(k-1)} \right\|_2^2 = \max_{j=k,\ldots,n} \left\| \mathbf{a}_j^{(k-1)} \right\|_2^2.$$

If $j^* = k$ skip to Step 1. Otherwise, interchange $\mathbf{a}_{j^*}^{(k-1)}$ with $\mathbf{a}_k^{(k-1)}$ and r_{ij^*} with r_{ik}, $i = 1, \ldots, k-1$.

The updating of the terms $\|\mathbf{a}_j^{k-1}\|_2^2$, $j = k, \ldots, n$, from one iteration to the next can be based on Pythagoras' theorem: Let $\mathbf{q}$ be a unit vector, $\mathbf{a}$ an arbitrary vector and $\alpha = \mathbf{q}^{\mathrm{T}}\mathbf{a}$. Then

$$\|\mathbf{a} - \alpha\mathbf{q}\|_2^2 = \|\mathbf{a}\|_2^2 - \alpha^2.$$

Hence, e.g., when $\mathbf{a}_j^{(k-1)}$ is modified in Step 3 the value of $\|\mathbf{a}_j^{(k-1)}\|_2^2$ can be updated by the rule

$$\left\| \mathbf{a}_j^{(k-1)} \right\|_2^2 := \left\| \mathbf{a}_j^{(k-1)} \right\|_2^2 - (r_{kj})^2.$$

Remark. The use of Pythagoras' updating rule has an inherent computational difficulty. Let ε denote the unit round-off in our computations. Then the size of the rounding error in the computed value of $\|\mathbf{a}_j^{(0)}\|_2^2 = \sum_{i=1}^m (a_{ij})^2$ is about $\varepsilon_j = \varepsilon\|\mathbf{a}_j^{(0)}\|_2^2$, and the same is true for the coming (updated) values of $\|\mathbf{a}_j^{(k-1)}\|_2^2$. Hence, the relative error in the updated value of $\|\mathbf{a}_j^{(k-1)}\|_2^2$ is about $\varepsilon_j/\|\mathbf{a}_j^{(k-1)}\|_2^2$. Therefore, when this ratio exceeds a certain threshold value, $\tau = \min\{\varepsilon^{1/4}, 0.01\}$ say, we cannot trust the current (updated) value of $\|\mathbf{a}_j^{(k-1)}\|_2^2$. Fortunately, there is a simple way to overcome this drawback: Once the current (updated) value of $\|\mathbf{a}_j^{(k-1)}\|_2^2$ becomes smaller than ε_j/τ, both $\|\mathbf{a}_j^{(k-1)}\|_2^2$ and ε_j are recomputed according to the rules

$$\left\|\mathbf{a}_j^{(k-1)}\right\|_2^2 = \sum_{i=1}^{m}\left(a_{ij}^{(k-1)}\right)^2$$

and

$$\varepsilon_j = \varepsilon\left\|\mathbf{a}_j^{(k-1)}\right\|_2^2,$$

where $\mathbf{a}_{ij}^{(k-1)}$ denotes the ith component of $\mathbf{a}_j^{(k-1)}$. This precaution ensures at least two correct digits in the computed value of $\|\mathbf{a}_j^{(k-1)}\|_2^2$, which is satisfactory for the purpose of column pivoting. In the following sections we shall see that the level of rounding errors in $\mathbf{a}_j^{(k-1)}$ is expected to stay about $\varepsilon\|\mathbf{a}_j^{(0)}\|_2$. Hence, the value of $\|\mathbf{a}_j^{(k-1)}\|_2^2$ is unlikely to be considerably smaller than $\varepsilon^2\|\mathbf{a}_j^{(0)}\|_2^2$. Therefore, if $\varepsilon \leqslant 10^{-8}$, then recomputation of $\|\mathbf{a}_j^{(k-1)}\|_2^2$ and ε_j is unlikely to occur more than three times.

The use of iterative orthogonalization was initiated by the works of Daniel et al. [6], Ruhe [16] and Hoffmann [12]. See also [1,3,4,14]. The iterative version of Step 1 is carried out as follows.

Step 1*: (Iterative orthogonalization of $\mathbf{a}_k^{(k-1)}$ with respect to $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$)
If $k = 1$ skip to Step 2. Otherwise, we generate a sequence of vectors $\mathbf{u}_\ell$, $\ell = 0, 1, 2, \ldots,$ in the following way. Starting from $\mathbf{u}_0 = \mathbf{a}_k$ and $\mathbf{u}_1 = \mathbf{a}_k^{(k-1)}$, the vector $\mathbf{u}_{\ell+1}$ is obtained by orthogonalizing $\mathbf{u}_\ell$ with respect to $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$. In practice the vectors $\mathbf{u}_\ell$, $\ell = 0, 1, \ldots,$ are overwritten on $\mathbf{a}_k^{(k-1)}$, so, only one vector $\mathbf{u} \equiv \mathbf{a}_k^{(k-1)}$ is used to denote the current value of $\mathbf{u}_\ell$. To stop the iterative orthogonalization process we use a preassigned constant $\rho > 1$. (Typical values of of ρ are $\rho = 2$ or $\rho = \sqrt{2}$.) With these notations at hand the details of the ℓth iteration, $\ell = 1, 2, \ldots,$ in which $\mathbf{u}_{\ell+1}$ is obtained from $\mathbf{u}_\ell$, are as follows: If

$$\rho^2\|\mathbf{u}_\ell\|_2^2 \geqslant \|\mathbf{u}_{\ell-1}\|_2^2, \tag{2.2}$$

then terminate. Otherwise, for $i = 1, \ldots, k-1$ do as follows:
Set $\alpha_i = q_i^{\mathrm{T}}\mathbf{u}_\ell$, $r_{ik} := r_{ik} + \alpha_i$ and $\mathbf{u}_\ell := \mathbf{u}_\ell - \alpha_i\mathbf{q}_i$.

