Reorthogonalization Methods in ABS Classes

József Abaffy

John von Neumann Faculty of Informatics, Óbuda University
Bécsi út 96/b, H-1034 Budapest, Hungary
abaffy.jozsef@nik.uni-obuda.hu

Szabina Fodor

Department of Computer Science, Corvinus University of Budapest
Fővám tér 13-15, H-1093 Budapest, Hungary
szabina.fodor@uni-corvinus.hu

Abstract: In this paper we analyze two subclasses of ABS class of methods which produce orthogonal projection vectors. We theoretically prove that the "twice is enough" selective reorthogonalization criterion of Parlett-Kahan [14] and of Hegedüs [8] can be used in the various ABS classes. Here we also provide a detailed numerical analysis of these ABS-based algorithms. We revealed that the ABS-based algorithm combined with the modified Parlett-Kahan criterion by Hegedüs provided more accurate results in the three considered cases (the rank of the coefficient matrix, the determination of the orthogonal bases, and the QR factorization) than the built-in rank and qr MATLAB functions.

Keywords: ABS methods; orthogonalization; reorthogonalization; Gram-Schmidt; Parlett-Kahan

1 Introduction

Orthogonalization is an important step of matrix calculation and produces matrices that are much easier to work with. Let $A=(a_1,...,a_n)$ be a $m\times n$ matrix ($m\geq n$) with full column rank ($rank(A)=n$). Orthogonalization yields an orthogonal basis $Q=(q_1,...,q_n)$ of $span(A)$ such that $A=QR$, where R is an upper triangular matrix. There are several approaches and algorithms for the orthogonalization of a matrix, including the Gram-Schmidt (GS) orthogonalization algorithm, as well as conceptually different approaches such as Householder transformations or Givens rotations [2], [5], [9], [10], [11].

Though the above approaches provide a theoretical foundation of orthogonalization, the calculation of the actual orthogonal basis can be problematic. Indeed, while it is critical that the computed vector $\overline{Q} = (\overline{q}_1,...,\overline{q}_n)$ is as close to the theoretical vector $Q = (q_1,...,q_n)$ as possible, this is often limited by the precision level of the computer used. The issue of limited accuracy of computing Q is known as the orthogonal basis problem [11].

Several investigators have also proposed to perform the orthogonalization several times (called "reorthogonalization"), though the benefit of repeated reorthogonalization is questionable and it seems that one reorthogonalization (i. e. two consecutive orthogonalization steps) is sufficient to ensure the orthogonality of the computed vectors at high accuracy [14], [3], and [4].

Reorthogonalization can also be performed as an optional step, depending on a certain criterion applied during the execution of the algorithm. This situation is called "selective reorthogonalization" and is based on a criterion dependent on the quality of the computed vector. A further assessment of this selective step is provided by Parlett [14] who analyzed the use of two vectors attributed to Kahan (Parlett-Kahan algorithm). Parlett showed that while two consecutive orthogonalization steps improved the accuracy of the computation, further orthogonalization steps failed to provide additional benefit, establishing the principle of "twice is enough". Recently, Hegedüs [8] provided a new theoretical basis for Parlett-Kahan's "twice is enough" algorithm and a modified reorthogonalization criterion.

In this paper we apply the classical Parlett-Kahan (PK) criterion and its modification by Hegedüs (referred to as the modified Parlett-Kahan or MPK criterion) on the ABS class of methods [1]. We considered the S1 and S4 subclasses of ABS, which generate orthogonal directions. Here we also summarize the characteristics of these ABS-based algorithms, and describe their implementation using Matlab R2007b. The numerical experiments revealed that the calculations of the rank of a matrix and of the orthogonal vectors, as well as the QR factorization are more accurate with the usage of our ABS based algorithms than the functions implemented in Matlab.

Finally we should emphasize that the ABS-based algorithms are easy to parallelize. This feature expands the practical usefulness of our algorithms. The results presented in this paper may provide important novel aspects of the efficient parallel implementation of matrix calculations.

2 The Parlett-Kahan and Modified Parlett-Kahan Algorithms

The twice-is-enough algorithm of Parlett and Kahan is based on the following orthogonalization step. Let z be the vector to be orthogonalized. The equation

$$p = \left(I - \frac{yy^T}{\|y\|^2} \right) z = \text{orth}(y, z) \tag{1}$$

is the exact orthogonalization of z. An undesirable feature of the Gram-Schmidt method is that in the presence of rounding errors the vector p can be far from orthogonal to y. This loss of orthogonality is signaled by cancellation, which magnifies previous rounding errors, which in turn will generally contain components in y. A solution for this problem is to repeat the procedure on the vector p.

It has been observed that one reorthogonalization is usually sufficient to produce a vector that is orthogonal to working accuracy - i.e., "twice is enough".

Obviously in reality we have only a numerical approximation of p, say x'. Let the error $e' \equiv x' - p$ satisfy $\|e'\| = \varepsilon_M \|z\|$, where ε_M is the machine precision unit and let κ be any fixed value in the range $\left[\frac{1}{0.83 - \varepsilon_M}, \frac{0.83}{\varepsilon_M} \right]$[14].

Algorithm-1 ***Parlett-Kahan algorithm - (PK)*** **[14]**

Calculate $x' = \text{orth}(y, z)$, where $\text{orth}(*)$ is given in (1).

