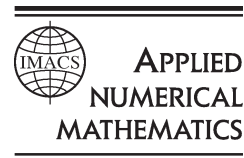




ELSEVIER

Applied Numerical Mathematics 30 (1999) 155–173



# A new family of block methods<sup>☆</sup>

U. Meier Yang<sup>a,\*</sup>, K.A. Gallivan<sup>b,1</sup>

<sup>a</sup> 1235 Bracebridge Ct., Campbell, CA 95008, USA

<sup>b</sup> Computational Science and Engineering Department, Florida State University, Tallahassee, FL 32306, USA

---

## Abstract

A new family of iterative block methods, the family of block EN-like methods, is introduced. Efficient versions are presented and computational complexity, memory requirements and convergence properties are investigated. Finally, leading evidence of the potential of the new family is demonstrated via a comparison of numerical results and performance to other block methods. © 1999 Elsevier Science B.V. and IMACS. All rights reserved.

**Keywords:** Block methods; Quasi-Newton methods; Multiple right-hand sides

---

## 1. Introduction

Large sparse linear systems with multiple right-hand sides occur in many applications, such as electromagnetics (see [7] for a review), and there is a need for efficient solvers. A variety of methods have been introduced in the last twenty years, such as block CG and block BiCG [5], block GMRES [11], single-seed methods [7], a hybrid block GMRES scheme [8] and block Quasi-Newton methods [6]. This paper generalizes a new family of iterative solvers for linear systems with single right-hand sides, the family of EN-like methods [3,4], to solve systems with multiple right-hand sides. This new family of block EN-like methods uses an approximation  $H_k$  of the inverse of the matrix  $A$  of the linear system to be solved and updates it with rank- $s$  updates during each iteration step, where  $s$  is the number of right-hand sides. The block versions have an increased data locality compared to the single right-hand side versions. The methods have theoretically finite termination, but in order to reduce memory requirements and computational complexity we will also consider their restarted and truncated versions. The new family is shown to have potential via numerical experiments comparing it to other block methods such as block GMRES, block Broyden methods and a new hybrid version of block GMRES.

We introduce the family of block EN-like methods in Section 2, focus on one of its members, the block EN method, in Section 3. Section 4 describes more efficient versions of the methods, including an efficient version of block GCR, which can be shown to be related to block EN, and presents computational

---

<sup>☆</sup> This research was supported in part by the National Science Foundation under Grants CCR-9120105 and CCR-9796315.

\* Corresponding author. E-mail: yang@math.fsu.edu.

<sup>1</sup> E-mail: gallivan@cs.fsu.edu.

complexities as well as memory requirements. In Section 5, convergence properties of the block EN-like methods are examined. Finally, in Section 6, numerical results are presented.

Throughout this paper, we assume  $s$  right-hand sides and a system  $\mathcal{A}X = B$  of order  $n$ , and we use large Roman letters to denote  $n \times s$ -matrices, large Greek letters denote  $s \times s$ -matrices and large calligraphic type style letters denote  $n \times n$ -matrices, e.g., the matrix  $\mathcal{A}$  of the linear systems or the approximation  $\mathcal{H}_k$  of  $\mathcal{A}^{-1}$ .

## 2. The family of block EN-like methods

In [3,4], a new family of iterative methods was introduced: the family of EN-like methods. In order to solve the linear system with  $s$  right-hand sides

$$\mathcal{A}X = B, \quad (1)$$

we propose block versions of the EN-like methods that are based on a generalization of the single right-hand side methods. Instead of updating  $\mathcal{H}_k$ , the approximation to  $\mathcal{A}^{-1}$  via a rank-one update, the block methods use a rank- $s$  update, while simultaneously improving an approximation  $X_k$  to the solution of the linear system by adding the estimate of the error,  $\mathcal{H}_{k+1}R_k$ , where  $R_k := B - \mathcal{A}X_k$  is the block residual. The choice of the actual rank- $s$  update is motivated by Broyden's method.