One motivation behind the stopping condition (2.2) lies in the following observations. Let B denote the $m \times (k-1)$ matrix, whose columns are $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$. In practice

$$B^{\mathrm{T}}B = I + E,$$

where $E = (e_{ij})$ is an error matrix which is generated by the rounding errors in the computation of $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$. Thus, once the unit vector $\hat{\mathbf{u}}_{\ell+1} = \mathbf{u}_{\ell+1}/\|\mathbf{u}_{\ell+1}\|_2$ satisfies a bound of the form

$$\left\| B^{\mathrm{T}}\hat{\mathbf{u}}_{\ell+1} \right\|_2 \leqslant (\rho^2 - 1)^{1/2}\|E\|_2,$$

there is no reason to continue the iterative orthogonalization process. Observe that for each index i, $1 \leqslant i \leqslant k-1$,

$$\mathbf{q}_i^{\mathrm{T}}\mathbf{u}_{\ell+1} = \alpha_i - \left(\alpha_i \mathbf{q}_i^{\mathrm{T}}\mathbf{q}_i + \cdots + \alpha_{k-1}\mathbf{q}_i^{\mathrm{T}}\mathbf{q}_{k-1}\right) = -(\alpha_i e_{ii} + \cdots + \alpha_{k-1} e_{i,k-1}).$$

Using matrix notations the last equalities are written as

$$B^{\mathrm{T}}\mathbf{u}_{\ell+1} = T\boldsymbol{\alpha}, \tag{2.3}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{k-1})^{\mathrm{T}} \in \mathbb{R}^{k-1}$ and $T = (t_{ij})$ is a $(k-1) \times (k-1)$ upper triangular matrix which consists of the upper triangular part of E. That is, $t_{ij} = 0$ when $i > j$ and $t_{ij} = e_{ij}$ when $i \leqslant j$. Note also that the factors α_i, $i = 1, \ldots, k-1$, satisfy the equality

$$\|\mathbf{u}_{\ell+1}\|_2^2 = \|\mathbf{u}_\ell\|_2^2 - \left(\alpha_1^2 + \cdots + \alpha_{k-1}^2\right) = \|\mathbf{u}_\ell\|_2^2 - \|\boldsymbol{\alpha}\|_2^2.$$

Assume now that (2.2) holds with respect to the $\mathbf{u}_\ell$ and $\mathbf{u}_{\ell+1}$. In this case

$$\|\boldsymbol{\alpha}\|_2^2 = \|\mathbf{u}_\ell\|_2^2 - \|\mathbf{u}_{\ell+1}\|_2^2 \leqslant \rho^2\|\mathbf{u}_{\ell+1}\|_2^2 - \|\mathbf{u}_{\ell+1}\|_2^2 \leqslant (\rho^2 - 1)\|\mathbf{u}_{\ell+1}\|_2^2,$$

so the vector $\hat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}/\|\mathbf{u}_{\ell+1}\|_2$ satisfies

$$\left\|\hat{\boldsymbol{\alpha}}\right\|_2 \leqslant (\rho^2 - 1)^{1/2}. \tag{2.4}$$

Hence, from (2.3) we conclude that

$$\left\| B^{\mathrm{T}}\hat{\mathbf{u}}_{\ell+1} \right\|_2 = \left\| T\hat{\boldsymbol{\alpha}} \right\|_2 \leqslant \|T\|_2 \; \left\|\hat{\boldsymbol{\alpha}}\right\|_2 \leqslant (\rho^2 - 1)^{1/2}\|T\|_2. \tag{2.5}$$

In other words, the error propagation in the orthogonalization process is controlled by the size of $\hat{\boldsymbol{\alpha}}$, while (2.2) ensures that $\|\hat{\boldsymbol{\alpha}}\|_2$ is smaller than $(\rho^2 - 1)^{1/2}$.

The implementation of the iterative orthogonalization idea to row-oriented MGS can be done in a number of ways. One option is to add the following step at the end of the basic iteration.

Step 4: (Iterative orthogonalization of $\mathbf{a}_{k+1}^{(k-1)}, \ldots, \mathbf{a}_n^{(k-1)}$ against $\mathbf{q}_k$)

Here for each column index j, $j = k+1, \ldots, n$, we generate a sequence of vectors $\mathbf{v}_\ell$, $\ell = 0, 1, 2, \ldots,$ in the following way. Starting with $\mathbf{v}_0 = \mathbf{a}_j$ and $\mathbf{v}_1 = \mathbf{a}_j^{(k-1)}$, the vector $\mathbf{v}_{\ell+1}$ is obtained by orthogonalizing $\mathbf{v}_\ell$ against $\mathbf{q}_k$. In practice the vectors $\mathbf{v}_\ell$, $\ell = 2, 3, \ldots,$ are overwritten on $\mathbf{a}_j^{(k-1)}$. The details of the ℓth iteration, in which $\mathbf{v}_{\ell+1}$ is obtained from $\mathbf{v}_\ell$, are as follows: If

$$\rho^2\|\mathbf{v}_\ell\|_2^2 \geqslant \|\mathbf{v}_{\ell-1}\|_2^2, \tag{2.6}$$

terminate. Otherwise, set

$$\alpha = \mathbf{q}_k^{\mathrm{T}}\mathbf{v}_\ell,\ r_{kj} := r_{kj} + \alpha,\ \mathbf{v}_{\ell+1} = \mathbf{v}_\ell - \alpha\mathbf{q}_k \text{ and } \|\mathbf{v}_{\ell+1}\|_2^2 = \|\mathbf{v}_\ell\|_2^2 - \alpha^2.$$

In order to allow more flexibility, the value of ρ in Step 4 may differ from its value in Step 1^*. Note also that Steps 3 and 4 can be merged into one step.