Case 1: If $\|x'\| \geq \|z\| / \kappa$ then accept $x = x'$ and $e = e'$,

otherwise compute $x'' = \text{orth}(y, x')$ with error

$$e'' \equiv x'' - \left(I - \frac{yy^T}{\|y\|^2} \right) x'$$

satisfying $\|e''\| = \varepsilon_M \|x'\|$ and go to **Case 2.**

Case 2: If $\|x''\| \geq \|x'\| / \kappa$ then accept $x = x''$ and $e = e'' - p$.

Case 3*:* If $\left\|x^{''}\right\| < \left\|x^{'}\right\| / \kappa$ then accept $x = 0$ and $e = -p$.

Remark 2.1 *The vector* x *computed by the algorithm ensures that* $\|e\| \le (1 + \frac{1}{\kappa})\varepsilon_M \|z\|$ *and* $\left|y^T x\right| \le \kappa \cdot \varepsilon_M \|y\|\|x\|$ *[14].*

Hegedüs [8] reformulated the original Parlett-Kahan algorithm and proved the improvement of orthogonality and gave new criteria for the cancellation due to calculation and estimation for the accuracy of computation.

Algorithm-2 ***modified Parlett-Kahan algorithm - (MPK)*** *[8]*

Here we consider one step of Gram-Schmidt orthogonalization with respect to cancellation.

Let $Q = (q_1, q_2, \ldots, q_{k-1}) \in \Re^{n\times(k-1)}$ and $a \in \Re^n$ be known and accurate. Vector a is orthogonalized to the subspace spanned by the orthonormal columns of matrix Q with one Gram-Schmidt step

$$\theta_k q_k = (I - QQ^T)a, \tag{2}$$

where θ_k is the norm of $\|a\|$. Compute

$$\eta = \frac{\theta_k}{\|a\|}.$$

If $\eta < \eta_{\min}$ then linear dependency has been detected, i.e. vectors $a, q_1, q_2, \ldots, q_{k-1}$ are linearly dependent at least computationally

If $\eta \ge \eta_{\max}$ then q_k is accepted, otherwise perform a reorthogonalization step.

$$\theta_k \hat{q}_k = (I - QQ^T)q_k$$

where θ_k is the norm of $\|q_k\|$. The vector $\hat{q}_k$ is accepted, and update $\eta_{\min} = \|Q^T q_k\|$.

Remark 2.2 *The initial value for* $\eta_{\min}$ *is* $4 \cdot \varepsilon_M$ *and Hegedüs proved that* $-\log_{10}\eta_{\min}$ *is the number of accurate digits.*

Remark 2.3 *Hegedüs showed that when the incoming vectors are exact and there are accurate digits in the computation, then one may expect the fulfillment of condition* $\eta_{\max} \le \eta$ *after the second orthogonalization step at most. The largest*

choice of $\eta_{\max}$ *is* $1/\sqrt{2}$ *to fulfill the condition. Hence the resulting vector* $\hat{q}_k$ *can be considered orthogonal to* $q_1, q_2, \ldots, q_{k-1}$ *up to computational accuracy [8].*

3 The Symmetric Scaled ABS Algorithm with Reorthogonalization

In this section, we briefly present the scaled symmetric ABS algorithm [1] and we apply reorthogonalization on the projection vectors (p_i) of these subclasses. The ABS algorithm was developed to solve systems of linear and non-linear equations. However, the ABS-based algorithms can be used for many other purposes, for example we can use these algorithms to compute an orthogonal basis of $span(A)$.

Instead of the original equation $Ax = b$, where $A \in \Re^{m,n}$, $b \in \Re^m$, $x \in \Re^n$, consider the scaled equations

$$V^T Ax = V^T b \tag{3}$$

where, $V = (v_1, \cdots, v_m) \in \Re^{m,m}$,is a nonsingular matrix. The set of the solutions of the equations (3) is the same as the set of the solution of $Ax = b$. Applying the non-scaled basic symmetric ABS algorithm for solving (3), we can obtain the symmetric scaled ABS algorithm. Denote the residual vector by $r(x) = Ax - b$. We remark here that not all orthogonalization processes need the residual vector.

Algorithm-3 ***Symmetric scaled ABS algorithm with reprojection***

Step1 - Initialization

Let $x_1 \in \Re^n$ arbitrary, $H_1 = I \in \Re^{n,n}$, where I is the unit matrix, $i = 1$, and $iflag = 0$.

Step2

Let $v_i \in \Re^m$ be arbitrary provided that $v_1, \cdots, v_i$ are linearly independent. Compute the residual error vector r_i.

If $r_i = 0$ then stop; x_i solves the equations

otherwise compute

$$\tau_i = v_i^T r_i$$
$$s_i = H_i A^T v_i \,.$$

Step3 - Linear dependency check

If $s_i \neq 0$ then go to **Step4.**

If $s_i = 0$ and $\tau_i = 0$ then set

$$x_{i+1} = x_i$$
$$H_{i+1} = H_i$$
$$iflag = iflag + 1$$

and if $i < m$ then go to **Step2** otherwise stop; x_i solves the equations.