The general block EN-like method is given by Algorithm 1.

**Algorithm 1.** Block EN-like method

**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ ,  $\mathcal{E}_0 = \mathcal{I} - \mathcal{A}\mathcal{H}_0$ .

**For**  $k = 0, 1, \dots$ :

$$\begin{aligned} \tilde{U}_k &= H_k \mathcal{E}_k R_k, \\ V_k &= F_k (F_k^H \mathcal{A} \mathcal{H}_k R_k)^{-H}, \\ \mathcal{H}_{k+1} &= \mathcal{H}_k + \tilde{U}_k V_k^H, \\ \mathcal{E}_{k+1} &= \mathcal{I} - \mathcal{A} \mathcal{H}_{k+1}, \\ X_{k+1} &= X_k + \mathcal{H}_{k+1} R_k, \\ R_{k+1} &= \mathcal{E}_{k+1} R_k \end{aligned}$$

**end.**

Here  $F_k$  needs to be chosen so that  $F_k^H \mathcal{A} \mathcal{H}_k R_k$  is nonsingular.

The family of block EN-like methods is related to the family of block Broyden methods recently introduced by O'Leary and Yeremin [6]. The block Broyden method can be defined in two ways, through approximating  $\mathcal{A}$  by a matrix  $\mathcal{B}_k$  or through approximating  $\mathcal{A}^{-1}$  by a matrix  $\mathcal{H}_k$ . We concentrate here on the second approach.

**Algorithm 2.** Block Broyden method (with  $\mathcal{H}_k$  approximating  $\mathcal{A}^{-1}$ )

**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ .

**For**  $k = 0, 1, \dots$ :

$$P_k = \mathcal{H}_k R_k,$$

$$\begin{aligned}
Q_k &= \mathcal{A}P_k, \\
X_{k+1} &= X_k + P_k \Phi_k, \\
R_{k+1} &= R_k - P_k \Phi_k, \\
\mathcal{H}_{k+1} &= \mathcal{H}_k + (P_k - \mathcal{H}_k Q_k)(F_k^H Q_k)^{-1} F_k^H
\end{aligned}$$

**end.**

Here  $F_k$  needs to be chosen so that  $F_k^H Q_k$  is nonsingular.

There are several possibilities from which to choose the  $s \times s$ -parameter matrix  $\Phi_k$ . The simplest, and typically poorly performing, choice is the  $s \times s$ -unity matrix  $I_s$ . A better choice is  $\Phi_k = (F_k^H Q_k)^{-1} F_k^H R_k$ , which is a generalization of the optimal line search principle suggested in [1] for Broyden's method for linear systems with a single right-hand side.

Algorithm 1a rewrites Algorithm 1 to resemble Algorithm 2 more closely.

**Algorithm 1a.** Block EN-like method

**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ .

**For**  $k = 0, 1, \dots$ :

$$\begin{aligned}
P_k &= \mathcal{H}_k R_k, \\
Q_k &= \mathcal{A}P_k, \\
\mathcal{H}_{k+1} &= \mathcal{H}_k + (P_k - \mathcal{H}_k Q_k)(F_k^H Q_k)^{-1} F_k^H, \\
\tilde{P}_k &= \mathcal{H}_{k+1} R_k, \\
\tilde{Q}_k &= \mathcal{A}\tilde{P}_k, \\
X_{k+1} &= X_k + \tilde{P}_k, \\
R_{k+1} &= R_k - \tilde{Q}_k
\end{aligned}$$

**end.**

Here  $F_k$  needs to be chosen so that  $F_k^H Q_k$  is nonsingular.

This form of the block EN-like method clearly shows that the block EN-like method is related to the block Broyden method as the Gauss–Seidel method is to the Jacobi method. The new approximation,  $\mathcal{H}_{k+1}$ , is used in the evaluation of  $X_{k+1}$  instead of  $\mathcal{H}_k$  as is in the block Broyden method. Additionally, one can show the following relationship between one step of the block EN-like method and two steps of the block Broyden method.