We shall finish this section with brief remarks on the effects of column pivoting. Recall that the basic iteration of row-oriented MGS with column pivoting is composed of Steps 0, 2 and 3. In this algorithm the column interchanges at the beginning of the kth iteration ensure (2.1), while at the end of Step 3

$$|r_{kj}| = \left|\mathbf{q}_k^{\mathrm{T}}\mathbf{a}_j^{(k-1)}\right| \leqslant \|\mathbf{q}_k\|_2 \left\|\mathbf{a}_j^{(k-1)}\right\|_2 = \left\|\mathbf{a}_j^{(k-1)}\right\|_2.$$

Therefore, since $r_{kk} = \|\mathbf{a}_k^{(k-1)}\|_2$,

$$|r_{kk}| \geqslant |r_{kj}| \qquad \text{for } j = k+1, \ldots, n. \tag{2.7}$$

Moreover, the equality

$$\left\|\mathbf{a}_j^{(k)}\right\|_2^2 = \left\|\mathbf{a}_j^{(k-1)}\right\|_2^2 - r_{kj}^2$$

shows that for each column index $j,\ \ j = k+1, \ldots, n$,

$$\left\|\mathbf{a}_j^{(k-1)}\right\|_2^2 = \sum_{i=k}^{j} r_{ij}^2$$

and

$$r_{kk}^2 \geqslant \sum_{i=k}^{j} r_{ij}^2. \tag{2.8}$$

Hence, in particular,

$$|r_{11}| \geqslant |r_{22}| \geqslant \cdots \geqslant |r_{nn}| \tag{2.9}$$

and

$$|r_{kk}| \geqslant |r_{ij}| \quad \forall k \leqslant i \leqslant j \leqslant n. \tag{2.10}$$

Let us turn now to see what happens to these relations when the basic iteration is extended to include iterative orthogonalization. Assume first that the basic iteration is composed of Steps 0, 2, 3 and 4. Observe that the changes in r_{ij} during Step 4 are essentially negligible (see Sections 4–6). Hence, (2.7) and (2.9) are expected to remain valid in this case.

The situation is more complicated when the basic iteration is composed of Steps 0, 1^* (or 1), 2 and 3. Here the orthogonalization process of Step 1^* (or 1) is capable of causing a substantial reduction in the size of $\|\mathbf{a}_k^{(k-1)}\|_2^2$. Hence, the addition of this step may violate (2.1) and the succeeding relations (2.7)–(2.10). Nevertheless, since $\mathbf{a}_k^{(k-1)}$ has already passed one orthogonalization before Step 1^* takes place, a significant reduction in the size of $\|\mathbf{a}_k^{(k-1)}\|_2^2$ is expected only when $\mathbf{a}_k^{(k-1)}$ becomes so small that it is considerably contaminated by rounding errors (see Sections 5 and 6). Thus, even in this case, the pivoting operations remain valuable: As before the deteriorating effects of rounding errors are deferred to the last iterations and the resulting *QR* factorization provides the information which is needed for handling rank-deficient problems.

3. Relations with other orthogonalization techniques

We shall start with a brief description of the iterative column-oriented MGS algorithm. This method was firstly proposed by Ruhe [16], while Hoffmann [12] investigates its performance. The kth iteration of Ruhe's algorithm, $k = 1, \ldots, n$, is composed of the following three steps.

Step $1'$: (Orthogonalization of $\mathbf{a}_k^{(k-1)}$ with respect to $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$)
If $k = 1$ skip to Step $3'$. Otherwise, for $i = 1, \ldots, k-1$, do as follows:
Set $r_{ik} = \mathbf{q}_i^{\mathrm{T}}\mathbf{a}_k^{(k-1)}$ and $\mathbf{a}_k^{(k-1)} := \mathbf{a}_k^{(k-1)} - r_{ik}\mathbf{q}_i$.
Step $2'$: Identical to Step 1^* above.
Step $3'$: (Normalization of $\mathbf{a}_k^{(k-1)}$)
Set $r_{kk} = \|\mathbf{a}_k^{(k-1)}\|_2$ and $\mathbf{q}_k = \mathbf{a}_k^{(k-1)}/r_{kk}$.

Observe that without Step $2'$ Ruhe's algorithm reduces to the column-oriented MGS. Note also that Ruhe's algorithm is numerically equivalent to restricted version of our algorithm in which the basic iteration is composed of Steps 1^*, 2 and 3. These two algorithms perform the same operations but in different order. This does not affect the construction of the vectors $\mathbf{a}_j^{(k-1)}$ and the way rounding errors are generated. So both methods produce the same numerical results. However, being a row-oriented scheme, our algorithm is capable of performing column pivoting. The following example illustrates the importance of this feature.

Let $A = [\mathbf{a}_1, \ldots, \mathbf{a}_n]$ be a well-conditioned matrix and let $\mathbf{a}_3$ be redefined by the rule $\mathbf{a}_3 := \mathbf{a}_1 + \mathbf{a}_2 + \tau\mathbf{a}_3$, where τ is smaller than the round-off unit in our computations. Then, in the third iteration of Ruhe's algorithm, $\mathbf{a}_3^{(2)}$ becomes so small that it is entirely contaminated by rounding errors. Thus, though $\mathbf{q}_3$ is orthogonal to $\mathbf{q}_1$ and $\mathbf{q}_2$, its components are essentially random numbers that come from rounding errors. This in turn implies that the components of the vectors $\mathbf{q}_4, \mathbf{q}_5, \ldots, \mathbf{q}_n$ also contain large "random" portion that comes from rounding errors and the same is true for the coefficients r_{ij}, $j = 3, \ldots, n$, $i = 1, \ldots, j$. Hence, the information that comes from this factorization is practically useless. Let us turn now to see how the new algorithm handles the above matrix. Assume that the basic iteration of our algorithm is composed of Steps 0, 1^*, 2 and 3. Recall that without Step 0, this algorithm is numerically equivalent to Ruhe's algorithm. Yet the column interchanges in Step 0 gradually moves $\mathbf{a}_3$ from the third column to the last column. This way the undesired effects of rounding errors are deferred until the last iteration and the only column of Q which is considerably contaminated by rounding errors is $\mathbf{q}_n$. Similarly, the only entry of R which might be effected by rounding errors is r_{nn}. All the other columns of Q (entries of R) are essentially "free" of rounding errors. Moreover, since r_{nn} is practically zero, the resulting QR factorization provides useful information on A and it is applicable for calculating pseudo-inverse solutions (see [4, pp. 103–107], [8, pp. 130–132] or [10, pp. 162– 166]).