If $s_i = 0$ and $\tau_i \neq 0$, stop set $iflag = -i.$ (*Incompatibility*).

Step4 - Compute the search direction p_i by

$$p_i = H_i^T z_i$$

where $z_i \in \Re^n$ is arbitrary saving for $z_i^T H_i A^T v_i \neq 0$. Compute the reorthogonalization step

$$p_i = H_i^T p_i \,.$$

Step5 - Update the approximate solution by

$$x_{i+1} = x_i - \alpha_i p_i$$

where the step size α_i is given by $\alpha_i = \frac{\tau_i}{v_i^T A p_i}$.

If $i = m$ then stop; x_{m+1} is the solution of the equations.

Step6 - Update the projection matrix H_i

$$H_{i+1} = H_i - \frac{H_i A^T v_i \cdot v_i^T A H_i}{v_i^T A H_i \cdot H_i A^T v_i} \,. \tag{4}$$

Step7 - Set $i = i + 1$ and go to **Step2**.

Remark 3.1 *The projection vectors p_i are orthogonal [1].*

Remark 3.2 *Observe that **Step5** is not always needed if we only wish to solve the linear system of equations.*

Remark 3.3 *Note that the denominator of the projection matrix (H_i) is non-zero because of **Step3**.*

Theorem 3.1 *Define $\overline{A}^i$ as*

$$\overline{A}^i = \left(\frac{H_1 A^T v_1}{\left\| H_1 A^T v_1 \right\|}, \ldots, \frac{H_i A^T v_i}{\left\| H_i A^T v_i \right\|} \right). \tag{5}$$

The $\overline{A}^i$ matrices are full rank. Moreover, if $H_1 = I$, then the columns of $\overline{A}^i$ are mutually orthogonal [1].

Remark 3.4 *Using the notation of (5) we have the following alternative formula for (4)*

$$H_{i+1} = I - \overline{A}^i \cdot \overline{A}^{iT}. \tag{6}$$

Remark 3.5 *Note that the choices of the $z_i \in \Re^n$ and the $v_i \in \Re^m$ are arbitrary saving for $z_i^T H_i A^T v_i \neq 0$. We considered two subclasses of the symmetric, scaled ABS algorithm, designated the **S1** and **S4** subclasses where the search vectors p_i are orthogonal.*

3.1 Reorthogonalization in Symmetric, Scaled ABS Algorithm using the Idea of the PK and Modified PK Algorithms

In this section we use the original Parlett-Kahan and the modified Parlett-Kahan algorithms in the scaled symmetric ABS algorithms. We only describe the **Step4** of symmetric scaled ABS algorithm where we determine the searching vectors as the other steps correspond to the steps of the original algorithm.

The ABS Parlett-Kahan algorithm is based on the following orthogonalization step.

Let z be a vector to be orthogonalized. Then

$$p_0 = \left(H - \frac{p \cdot p^T}{\left\| p \right\|^2} \right) z \tag{7}$$

where p is the accurate vector computed at the ith step and p_0 is the exact

orthogonalization of z. Obviously we can have an approximation of p_0 only, say x'. Let the error $e' \equiv x' - p_0$ satisfy $\|e'\| = \varepsilon_1 \|z\|$ [16].

Algorithm-4 ***Reorthogonalization in ABS with the Parlett-Kahan algorithm (ABS-PK)***

Step4-PK Compute the search direction p_i by

$$p_i = H_i^T z_i$$

where $z_i \in \Re^n$ is arbitrary saving for $z_i^T H_i A^T v_i \neq 0.$

If $\|p_i\| \geq \frac{\|z_i\|}{\kappa}$ then accept p_i and using the notation of the original Parlett-Kahan $x = p_i$

otherwise compute

$$\hat{p}_i = H_i^T p_i$$

If $\|\hat{p}_i\| \geq \frac{\|p_i\|}{\kappa}$

then accept $\|\hat{p}_i\|$ i.e. $p_i = \hat{p}_i$ and using the notation of the original Parlett-Kahan $x = p_i = \hat{p}_i$

otherwise linear dependency is detected

$$\begin{aligned} p_i &= 0 \\ x_{i+1} &= x_i \\ H_{i+1} &= H_i \\ iflag &= iflag + 1 \end{aligned}$$

go to **Step2.**

Lemma 3.1.1 *The vector* x *computed by the algorithm ensures that* $\|e\| \leq (1+1/\kappa)\varepsilon_M \|z\|$ *and* $|p^T x| \leq \kappa \cdot \varepsilon_M \|p\| \|x\|.$

Proof We recall Parlett's proof for the case of our orthogonalization algorithm. Differences are only in those steps where the orthogonalization expression is explicitly used. We consider the two cases:

Case 1:

$$\|e\| = \|e^{'}\| \leq \varepsilon_1 \|z\|$$

$$|p^T x| = |p^T x^{'}| = |p^T e^{'}| \leq \|p\| \|e^{'}\| \leq \varepsilon_1 \|p\| \|z\| \leq \kappa \varepsilon_1 \|p\| \|x\|$$

because $|p^T p_0| = 0$ and it follows $\|z\| \leq \kappa \|x^{'}\| = \kappa \|x\|$