**Lemma 1.** *One step of an EN-like method can be expressed:*

$$\tilde{X}_{k+1} = X_k + \mathcal{H}_k R_k, \tag{2}$$

$$\tilde{R}_{k+1} = R_k - \mathcal{A}\mathcal{H}_k R_k, \tag{3}$$

$$X_{k+1} = \tilde{X}_{k+1} + \mathcal{H}_k \tilde{R}_{k+1} \Psi_k, \tag{4}$$

where

$$\Psi_k = (F_k^H Q_k)^{-1} F_k^H R_k.$$

**Proof.** The proof is straightforward.  $\square$

Lemma 1 shows that one step of a block EN-like method can be considered as a step of the corresponding block Broyden method with  $\Phi_k = I_s$  followed by a step of the block version of Broyden's method using the optimal line search principle of [1] without updating the approximation to  $\mathcal{A}^{-1}$ . Note also the following relationships for the block EN-like method, that is of importance in Section 5:

$$\tilde{P}_k = P_k(I_s + \Psi_k) - \mathcal{H}_k Q_k \Psi_k \quad (5)$$

and

$$\tilde{Q}_k = Q_k + \mathcal{E}_k Q_k \Psi_k. \quad (6)$$

One question that immediately arises concerning the block EN-like method is how to choose  $F_k$ . There are two categories of choices for  $F_k$ , the first is inspired by the version of the block Broyden method that approximates  $\mathcal{A}^{-1}$  as in Algorithm 2, the second is inspired by the version of the block Broyden method that approximates the matrix  $\mathcal{A}$ . For each member of the first category, there is a dual member in the second category. We will focus here only on a few choices.

The block EN-like method with  $F_k = Q_k$  is called block BEN method since the corresponding Broyden method for  $s = 1$  is often called Broyden's 'bad' method. The block GEN method, where  $F_k = \mathcal{H}_k^H P_k$ , is its dual and the corresponding Broyden method for  $s = 1$  is called Broyden's 'good' method. The choice  $F_k = (I - \sum_{i=1}^{ks} \hat{q}_i \hat{q}_i^H) Q_k$ , where  $\hat{q}_i, i = 1, \dots, ks$ , are an orthonormal basis of the space spanned by the columns of  $Q_i, i = 0, \dots, k-1$ , yields the block PEN method, named so after Broyden's method with projected updates. Finally, the choice  $F_k = \mathcal{E}_k^H \mathcal{E}_k Q_k$  yields the block version of the original EN method. We will consider the properties of this method in more detail in the following section.

### 3. The block EN method and some of its properties

The EN method was first proposed by Eirola and Nevanlinna [2]. Their particular choice of  $\tilde{U}_k$  and  $V_k$  can be motivated as follows: The rank- $s$  update, denoted  $\tilde{U}_k V_k^H$ , is chosen so that the  $k$ th approximation of the inverse,  $\mathcal{H}_k$ , is no worse than the  $(k-1)$ st approximation  $\mathcal{H}_{k-1}$ . If we quantify the quality of the approximation by defining the error matrix

$$\mathcal{E}_k := \mathcal{I} - \mathcal{A} \mathcal{H}_k,$$

the constraints on the rank- $s$  update can be achieved by first choosing

$$V_k = \mathcal{E}_k^H \mathcal{A} \tilde{U}_k (\tilde{U}_k^H \mathcal{A}^H \mathcal{A} \tilde{U}_k)^{-1},$$

and then considering  $\tilde{U}_k$ . If we define the residual  $R_k := B - \mathcal{A} X_k$ , the best choice is

$$\tilde{U}_k = \mathcal{A}^{-1} \mathcal{E}_k R_k,$$

which yields to  $R_{k+1} = 0$ . Of course, such a choice clearly begs the question of solving the system of linear equations. Therefore, we use  $\mathcal{H}_k$ , the best available approximation of  $\mathcal{A}^{-1}$ , and set

$$\tilde{U}_k = \mathcal{H}_k \mathcal{E}_k R_k.$$

This method has several interesting properties, some of which we consider later in the context of the more general class of EN-like methods. Theorem 1 summarizes a few of the basic characteristics of the block EN method.