4. The need for reorthogonalization

In this section, we briefly explain the need for reorthogonalization in the Gram–Schmidt process. For this purpose we consider the orthogonalization of $\mathbf{a}_k$ with respect to the vectors $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$. To simplify the coming discussion, we make the assumption that these vectors are mutually orthogonal. So the $m \times (k-1)$ matrix

$$B = [\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}]$$

is assumed to satisfy

$$B^{\mathrm{T}} B = I. \tag{4.1}$$

Let the sequence $\mathbf{u}_\ell,\ \ell = 0, 1, 2, \ldots,$ be generated as in Step 1* of our algorithm (which is also Step 2′ of Ruhe's algorithm). Recall that $\mathbf{u}_0 = \mathbf{a}_k$ and that $\mathbf{u}_{\ell+1}$ is obtained by orthogonalizing $\mathbf{u}_\ell$ with respect to $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$. The orthogonalization process is done in the MGS style:

For $i = 1, \ldots, k-1$ set $\alpha_i = \mathbf{q}_i^{\mathrm{T}} \mathbf{u}_\ell$ and $\mathbf{u}_\ell := \mathbf{u}_\ell - \alpha_i \mathbf{q}_i$. Finally, set $\mathbf{u}_{\ell+1} = \mathbf{u}_\ell$.

Define $\mathbb{S} = \mathrm{Span}\{\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}\}$ and let $\mathbb{T} = \mathrm{Null}(B^{\mathrm{T}})$ denote the orthogonal complement of $\mathbb{S}$ in $\mathbb{R}^m$. Then each vector $\mathbf{u}_\ell$ has a unique decomposition of the form

$$\mathbf{u}_\ell = \mathbf{s}_\ell + \mathbf{t}_\ell, \tag{4.2}$$

where $\mathbf{s}_\ell \in \mathbb{S}$ and $\mathbf{t}_\ell \in \mathbb{T}$. Of course, in exact arithmetic $\mathbf{s}_\ell = \mathbf{0}$ and $\mathbf{t}_\ell = \mathbf{t}_0$ for $\ell = 1, 2, \ldots$ However, in practice $\mathbf{s}_\ell$ differs from $\mathbf{0}$ because of rounding errors in the orthogonalization process. In floating point arithmetic

$$\|\mathbf{s}_\ell\|_2 = \beta_\ell \gamma \varepsilon \|\mathbf{u}_{\ell-1}\|_2, \tag{4.3}$$

where ε denotes the machine precision (unit round-off) in our computations, γ is some constant that depends on $k,\ m,$ and the details of the arithmetic, and β_ℓ is some number from the interval $[-1, 1]$. See [2,4,11]. Consequently, the unit vector $\hat{\mathbf{u}}_\ell = \mathbf{u}_\ell / \|\mathbf{u}_\ell\|_2$ satisfies

$$\left\| B^{\mathrm{T}} \hat{\mathbf{u}}_\ell \right\|_2 = \left\| B^{\mathrm{T}} \mathbf{s}_\ell \right\|_2 \Big/ \|\mathbf{u}_\ell\|_2 = \|\mathbf{s}_\ell\|_2 / \|\mathbf{u}_\ell\| = \beta_\ell \gamma \varepsilon \|\mathbf{u}_{\ell-1}\|_2 / \|\mathbf{u}_\ell\|_2. \tag{4.4}$$

In other words, the deviation from orthogonality is proportional to the ratio $\|\mathbf{u}_{\ell-1}\|_2/\|\mathbf{u}_\ell\|_2$. The larger is this ratio, the larger the deviation!

These observations clearly explain why reorthogonalization is essential to ensure orthogonality in the Gram–Schmidt process. In practice the ratio

$$\|\mathbf{u}_0\|_2/\|\mathbf{u}_1\|_2 = \|\mathbf{a}_k\|_2/\|\mathbf{u}_1\|_2 \tag{4.5}$$

can be arbitrarily large. Hence, stopping the iterative orthogonalization process after one iteration, without paying attention to this ratio, is doomed to produce large deviation from orthogonality. The more ill-conditioned is A, the larger is the possibility of having a situation in which the ratio (4.5) is large.

At the same time we see that the termination condition (2.2) ensures that the size of $\|B^{\mathrm{T}}\hat{\mathbf{u}}_\ell\|_2$ stays below $\gamma\varepsilon\rho$, which is as good as we can ask for. This raises the question of what leads us to believe that (2.2) will eventually hold and how many iterations are needed to achieve this goal. The answer is given in the next section.

5. How many orthogonalizations are necessary?

We continue the former discussion with an attempt to clarify the situation after two successive orthogonalizations. Let the positive number ν be defined by the equality

$$\|\mathbf{s}_1\|_2 = \nu\|\mathbf{t}_1\|_2. \tag{5.1}$$

Then

$$\|\mathbf{u}_1\|_2^2 = \|\mathbf{s}_1 + \mathbf{t}_1\|_2^2 = \|\mathbf{s}_1\|_2^2 + \|\mathbf{t}_1\|_2^2 = (1+\nu^2)\|t_1\|_2^2. \tag{5.2}$$

In practice ν is not expected to be much larger than one. Otherwise, $\mathbf{u}_1$ is practically zero, as $\mathbf{s}_1$ is generated by rounding errors. Furthermore, even in this case, when $\mathbf{u}_1$ is considerably contaminated by rounding errors, it is still reasonable to assume that ν has a moderate size, because of the random nature of the rounding errors. To see the last point assume for a moment that the "true" value of $\mathbf{u}_1$ should be $\mathbf{0}$, so $\mathbf{u}_1$ is entirely made of rounding errors. Then a priori we have no reason to expect that $\|\mathbf{t}_1\|$ is considerably smaller than $\|\mathbf{s}_1\|_2$, especially, when k is small with respect to m. However, as this is only a "probabilistic" argument, we might face some exceptions.