Case 2:

$$|p^T x| = |p^T x^{"}| = |p^T e^{"}| \leq \|p\| \|e^{"}\| \leq \varepsilon_2 \|p\| \|x^{'}\| = \varepsilon_2 \|p\| \|x\|$$

by the definition of $x^{'}$. Now

$$\|e\| = \|x^{"} - p\| = \|e^{"} + (H - p \cdot p^T / \|p\|^2) x^{'} - p_0\|$$

where we used the definition of $x^{"}$ From this we get using the definition of $x^{'}$ and that $|p^T p_0| = 0$ that

$$\|e\| \leq \|e^{"}\| + \|H\| \|(I - p \cdot p^T / \|p\|^2)(e^{'} + p_0) - p_0\|$$

$$\leq \varepsilon_2 \|x^{'}\| + \|(I - p \cdot p^T / \|p\|^2) e^{'}\|$$

therefore

$$\leq \varepsilon_2 \|x^{'}\| + \|e^{'}\| \leq \varepsilon_2 \|x^{'}\| + \varepsilon_1 \|z\| \leq \max(\varepsilon_1, \varepsilon_2)(1 + 1/\kappa) \|z\|$$

because the reorthogonalization has to be performed when $\|x^{'}\| < \|z\| / \kappa$. ■

Theorem 3.1.2 *The* $\|x^{"} - p_o\| \leq \|x^{'} - p_o\|$, *where* p_0 *is the exact orthogonalization of* z.

Proof Using the definition of $x^{"}$ we get

$$\|x^{"} - p_o\| = \|H\left(I - pp^T / \|p\|^2\right) x^{'} - p_o\|$$

$$= \left\| H\left(I - pp^T / \|p\|^2\right)(x' - p_o) \right\| \le \left\| (x' - p_o) \right\|$$

where we used the orthogonality of p and p_o and

$$\left\| H\left(I - pp^T / \|p\|^2\right) \right\| \le \|H\| \left\| I - pp^T / \|p\|^2 \right\| = 1. \blacksquare$$

Theorem 3.1.3 $\left\| x'' - p \right\| \le \left\| x' - p \right\|$ *where* p *is the accurate orthogonal vector of the Parlett-Kahan algorithm.*

Proof If $H = I$ and $p = y$ of ABS Parlett-Kahan algorithm, then the theorem follows from the Parlett-Kahan algorithm. ■

Remark 3.1.1 *It is worthwhile to emphasize that the ABS-Parlett-Kahan algorithm is different from the original one, because of the existence of the projection matrix* H *. The projection matrix ensures the new orthogonal vector* p_i *is orthogonal for all previous orthogonal vectors* $p_1, ..., p_{i-1}$.

For this algorithm the same lemma is valid as for the original Parlett-Kahan algorithm. It is enough to note that the Euclidean norm of the H_i projection matrices is equal to one. Therefore we recall only the lemma of the algorithm (see [14]) without proof.

Lemma 3.1.4 *The vector* x *computed by the algorithm ensures that* $\|e\| \le (1 + 1/\kappa)\varepsilon \|z\|$ *and* $\left| p^T x \right| \le \kappa \varepsilon \|p\| \|x\|$.

Algorithm-5 ***Reorthogonalization in ABS with the modified Parlett-Kahan algorithm (ABS-MPK)***

Step4-MPK Compute the search direction p_i by

$$\theta_i p_i = H_i^T z_i$$

where $z_i \in \Re^n$ is arbitrary provided that $z_i^T H_i A^T v_i \neq 0$ and $\theta_i = \left\| H_i^T z_i \right\|$.

Compute

$$\eta = \frac{\theta_i}{\|z_i\|}$$

and

$$\eta_{\min} = \frac{\left\| \overline{A}^{i} p_i \right\|}{\left\| p_i \right\|}$$

If $\eta < \eta_{\min}$

then linear dependency is detected

$$p_i = 0$$
$$x_{i+1} = x_i$$
$$H_{i+1} = H_i$$
$$iflag = iflag + 1$$

go to **Step2.**

If $\eta \geq \eta_{\max}$

then p_i is accepted

otherwise a reorthogonalization is needed

$$\theta_i p_i = H_i^T p_i$$

where $\theta_i = \left\| H_i^T p_i \right\|$.

Remark 3.1.2 *We should emphasize that the computational demands of checking the computational linear dependency is very low. It is sufficient to check the value of the computed* η *and* $\eta_{\min}$.

4 Numerical Experiments

Next we were interested in numerical features of the different ABS algorithms. To this end, two variants of the scaled symmetric ABS class of algorithms were implemented in MATLAB version R2007b [12].

S1 (Subclass-S1 of ABS): z_i is selected such that

$$z_i = A^T v_i, \text{ where } v_i = e_i \text{ the } ith \text{ unit vector}.$$

S4 (Subclass-S4 of ABS): z_i is selected such that

$$z_i = r_i, \text{ where } r_i \text{ is the } ith \text{ residual vector}.$$

As we did not want to check the accuracy of the solution we defined it for the sake of minimizing the error of the residual vector calculation in the S4 subclass as $(1,1,...1)$.

The below experiments were performed on a Toshiba personal computer with Intel i7-2620M CPU with integrated graphics, 8 GB RAM and 450 GB hard disk drive, running Microsoft Windows 7 Professional and MATLAB version R2007b. No software other than the operating system tasks, MATLAB and ESET NOD32 antivirus were running during the experiments.