**Theorem 1.** Assume that  $\tilde{\mathcal{A}}\tilde{U}_k$  is of full rank. Decompose  $\tilde{\mathcal{A}}\tilde{U}_k$  into the orthogonal  $n \times s$ -matrix  $C_k$  and the upper triangular  $s \times s$ -matrix  $\Sigma$ . Define the following matrix:

$$U_k := \tilde{U}_k \Sigma_k^{-1}.$$

Then, the block EN method has the following properties:

- (i)  $\mathcal{E}_{k+1} = (\mathcal{I} - C_k C_k^H) \mathcal{E}_k$ ,
- (ii)  $C_i^H C_j = \Delta_{ij}$  (orthogonality),
- (iii)  $U_i^H \mathcal{A}^H \mathcal{A} U_j = \Delta_{ij}$  ( $\mathcal{A}^H \mathcal{A}$ -conjugacy),
- (iv)  $C_i^H R_{j+1} = 0, j \geq i$ ,
- (v) the individual singular values of  $\mathcal{E}_k$  do not increase with increasing  $k$ s,

where

$$\Delta_{ij} = \begin{cases} I_s & \text{for } i = j, \\ 0 & \text{for } i \neq j. \end{cases}$$

**Proof.** (i):

$$\begin{aligned} \mathcal{E}_{k+1} &= \mathcal{I} - \mathcal{A}\mathcal{H}_{k+1} = \mathcal{I} - \mathcal{A}\mathcal{H}_k - \tilde{\mathcal{A}}\tilde{U}_k(\tilde{U}_k^H \mathcal{A}^H \tilde{\mathcal{A}}\tilde{U}_k)^{-1} \tilde{U}_k^H \mathcal{A}^H \mathcal{E}_k \\ &= \mathcal{E}_k - C_k \Sigma_k (\Sigma_k^H C_k^H C_k \Sigma_k)^{-1} \Sigma_k^H C_k^H \mathcal{E}_k = (\mathcal{I} - C_k C_k^H) \mathcal{E}_k. \end{aligned}$$

(ii) For  $i = j$ ,  $C_i^H C_i = I_s$  according to its definition. If we define  $Q_i := \mathcal{A}\mathcal{H}_i R_i$ , then we have

$$C_i = \tilde{\mathcal{A}}\tilde{U}_i \Sigma_i^{-1} = \mathcal{E}_i Q_i \Sigma_i^{-1}.$$

It is easy to see that

$$\mathcal{E}_{k+1} Q_k = (\mathcal{I} - C_k C_k^H) \mathcal{E}_k Q_k = (\mathcal{I} - C_k C_k^H) C_k \Sigma_k = 0,$$

and using induction and (i), we get for  $j > 0$

$$\mathcal{E}_{k+j} Q_k = 0.$$

We prove  $C_i^H C_j = 0$  for  $j > i$ , using induction. Certainly,

$$\begin{aligned} C_0^H C_1 &= \Sigma_0^{-H} Q_0^H \mathcal{E}_0^H \mathcal{E}_1 Q_1 \Sigma_1^{-1} = \Sigma_0^{-H} Q_0^H \mathcal{E}_0^H (I - C_0 C_0^H) \mathcal{E}_0 Q_1 \Sigma_1^{-1} \\ &= \Sigma_0^{-H} Q_0^H \mathcal{E}_1^H \mathcal{E}_0 Q_1 \Sigma_1^{-1} = 0. \end{aligned}$$