On the other hand, the vector $\mathbf{u}_2 = \mathbf{s}_2 + \mathbf{t}_2$ is likely to have a different structure: The size of $\mathbf{s}_2$ is expected to be about

$$\gamma\varepsilon\|\mathbf{u}_1\|_2 = \gamma\varepsilon(1+\nu^2)^{1/2}\|\mathbf{t}_1\|_2,$$

while $\mathbf{t}_2$ is expected to stay close to $\mathbf{t}_1$. In practice it is highly unlikely that ν exceeds $1/(\gamma\varepsilon)$, so $\|\mathbf{s}_2\|_2$ is expected to be (considerably) smaller than $\|\mathbf{t}_2\|_2$. In this case, when $\|\mathbf{s}_2\|_2$ is considerably smaller than $\|\mathbf{t}_2\|_2$, there exists a small number, δ, such that $|\delta| \ll 1$ and

$$\|\mathbf{u}_2\|_2^2 = (1+\delta)\|\mathbf{t}_1\|_2^2. \tag{5.3}$$

Combining (5.2) and (5.3) gives the equality

$$\|\mathbf{u}_1\|_2^2/\|\mathbf{u}_2\|_2^2 = (1+\nu^2)/(1+\delta), \tag{5.4}$$

which means that if ρ^2 is smaller than $(1+\nu^2)/(1+\delta)$ we will need a third orthogonalization!

Nevertheless, there are three pieces of good news that emerge from the above discussion. First, we see that a fourth orthogonalization is almost never needed: In practice it is highly unlikely that ν exceeds $1/(\gamma\varepsilon)$, so $\|\mathbf{s}_2\|_2$ is expected to be considerably smaller than $\|\mathbf{t}_2\|_2$, which ensures that the third orthogonalization ends with satisfactory results. Second, the use of column pivoting leads to the inequalities

$$\left\|\mathbf{a}_j^{(k-1)}\right\|_2 \leqslant \|\mathbf{u}_1\|_2, \quad j = k, \ldots, n, \tag{5.5}$$

while a third orthogonalization takes place only when $\|\mathbf{u}_1\|_2$ is about $\gamma\varepsilon\|\mathbf{a}_k\|_2$. In other words, a third orthogonalization is needed only when we reach a point from which the orthogonalization process is actually dominated by rounding errors! Third, the use of column pivoting delays the need for three orthogonalizations (and the propagation of rounding errors) to the last iterations. Moreover, from (2.2) we see that unless a third orthogonalization is carried out the vector $\mathbf{u}_2$ satisfies

$$\|\mathbf{u}_2\|_2 \geqslant \|\mathbf{u}_1\|_2/\rho. \tag{5.6}$$

In this case the inequalities (2.7)–(2.10) are replaced by

$$|r_{kk}| = \|\mathbf{u}_2\|_2 \geqslant \|\mathbf{u}_1\|_2/\rho \geqslant \left\|\mathbf{a}_j^{(k-1)}\right\|_2/\rho, \quad j = k+1, \ldots, n, \tag{5.7}$$

and

$$|r_{kk}| \geqslant |r_{ij}|/\rho \quad \forall k \leqslant i \leqslant j \leqslant n. \tag{5.8}$$

A violation of (5.8) is possible only after a third orthogonalization is performed. Yet, as we have seen, at this stage rounding errors actually dominate the Gram–Schmidt process.

It is instructive to inspect our conclusions in view of the experiments made by Daniel et al. [6] and Hoffmann [12]. The latter work examines the iterative CGS and the iterative MGS on a large collection of "randsvd" matrices. In these experiments there was no substantial difference between the two methods. The deviation from orthogonality in the final matrix, Q, was proportional to ρ. Thus, to gain maximal precision it is necessary to use a small ρ, e.g., $\rho = 2$. Yet, in all the experiments made by Hoffmann [12], a third orthogonalization never occurred. The explanation of this phenomenon is, perhaps, that the largest condition number is 10^{11} while the machine precision is $\varepsilon = 0.5 \cdot 10^{-14}$. That is, the condition numbers of the tested matrices are considerably smaller than $1/\varepsilon$. If the condition number of A exceeds $1/\varepsilon$, then it is possible that at some stage $\mathbf{u}_1$ is entirely contaminated by rounding errors. This paves the way for ν to be larger than one, so the iterative Gram–Schmidt algorithm may need more than two orthogonalizations. Indeed the experiments of Daniel et al. [6] illustrate this point. These experiments apply the iterative CGS algorithm (with

$\rho = \sqrt{2}$ and $\rho = 10$) on $m \times n$ sections of Hilbert matrix. Here $\varepsilon = 0.5 \cdot 8^{-12}$ while the condition number of the tested matrices is considerably larger than $1/\varepsilon$. Thus, as expected, three orthogonalizations are often needed. Moreover, restricting the number of orthogonalizations to two results in a total loss of orthogonality! In other words, when treating ill-conditioned matrices iterative orthogonalization is essential to ensure orthogonality. Nevertheless, there is no report on a case in which more than three orthogonalizations are needed.

6. The iterative MGS and the Gauss–Seidel iteration

A further insight is gained by discarding the assumption that the vectors $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$ are mutually orthogonal. Now (4.1) is replaced by the equality

$$B^{\mathrm{T}} B = I + E, \tag{6.1}$$

where E is an error-matrix that comes from rounding errors in the computation of $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$. Orthogonalizing $\mathbf{a}_k$ with respect to these vectors means finding the projection of $\mathbf{a}_k$ on $\mathbb{T}$, the orthogonal complement of $\mathbb{S} = \mathrm{Span}\{\mathbf{q}_1 \ldots, \mathbf{q}_{k-1}\}$. This can be done by solving the least-squares problem

$$\min \|\mathbf{a}_k - B\mathbf{x}\|_2^2$$

or the corresponding system of normal equations

$$B^{\mathrm{T}} B\mathbf{x} = B^{\mathrm{T}}\mathbf{a}_k. \tag{6.2}$$

Let the sequence $\mathbf{x}_\ell,\ \ell = 0, 1, 2, \ldots,$ be obtained by applying the Gauss–Seidel iteration for solving (6.2), starting with $\mathbf{x}_0 = \mathbf{0} \in \mathbb{R}^{k-1}$. Ruhe [16] was the first to note that in this case