The experiments testing the calculated rank of the coefficient matrix A, the orthogonal deviation
$(-\log_{10}(\max(\max(\ abs(I-QQ^T))))$, and the error of QR factorization
$(-\log_{10}(\max(\max(\ abs(A-QR))))$ were performed.

First we tested the different ABS based algorithms on randomly generated dense, symmetric, positive definite, full rank matrices (*SPD*). The random matrices were generated using MATLAB.

We use the following annotations:

ABS-S1 (AREO): symmetric scaled ABS algorithm with S1 selection, reprojection in every step, linear dependency check according to Hegedüs.

ABS-S1 (PK): symmetric scaled ABS algorithm with S1 selection and reorthogonalization using the Parlett-Kahan algorithm.

ABS-S1 (MPK): symmetric scaled ABS algorithm with S1 selection and reorthogonalization using the modified Parlett-Kahan algorithm.

ABS-S4 (AREO): symmetric scaled ABS algorithm with S4 selection, reprojection in every step, linear dependency check according Hegedüs.

ABS-S4 (PK): symmetric scaled ABS algorithm with S4 selection and reorthogonalization using the Parlett-Kahan algorithm.

ABS-S4 (MPK): symmetric scaled ABS algorithm with S4 selection and reorthogonalization using the modified Parlett-Kahan algorithm.

As shown in Fig. 1, there were no significant differences between the accuracy of the calculations for S1 and S4 variants of ABS algorithms. This was in contrast to our expectations based on the fact that the z_i vectors in the selection S4 were calculated as a function of the row vectors of the matrix A. The likely reason for the discrepancy is that the well-chosen solution vector minimizes the rounding error even if the approximation of the solution in step i is not exact. The difference between the variants of ABS based algorithms is in the calculated rank.

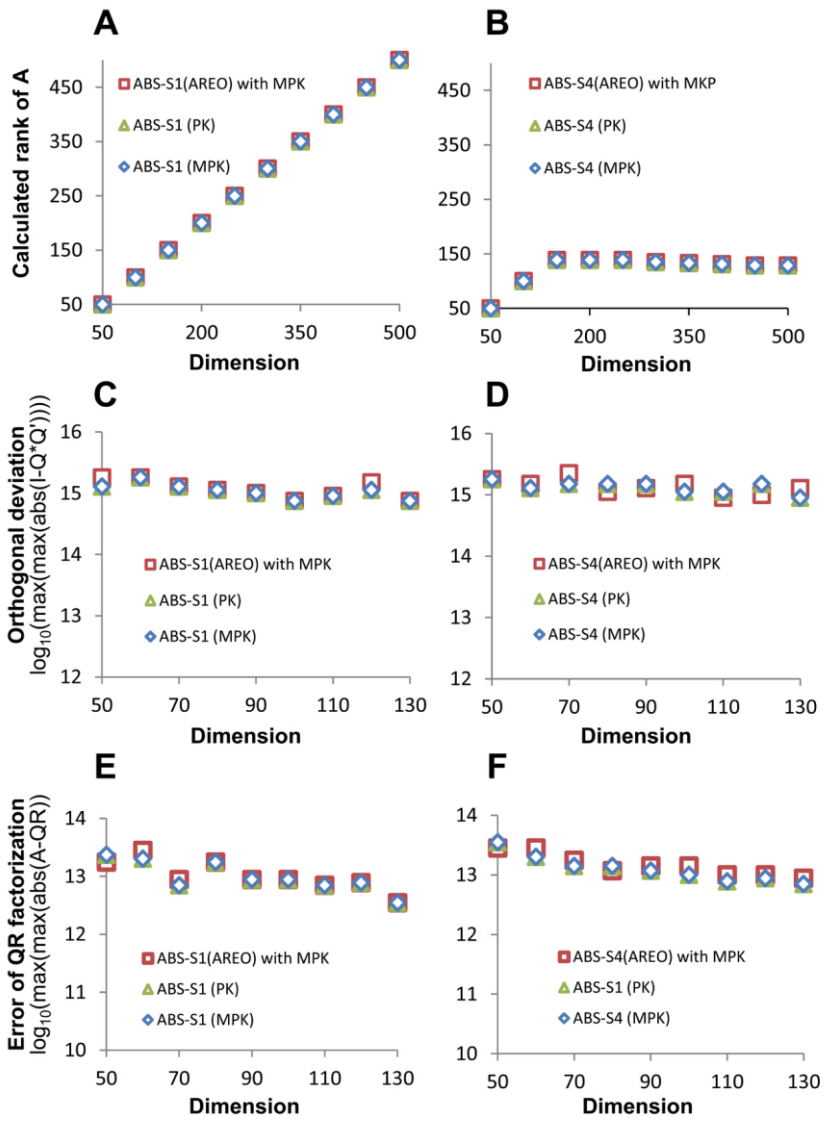

Figure 1
Comparison of different ABS-based algorithms on randomly generated, dense, symmetric, positive definite, full rank matrices

All algorithms based on S4 selection detect incorrect linear dependency in the projection vectors because of the loss of accurate digits in the computation of orthogonal vectors. If the judgment is made on aspects of finding an orthogonal basis between the two subclasses of ABS based algorithms then algorithms with S1 selection outperform the subclass S4. We should note that the subclass S4 has many valuable numerical features [1] and further studies are needed to test them.