We assume for  $j > i$

$$C_i^H C_j = 0.$$

Since the multiplication of  $I - VV^H$  and  $I - WW^H$  is commutative, if  $V^H W = 0$ , we have

$$\begin{aligned} C_i^H C_{j+1} &= \Sigma_i^{-H} Q_i^H \mathcal{E}_i^H \mathcal{E}_{j+1}^H Q_{j+1} \Sigma_{j+1}^{-1} = \Sigma_i^{-H} Q_i^H \mathcal{E}_i^H (\mathcal{I} - C_j C_j^H) \cdots (\mathcal{I} - C_i C_i^H) \mathcal{E}_i Q_{j+1} \Sigma_{j+1}^{-1} \\ &= \Sigma_i^{-H} Q_i^H \mathcal{E}_{j+1}^H \mathcal{E}_i Q_{j+1} \Sigma_{j+1}^{-1} = 0. \end{aligned}$$

(iii) follows from (ii) since  $C_i = \mathcal{A}U_i$ .

(iv) Using (i), we have

$$C_i^H R_{i+1} = C_i^H \mathcal{E}_{i+1} R_i = C_i^H (I - C_i C_i^H) \mathcal{E}_i R_i = 0.$$

(v) can be proved using (i) and [9, Theorem 5.9]. Since the singular values of  $\mathcal{E}_k$  are the square roots of the eigenvalues of the Hermitian matrix  $\mathcal{E}_k^H \mathcal{E}_k$ , the following relationship is valid:

$$\begin{aligned} \sigma_i(\mathcal{E}_{k+1}) &= \min_{\dim(S)=i} \max_{x \in S \setminus \{0\}} \frac{\|\mathcal{E}_{k+1}x\|^2}{\|x\|^2} = \min_{\dim(S)=i} \max_{x \in S \setminus \{0\}} \frac{\|(\mathcal{I} - C_k C_k^H) \mathcal{E}_k x\|^2}{\|x\|^2} \\ &\leq \min_{\dim(S)=i} \max_{x \in S \setminus \{0\}} \frac{\|\mathcal{E}_k x\|^2}{\|x\|^2} = \sigma_i(\mathcal{E}_k). \quad \square \end{aligned}$$

#### 4. Reduced complexity versions

One of the disadvantages of the methods considered so far is their computational complexity. While  $\mathcal{A}$  is sparse,  $\mathcal{H}_k$  is, in general, dense and therefore its multiplication with an  $n \times s$ -matrix requires  $O(sn^2)$  operations compared to  $O(sn)$  for the multiplication of  $\mathcal{A}$  with the matrix. More efficient versions of the methods avoid the explicit computation of  $\mathcal{H}_k$  and use only  $\mathcal{H}_0$ , the original approximation of  $\mathcal{A}^{-1}$ , that is, in general, sparse or at least its application to a vector can be done in a manner of low complexity. If we replace  $\mathcal{H}_{k+1}$  with its definition, we get

$$\mathcal{H}_{k+1}X = \mathcal{H}_kX + (P_k - \mathcal{H}_k Q_k)(F_k^H Q_k)^{-1}(F_k^H X) = \mathcal{H}_0X + \sum_{i=0}^k (P_i - \mathcal{H}_i Q_i)(F_i^H Q_i)^{-1}(F_i^H X).$$

Using the definition

$$Z_k = P_k - \mathcal{H}_k Q_k, \tag{7}$$

the general block EN-like method can be written as follows.

**Algorithm 3.** Block EN-like method (efficient version)

**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ .

**For**  $k = 0, 1, \dots$ :

$$\begin{aligned} P_k &= \mathcal{H}_0 R_k + \sum_{i=0}^{k-1} Z_i F_i^H R_k, \\ Q_k &= \mathcal{A}P_k, \\ \Sigma_k &= F_k^H Q_k, \\ T_k &= R_k - Q_k, \\ Z_k &= \left( \mathcal{H}_0 T_k + \sum_{i=0}^{k-1} Z_i F_i^H T_k \right) \Sigma_k^{-1}, \\ S_k &= P_k + Z_k F_k^H R_k, \\ X_{k+1} &= X_k + S_k, \\ R_{k+1} &= R_k - \mathcal{A}S_k \end{aligned}$$

**end.**

Setting  $F_i = Q_i$  yields block BEN. Block GEN, however, requires further transformation, since the evaluation of  $F_k$  involves  $\mathcal{H}_k^H$ . The new version of block GEN contains the matrix  $F_i^H Q_k = P_i^H \mathcal{H}_i Q_k$ .