$$\mathbf{u}_\ell = \mathbf{a}_k - B\mathbf{x}_\ell, \quad \ell = 0, 1, 2, \ldots, \tag{6.3}$$

where, as before, the sequence $\{\mathbf{u}_\ell\}$ is obtained by the iterative MGS process described in Step 1*. (A similar relation exists between the Jacobi iteration and the iterative CGS.) Let the matrices D and L be defined by the equality

$$E = D - L - L^{\mathrm{T}},$$

and the restrictions that D is a diagonal matrix and L is a strictly lower triangular matrix. In order to investigate the behavior of the sequences $\{\mathbf{x}_\ell\}$ and $\{\mathbf{u}_\ell\}$, we assume that

$$\|D\|_2 \ll 1 \tag{6.4}$$

and

$$\|L\|_2 \ll 1. \tag{6.5}$$

These assumptions imply that $B^{\mathrm{T}} B$ is a positive-definite matrix. So the theory of the Gauss–Seidel method ensures that the sequence $\{\mathbf{x}_\ell\}$ converges to $\mathbf{x}^*$, the unique solution of (6.2). Furthermore, the rate of convergence is determined by the rule

$$\mathbf{x}_\ell - \mathbf{x}^* = H^\ell \left(\mathbf{x}_0 - \mathbf{x}^*\right), \quad \ell = 0, 1, 2, \ldots,$$

where

$$H = (I + D - L)^{-1} L^{\mathrm{T}}$$

is the corresponding “iteration matrix”. Note that

$$(I + D - L)^{-1} = I + (L - D) + (L - D)^2 + \cdots,$$

so $\|H\|_2$ is only slightly larger than $\|L\|_2$. Now the relations

$$\begin{aligned} B^{\mathrm{T}}\mathbf{u}_\ell &= B^{\mathrm{T}}(\mathbf{a}_k - B\mathbf{x}_\ell) \\ &= B^{\mathrm{T}}B\left(\mathbf{x}^* - \mathbf{x}_\ell\right) = B^{\mathrm{T}}BH^\ell\left(\mathbf{x}^* - \mathbf{x}_0\right) = B^{\mathrm{T}}BH^\ell\mathbf{x}^* \end{aligned} \tag{6.6}$$

imply the bound

$$\left\|B^{\mathrm{T}}\mathbf{u}_\ell\right\|_2 \leqslant \left\|B^{\mathrm{T}}B\right\|_2 \|H\|_2^\ell \|\mathbf{x}^*\|_2. \tag{6.7}$$

Moreover, since $\mathbf{x}^* = (B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}}\mathbf{a}_k$ and

$$\left\|\left(B^{\mathrm{T}}B\right)^{-1}\right\|_2 \leqslant 1/(1 - \|E\|_2),$$

it follows that

$$\left\|B^{\mathrm{T}}\mathbf{u}_\ell\right\|_2 \leqslant \alpha\|H\|_2^\ell \|\mathbf{a}_k\|_2, \tag{6.8}$$

where

$$\alpha = \left\|B^{\mathrm{T}}\right\|_2 \left\|B^{\mathrm{T}}B\right\|_2 /(1 - \|E\|_2)$$

is a constant that satisfies $|1 - \alpha| \ll 1$. Substituting (4.2) into (6.8) gives

$$\|\mathbf{s}_\ell\|_2 \leqslant \beta\|H\|_2^\ell \|\mathbf{a}_k\|_2, \tag{6.9}$$

where β is another constant that satisfies $|1 - \beta| \ll 1$. Thus, in exact arithmetic, the sequence $\{\|\mathbf{s}_\ell\|_2\}$ converges to zero and the rate of convergence depends on the size of $\|L\|_2$.

The sequence $\{\mathbf{t}_\ell\}$ behaves in a different way. In exact arithmetic $\mathbf{t}_\ell = \mathbf{t}_0$ for $\ell = 1, 2, \ldots,$ so the changes between $\mathbf{t}_\ell$ and $\mathbf{t}_{\ell-1}$ come from rounding errors. Therefore, if $\|\mathbf{t}_0\|$ is considerably larger than $\gamma\varepsilon\|\mathbf{a}_k\|_2$, then the vectors $\mathbf{t}_\ell,\ \ell = 1, 2, \ldots,$ remain almost unchanged during the iterative orthogonalization process. On the other hand, if $\|\mathbf{t}_0\|_2$ is smaller than $\gamma\varepsilon\|\mathbf{a}_k\|_2$ or about that size, then $\|\mathbf{t}_1\|$ is essentially made of rounding errors. This may result in a substantial difference between $\mathbf{t}_0$ and $\mathbf{t}_1$. Yet, since $\|\mathbf{u}_1\|_2$ is much smaller than $\|\mathbf{u}_0\|_2$, in the next iterations there are minor changes between $\mathbf{t}_\ell$ and $\mathbf{t}_{\ell+1}$.

Summarizing the above discussion we see that a small value of ρ leads to slow build up of deviation from orthogonality. This allows us to assume that $\|L\|_2$ is not much larger than $\gamma\varepsilon$, so the probability to need a fourth orthogonalization remains small. Moreover, a third orthogonalization is carried out only when

$$\|\mathbf{u}_2\|_2 < \|\mathbf{u}_1\|_2/\rho, \tag{6.10}$$

while the assumption that $\|L\|_2$ is close to $\gamma\varepsilon$ implies that (6.10) is possible only when $\|\mathbf{t}_0\|_2$ is smaller than $\gamma\varepsilon\|\mathbf{a}_k\|_2$ or about that size. This in turn means that $\|\mathbf{u}_1\|_2$ is about $\|L\|_2\|\mathbf{a}_k\|_2$. In other words, the size of $\mathbf{u}_1$ is close to that of the rounding errors.