Next we compared our best-performing ABS-based algorithm (**ABS-S1 (AREO)**) with other well-known algorithms. We selected the built-in **qr** function of MATLAB and the classic Gram-Schmidt orthogonalization (**CGS**) algorithm [2]. We implemented the CGS algorithm with reorthogonalization in every step and the linear dependency check proposed by Hegedüs.

We should mention that the formula (6) gives a connection between the CGS and the ABS-S1 methods. The ABS-based algorithm with S1 selection is practically the block version of the CGS method. Numerically we could not see a significant difference between the block and the original version of update of the projection matrix H_i (data not shown). However, it is worthwhile to do further such research in the future with alternative formulas of updating the projection matrices.

We tested the algorithms on randomly generated dense, symmetric, positive definite, full rank matrices (*SPD*), Pascal (*Pascal*), normalized Pascal (*normalized Pascal*), Vandermonde (*Vandermonde*), and normalized Vandermonde (*normalized Vandermonde*) matrices. The matrices were generated using built-in MATLAB functions.

It should be mentioned that the ABS-based algorithms use the row of the matrices in the *ith* step while the **CGS** and the **qr** algorithms use the columns of the A. To correct for that discrepancy, the transposed matrix A has been applied on the latter two algorithms.

As shown in Fig. 2, we compared the computed rank of the different matrices. Note that when the condition number of the coefficient matrix (A) is not extremely high, like for the SPD matrices, then all our algorithms calculated the same, exact rank of matrices (data not shown). However, we could see a significant difference on the computed rank of special (like Pascal, normalized Pascal) matrices, and we wanted to verify the actual rank of the matrices represented in double precision. We used the Multiprecision Computing Toolbox of MATLAB [13] and we recomputed the rank of the matrices originally represented in double precision into quadruple precision. We could not see any loss of the ranks because of the non-exact representations.

After all, we conclude that the MATLAB rank function performed poorly on computing the rank of special matrices, and only the ABS-S1 algorithm gave the exact rank for the normalized matrices A. Therefore, it would be worth using the ABS-S1 algorithm for calculating the rank of matrices.

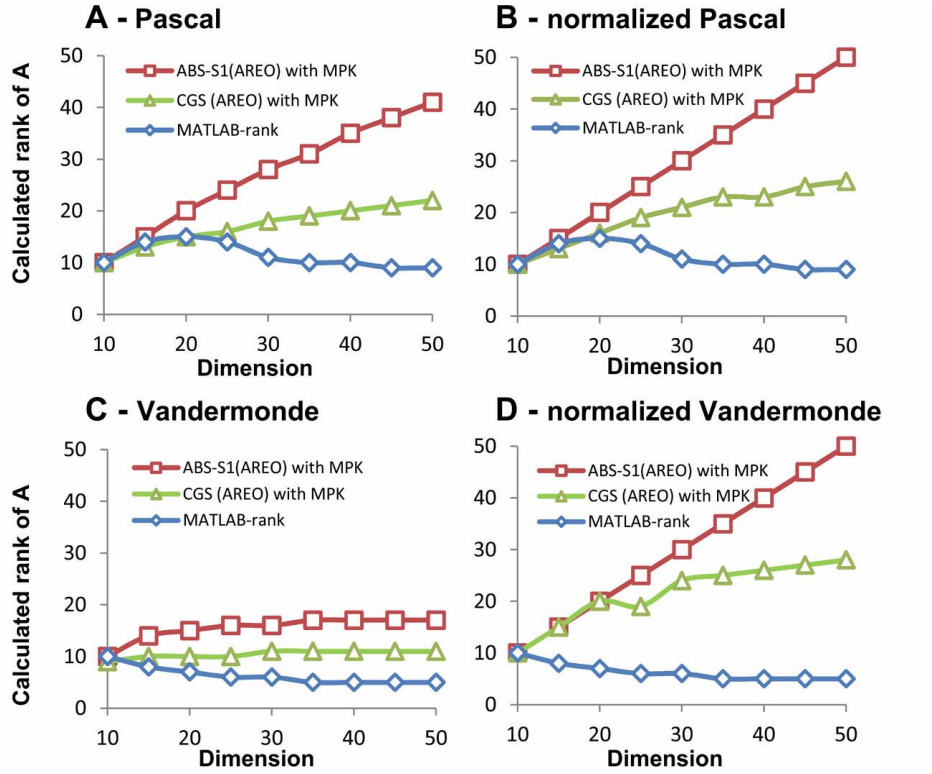

Figure 2
Comparison of the calculated rank of the coefficient matrices

Then we compared the accuracy of the computed orthogonal basis with our algorithms. To describe the deviation of orthogonal deviation, we use the $-\log_{10}(\max(\max(\ abs(I - Q \cdot Q^T))))$ formula where I is the unit matrix and Q is the orthogonal basis. Hegedüs proved [8] that this formula gives the number of accurate digits. Using double precision number presentation (IEEE 754 standard [15]), the maximum value of the formula is less than *16*.