Therefore, one needs to determine

$$T_k^{(i)} = \mathcal{H}_i Q_k. \quad (8)$$

This can be evaluated without increasing the number of operations or the storage needed, but the evaluation is highly recursive and leads to a decrease in parallelism. The efficient version for block GEN is:

**Algorithm 4.** Block GEN method ( $F_k = \mathcal{H}_k^H P_k$ )

**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ ,

**For**  $k = 0, 1, \dots$ :

$$\begin{aligned} P_k^{(0)} &= \mathcal{H}_0 R_k, \\ P_k^{(i)} &= P_k^{(i-1)} + Z_{i-1} (P_{i-1}^{(i-1)})^H P_k^{(i-1)}, \quad i = 1, \dots, k, \\ Q_k &= \mathcal{A} P_k^{(k)}, \\ T_k^{(0)} &= \mathcal{H}_0 Q_k, \\ T_k^{(i)} &= T_k^{(i-1)} + Z_{i-1} (P_{i-1}^{(i-1)})^H T_k^{(i-1)}, \quad i = 1, \dots, k, \\ \Sigma_k &= (P_k^{(k)})^H T_k^{(k)}, \\ Z_k &= (P_k^{(k)} - T_k^{(k)}) \Sigma_k^{-1}, \\ S_k &= P_k^{(k)} + Z_k (P_k^{(k)})^H P_k^{(k)}, \\ X_{k+1} &= X_k + S_k, \\ R_{k+1} &= R_k - \mathcal{A} S_k \end{aligned}$$

**end.**

For both block BEN and block GEN,  $R_{k+1}$  can also be evaluated through

$$R_{k+1} = B - \mathcal{A} X_{k+1}$$

without increasing the number of operations. In some cases, this approach appears to be more stable (see Section 6).

Setting  $F_k = \mathcal{E}_k^H \mathcal{E}_k Q_k$  yields the block EN method. We use

$$\tilde{C}_k = \mathcal{E}_k Q_k, \quad \tilde{U}_k = \mathcal{H}_k \mathcal{E}_k R_k$$

and the orthogonality of the columns of the  $C_i$ , see Theorem 1, to develop the efficient version below.  $[Q, \Sigma] = \text{GS}(C)$  denotes the application of the (modified or classical) Gram–Schmidt algorithm to the  $n \times s$ -matrix  $C$ , which generates the orthogonal  $n \times s$ -matrix  $Q$  and the upper triangular  $s \times s$ -matrix  $\Sigma$ .

**Algorithm 5.** Block EN method

**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ ,

**For**  $k = 0, 1, \dots$ :

$$\begin{aligned} \Psi_i &= C_i^H \mathcal{A} \mathcal{H}_0 R_k, \quad i = 0, \dots, k-1, \\ P_k &= \mathcal{H}_0 R_k - \sum_{i=0}^{k-1} U_i \Psi_i, \end{aligned}$$

$$\begin{aligned}
T_k &= R_k - \mathcal{A}\mathcal{H}_0 R_k + \sum_{i=0}^{k-1} C_i \Psi_i, \\
\Phi_i &= C_i^H \mathcal{A}\mathcal{H}_0 T_k, \quad i = 0, \dots, k-1, \\
\tilde{U}_k &= \mathcal{H}_0 T_k - \sum_{i=0}^{k-1} U_i \Phi_i, \\
\tilde{C}_k &= \mathcal{A}\mathcal{H}_0 T_k - \sum_{i=0}^{k-1} C_i \Phi_i, \\
[C_k, \Sigma_k] &= \text{GS}(\tilde{C}_k), \\
U_k &= \tilde{U}_k \Sigma_k^{-1}, \\
\Delta_k &= C_k^H T_k, \\
X_{k+1} &= X_k + P_k + U_k \Delta_k, \\
R_{k+1} &= T_k - C_k \Delta_k
\end{aligned}$$