7. Solving linear least-squares problems

In this section, we consider the use of the computed *QR* factorization (1.1) for solving the standard least-squares problem

$$\min\|A\mathbf{x}-\mathbf{b}\|_2^2. \tag{7.1}$$

Let $\mathbf{x}^* \in \mathbb{R}^n$ denote the unique solution of (7.1) and let

$$\mathbf{r}^* = \mathbf{b} - A\mathbf{x}^* \tag{7.2}$$

denote the corresponding residual vector. As before $\mathbb{T}$ denotes the orthogonal complement of $\mathbb{S} = \mathrm{Span}\{\mathbf{q}_1, \ldots, \mathbf{q}_n\}$ in $\mathbb{R}^m$. Note that $\mathbf{r}^*$ is the projection of $\mathbf{b}$ on $\mathbb{T}$. Hence, $\mathbf{r}^*$ can be obtained by applying one sweep of the MGS algorithm to orthogonalize $\mathbf{b}$ against $\mathbf{q}_1, \ldots, \mathbf{q}_n$. This results in an extended *QR* factorization

$$[A, \mathbf{b}] = [Q, \mathbf{u}]\begin{bmatrix} R & \mathbf{z} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix}, \tag{7.3}$$

where $\mathbf{z} \in \mathbb{R}^n$ is determined by the MGS process and $\mathbf{u} = \mathbf{r}^*$. From (7.3) we derive the relations

$$A\mathbf{x} - \mathbf{b} = [A, \mathbf{b}]\begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} = Q(R\mathbf{x} - \mathbf{z}) - \mathbf{u} \tag{7.4}$$

and

$$\|A\mathbf{x} - \mathbf{b}\|_2^2 = \|Q(R\mathbf{x} - \mathbf{z}) - \mathbf{u}\|_2^2 = \|Q(R\mathbf{x} - \mathbf{z})\|_2^2 + \|\mathbf{u}\|_2^2, \tag{7.5}$$

where the last equality relies on the fact that $\mathbf{u} \in \mathbb{T}$. Now we see that the least-squares solution, $\mathbf{x}^*$, can be computed by solving the upper triangular linear system

$$R\mathbf{x} = \mathbf{z} \tag{7.6}$$

via back substitution. This constitutes the standard method for solving linear least-squares problems via the Gram–Schmidt *QR* factorization, e.g. [2], [3], [4, p. 65], [11, pp. 396, 397] and [17, pp. 297, 298].

However, the standard approach is not always satisfactory. In some applications it is desired that the computed residual will satisfy

$$\left\|A^{\mathrm{T}}\mathbf{r}^*\right\|_2 \leqslant \gamma\varepsilon\|\mathbf{r}^*\|_2. \tag{7.7}$$

This is the case, e.g., in certain affine scaling methods, where $\mathbf{r}^*/\|\mathbf{r}^*\|_2$ is used as a search direction in $\mathrm{Null}(A^{\mathrm{T}})$ while it is known that $\|\mathbf{r}^*\|_2$ tends to be considerably smaller than $\|\mathbf{b}\|_2$. See [7] for a detailed discussion of this issue. Yet, as we have seen, in this case one sweep of the MGS is not sufficient to ensure orthogonality. To rectify this flaw we use iterative orthogonalization of $\mathbf{b}$ against $\mathbf{q}_1, \ldots, \mathbf{q}_n$. The modified algorithm uses the vectors $\mathbf{z} = (z_1, \ldots, z_n)^{\mathrm{T}} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^m$, where $\mathbf{u}$ starts as $\mathbf{b}$ and ends as $\mathbf{r}^*$.

A modified least-squares algorithm

Step $1''$: (Orthogonalization of $\mathbf{b}$ against $\mathbf{q}_1, \ldots, \mathbf{q}_n$)
Set $\mathbf{u} = \mathbf{b}$. Then for $j = 1, 2, \ldots, n$ do as follows:
Set $z_j = \mathbf{q}_j^{\mathrm{T}}\mathbf{u}$ and $\mathbf{u} := \mathbf{u} - z_j\mathbf{q}_j$.

Step $2''$: (Iterative orthogonalization of $\mathbf{u}$ against $\mathbf{q}_1, \ldots, \mathbf{q}_n$)
Starting with $\mathbf{u}_0 = \mathbf{b}$ and $\mathbf{u}_1 = \mathbf{u}$ we generate a sequence of vectors, $\mathbf{u}_\ell$, $\ell = 0, 1, \ldots,$ where $\mathbf{u}_{\ell+1}$ is obtained by orthogonalizing $\mathbf{u}_\ell$ against $\mathbf{q}_1, \ldots, \mathbf{q}_n$. In practice all the vectors $\mathbf{u}_\ell$, $\ell = 0, 1, 2, \ldots,$ are overwritten on $\mathbf{u}$. The details of the ℓth iteration, in which $\mathbf{u}_{\ell+1}$ is obtained from $\mathbf{u}_\ell$, are as follows:
If (2.2) holds, terminate. Otherwise, for $j = 1, \ldots, n$ do as follows:
Set $\alpha_j = \mathbf{q}_j^{\mathrm{T}}\mathbf{u}$, $z_j := z_j + \alpha_j$ and $\mathbf{u} := \mathbf{u} - \alpha_j\mathbf{q}_j$.

Step $3''$: (Computation of $\mathbf{x}^*$)
Use back substitution to solve the upper triangular system (7.6).

Observe that without Step $2''$ the above algorithm is identical to the standard solution method. The most important feature of the proposed algorithm is that reorthogonalization is not added automatically. Each iteration uses (2.2) to decide whether a further orthogonalization is needed. This saves unnecessary iterations. Another feature that characterizes our method is that each reorthogonalization includes updating of $\mathbf{z}$. If Q is nearly orthogonal, then the resulting changes in $\mathbf{z}$ are expected to be negligible. However, this is not necessarily true when Q has some deviation from orthogonality. That is, $Q^{\mathrm{T}}Q = I + E$, where E is small but not negligible. In this case the iterative MGS behaves like a Gauss–Seidel method, so it might need more than two orthogonalizations to achieve small deviation from orthogonality.