As shown in Fig. 3, the MATLAB qr function computed the orthogonal basis the least accurately for every test case. There is no significant difference between the ABS-S1 and the CGS algorithms for the difficult test problems (like Pascal, Vandermonde). However, the CGS algorithm surpasses the ABS-S1 on well-conditioned matrices like our SPD matrices.

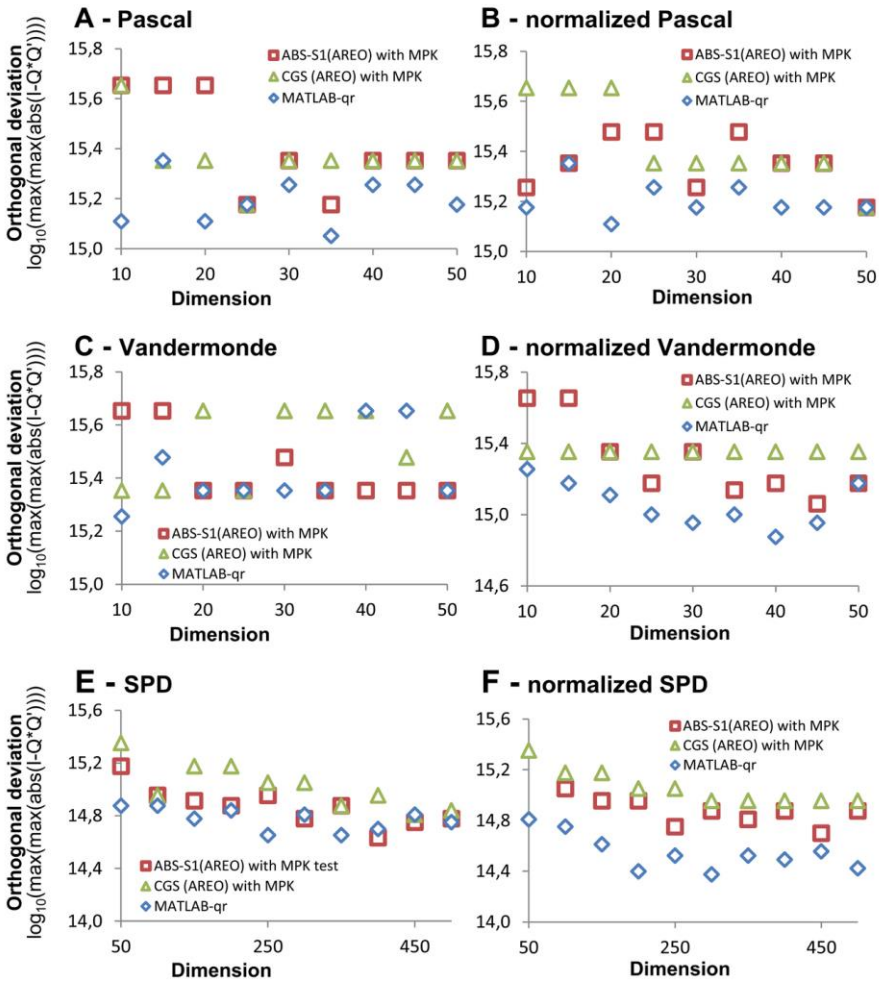

Figure 3
Comparison of the orthogonal deviation

Last we tested the accuracy of the QR factorization. We used a formula $(-\log_{10}(\max(\max(\ abs(A-QR))))$ for describing the precision of QR factorization, which gives the number of accurate digits of the factorization. The maximum value of the accurate digits is less than 16 in floating point representation and larger values mean more precise results.

As shown in Fig. 4 the CGS algorithm surpasses the ABS-S1 and the **qr** algorithms on every case, being the most salient on the normal test cases like SPD and normalized SPD matrices. No significant difference can be seen between the ABS-S1 algorithm and the MATLAB **qr** function.