Finally, we would like to clarify that the modified algorithm is not necessarily better than Householder's method or other least-squares solvers. In fact, as explained in [7], the rival methods can also be modified to ensure (7.7). Our point is that this is the "right way" to apply the Gram–Schmidt *QR* factorization for solving least-squares problems. The reader is referred to [4,10,17] for a detailed comparison between the Gram–Schmidt and the Householder approaches.

8. Concluding remarks

The use of iterative orthogonalization is aimed at ensuring small deviation from orthogonality in the columns of Q. Former implementations of this idea concentrate on CGS and column-oriented MGS. Moreover, it was generally accepted that a row-oriented iterative MGS is not possible [12, p. 338]. Yet, as this paper shows, there is an elegant way to resolve this difficulty. The interest that we have in a row-oriented iterative MGS comes from the observation that this method maintains small deviation from orthogonality and, at the same time, it is capable of applying column pivoting.

The advantage of column pivoting is illustrated in Section 3. We have seen that Ruhe's algorithm is numerically equivalent to a restricted version of our algorithm in which the basic iteration is composed of Steps 1*, 2 and 3. The addition of column pivoting results in a significant improvement in the performance of the algorithm. First, it defers the deteriorating effects of rounding errors to the last iteration. Now only the last column of Q and r_{nn} are considerably effected by rounding errors. Second, the resulting QR factorization enables us to handle rank-deficient least-squares problems.

It is true that the orthogonalizations in Step 1* may violate (2.1) and, perhaps, some of the succeeding relations. However, this should not be considered as a real flaw. On the contrary, a significant reduction in the size of $\|\mathbf{a}_k^{(k-1)}\|_2$ is possible only when two orthogonalizations are not enough. In this case the size of $\|\mathbf{a}_k^{(k-1)}\|_2$ is about $\gamma\varepsilon\|\mathbf{a}_k\|_2$, so the vector $\mathbf{a}_k^{(k-1)}$ is considerably contaminated by rounding errors. In other words, strong violation of (2.7)–(2.10) is not expected until we reach an iteration that requires three (or more) orthogonalizations. Yet from this stage onwards the orthogonalization process is actually controlled by rounding errors. Knowing that stage is valuable information when handling rank-deficient problems. Note that the last conclusion does not require a priori knowledge of ε or γ. The important point is, again, that pivoting operations delay the need for three orthogonalizations (and the domination of rounding errors) to the last stages of the Gram–Schmidt process.

Note that there is considerable flexibility in applying an iterative MGS scheme. This is one of the reasons that the presentation in Section 2 starts with a simple iteration that consists of only three steps. Then it is shown that the basic iteration can be modified in several ways. A further flexibility comes from the choice of the termination factor, ρ, and the way one ensures that the iterative orthogonalization process will terminate in a finite number of iterations.

The use of the MGS algorithm for solving linear least-squares problems is considered by a number of authors. However, none of the former descriptions of this method explicitly uses iterative orthogonalization. Perhaps, because they concentrate on the computation of $\mathbf{x}^*$. Yet in some applications it is necessary to ensure that the normalized residual, $\mathbf{r}^*/\|\mathbf{r}^*\|_2$, belongs to $\mathrm{Null}(A^{\mathrm{T}})$ even when $\|\mathbf{r}^*\|_2$ is small compared to $\|\mathbf{b}\|_2$. The algorithm proposed in Section 7 is a natural extension of the standard solution method that allows it to use iterative orthogonalization. The test (2.2) prevents unnecessary iterations and, at the same time, ensures accurate computation of small residuals. A further advantage of the modified scheme is its ability to provide accurate results when Q has some deviation from orthogonality.

References

[1] N.N. Abdelmalek, Round-off error analysis from Gram–Schmidt method and solution of linear least-squares problems, BIT 11 (1971) 345–368.
[2] A. Bjorck, Solving linear least-squares problems by Gram–Schmidt orthogonalization, BIT 7 (1967) 1–21.
[3] A. Bjorck, Numerics of Gram–Schmidt orthogonalization, Linear Algebra Appl. 197/198 (1994) 297–316.
[4] A. Bjorck, Numerical Methods for Least-squares Problems, SIAM, Philadelphia, PA, 1996.
[5] A. Bjorck, C.C. Paige, Loss and recapture of orthogonality in the modified Gram–Schmidt algorithm, SIAM J. Matrix Anal. Appl. 13 (1992) 176–190.
[6] J.W. Daniel, W.B. Gragg, L. Kaufman, G.W. Stewart, Reorthogonalization and stable algorithms for updating the Gram–Schmidt *QR* factorization, Math. Comp. 30 (1976) 772–795.
[7] A. Dax, Loss and retention of accuracy in affine scaling methods, Technical Report, Hydrological Service of Israel, 1999.
[8] J.W. Demmel, Applied Numerical Linear Algebra, SIAM, Philadelphia, PA, 1997.
[9] G.H. Golub, Numerical methods for solving linear least-squares problems, Numer. Math. 7 (1965) 206–216.
[10] G.H. Golub, C.F. Van Loan, Matrix Computation, Johns Hopkins University Press, Baltimore, MD, 1983.
[11] N.J. Higham, Accuracy and Stability of Numerical Algorithms, SIAM, Philadelphia, PA, 1996.
[12] W. Hoffmann, Iterative algorithms for Gram–Schmidt orthogonalization, Computing 41 (1989) 335–348.
[13] T.L. Jordan, Experiments on error growth associated with some linear least-squares procedures, Math. Comp. 22 (1968) 579–588.
[14] B.N. Parlett, The Symmetric Eigenvalue Problem, Prentice-Hall, Englewood Cliffs, NJ, 1980.
[15] J.R. Rice, Experiments on Gram–Schmidt orthogonalization, Math. Comp. 20 (1966) 325–328.
[16] A. Ruhe, Numerical aspects of Gram–Schmidt orthogonalization of vectors, Linear Algebra Appl. 52/53 (1983) 591–601.
[17] G.W. Stewart, Matrix Algorithms, vol. 1: Basic Decompositions, SIAM, Philadelphia, PA, 1998.