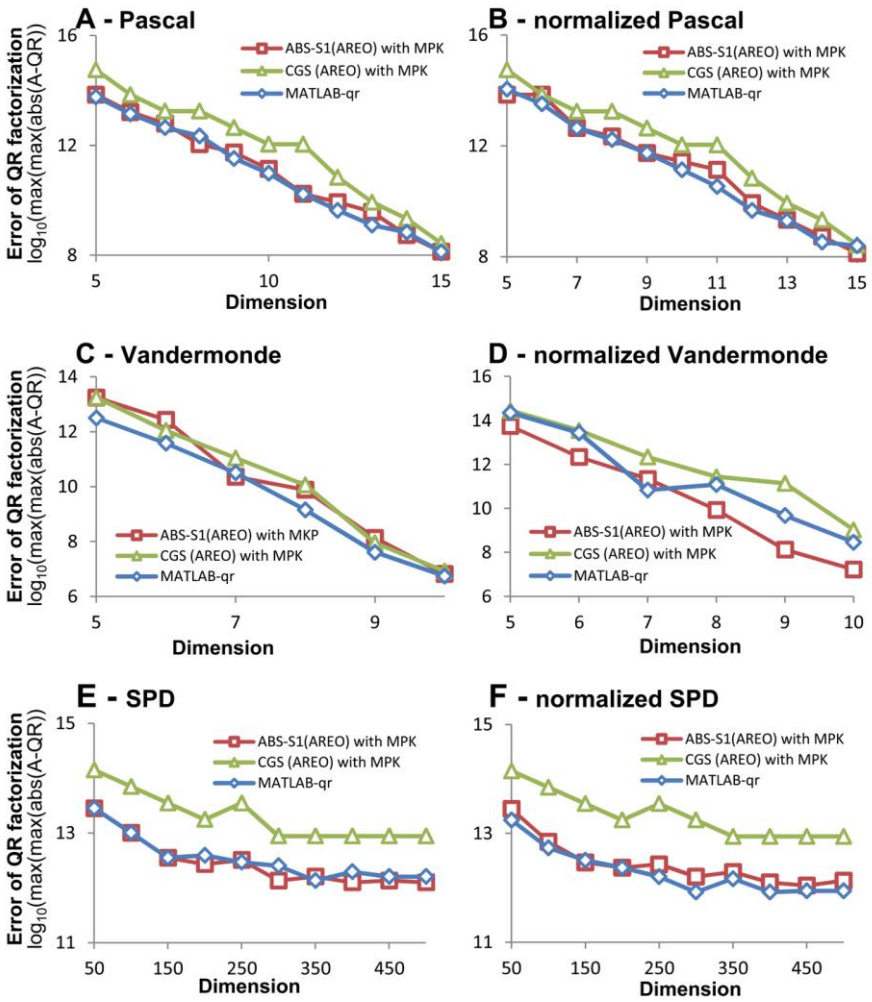

Figure 4
Comparison of the QR factorization

However, we should mention that we compared the accuracy of QR factorization in the range where neither of the tested algorithms found linear dependency. This strictly narrowed down the examined dimension of matrices.

It should be noted that we did not test the algorithms with pivoting the coefficient matrices (A). The different pivoting strategies are numerically very expensive and the accuracy of the calculations (at least 10^{-14}) did not substantiate these efforts. However, further analyses will need to be performed for testing the effects of the pivoting strategies.

Discussion and Conclusions

Here we presented new ABS based algorithms for creating orthogonal basis. We numerically verified that the modified Parlett-Kahan (MPK) reorthogonalization criterion by Hegedüs can be successfully used. Our numerical experiments revealed that the ABS-based algorithm combined with the modified Parlett-Kahan provided more accurate results in the three considered cases (the rank of the coefficient matrix, the determination of the orthogonal bases, and the ***QR*** factorization) than the built-in MATLAB functions.

Acknowledgement

The authors thank Csaba Hegedüs for inspiring discussions and helpful suggestions.

References

[1] Abaffy, J., and Spedicato, E., "*ABS Projection Algorithms: Mathematical Techniques for Linear and Nonlinear Equations*", Ellis Horwood Limited, John Wiley and Sons, Chichester, England (1989)

[2] Björck, A Numerical Methods for Least Squares Problems. SIAM, Philadelphia, PA, 1996

[3] Giraud, L.,Langou, J., and Rozloznik, M "On the Round-off Error Analyisis of the Gram-Schmidt Algorithm with Reorthogonalization, CERFACS Technical Report No. TR/PA/02/33, pp. 1-11 (2002)

[4] Giraud, L., Langou, J., and Rozloznik, M. "The Loss of Orthogonality in the Gram-Schmidt Orthogonalization Process", Computers and Mathematics with Applications, Vol. 51, pp. 1069-1075, 2005

[5] Golub, G., H, and Van Loan, C., F Matrix Computations, 3rd ed. John Hopkins Univ. Press, Baltimore, MD (1996)

[6] Daniel, J., W., Gragg, W., B., Kaufman, L., and Stewart, G., W. *"Reorthogonalization and Stable Algorithms for Updating the Gram-Scmidt QR Factorization"*, Mathematics of Computation, Vol. 30, No. 136, pp. 772-795 (1976)

[7] Hegedüs, C. J. "*Short Proofs for the Pseudoinverse in Linear Algebra"*, Annales Univ. Sci. Budapest, 44, 115-121 (2001)

[8] Hegedüs, C. J. "*Reorthogonalization Methods Revisited*", Acta Polytechnica, submitted

[9] Hegedüs, C. J. "Generating Conjugate Directions for Arbitrary Matrices by Matrix Equations. I, II", Computers and Mathematics with Applications 21:(1) pp. 71--85, 87--94-71. (1991)

[10] Hegedüs C. J., Bodócs L. "General Recursions for A-Conjugate Vector Pairs", Report No. 1982/56 Central Research Institute for Physics, Budapest

[11] Higham, N. J. *"Accuracy and Stability of Numerical Algorithms"*, SIAM, Philadelphia, (1996)

[12] Guide MATLAB User's, The MathWorks Inc, Natick, MA, 2007

[13] Multiprecision Computing Toolbox User's Manual, Advanpix's, Yokohama, Japan.
http://www.advanpix.com/documentation/users-manual/

[14] Parlett, B., N. "*The symmetric Eigenvalue Problem"*, Englewood Cliffs, N. J. Prentice-Hall (1980)

[15] Smith W., S. "Fixed versus Floating Point". The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Pub. p. 514, ISBN 0966017633, 1997 Retrieved December 31, 2012

[16] Wilkinson, J. H. "Rounding Errors in Algebraic Processes", Prentice-Hall (1963)