**end.**

An alternative approach is possible when evaluating  $T_k$  and  $\tilde{C}_k$  through

$$T_k = R_k - \mathcal{A}P_k, \quad \tilde{C}_k = \mathcal{A}\tilde{U}_k.$$

If  $\mathcal{A}$  is very sparse and  $k$  large, this approach saves a significant number of operations. However, to find the most efficient implementation, one needs to consider also how fast the multiplication of two dense matrices can be performed in comparison to the multiplication of a sparse matrix with a dense matrix on the target computer.

As in the case of a single right-hand side, it is possible to derive a scaling-invariant block EN method [12,3]. This method, called block SEN, can be obtained from the previous algorithm by introducing an  $s \times s$ -matrix  $\Gamma_k$ :

$$\Gamma_k = [(\mathcal{A}\mathcal{H}_0 R_k)^H \mathcal{A}\mathcal{H}_0 R_k]^{-1} (\mathcal{A}\mathcal{H}_0 R_k)^H R_k \quad (9)$$

and replacing the evaluations of  $\Psi_i$ ,  $P_k$  and  $T_k$  in Algorithm 5 with

$$\begin{aligned}
\Psi_i &= C_i^H \mathcal{A}\mathcal{H}_0 R_k \Gamma_k, \quad i = 0, \dots, k-1, \\
P_k &= \mathcal{H}_0 R_k \Gamma_k - \sum_{i=0}^{k-1} U_i \Psi_i, \\
T_k &= R_k - \mathcal{A}\mathcal{H}_0 R_k \Gamma_k + \sum_{i=0}^{k-1} C_i \Psi_i.
\end{aligned}$$

It is easy to show that for the block SEN method  $\|R_{k+1}\|$  cannot be larger than  $\|R_k\|$ .

We also include the block GCR method which is interesting in this context because of its relationship to block EN and its equivalence to block GMRES.



**Algorithm 6.** Block GCR**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ .**For**  $k = 0, 1, \dots$ :

$$\Phi_i = C_i^H \mathcal{A} \mathcal{H}_0 R_k, \quad i = 0, \dots, k-1,$$

$$\tilde{U}_k = \mathcal{H}_0 R_k - \sum_{i=0}^{k-1} U_i \Phi_i,$$

$$\tilde{C}_k = \mathcal{A} \mathcal{H}_0 R_k - \sum_{i=0}^{k-1} C_i \Phi_i,$$

$$[C_k, \Sigma_k] = \text{GS}(\tilde{C}_k),$$

$$U_k = \tilde{U}_k \Sigma_k^{-1},$$

$$\Delta_k = C_k^H R_k,$$

$$X_{k+1} = X_k + U_k \Delta_k,$$

$$R_{k+1} = R_k - C_k \Delta_k$$

**end.**

As for block EN,  $\tilde{C}_k$  can be evaluated here as  $\tilde{C}_k = \mathcal{A} \tilde{U}_k$ .

If we do not require the evaluation of  $X_{k+1}$  at each iteration step, it is possible to achieve further savings in block GCR and block EN by avoiding the evaluation of  $U_k$ . Only the coefficients need to be saved. Such an approach has been used in [10] for single right-hand side GCR. The block version of this implementation of GCR is

**Algorithm 7.** Block eGCR**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ ,  $\tilde{X}_0 = X_0$ .**For**  $k = 0, 1, \dots, m$ :

$$\hat{U}_k = \mathcal{H}_0 R_k,$$

$$\Phi_{i,k} = C_i^H \mathcal{A} \hat{U}_k, \quad i = 0, \dots, k-1,$$

$$\tilde{C}_k = \mathcal{A} \hat{U}_k - \sum_{i=0}^{k-1} C_i \Phi_{i,k},$$

$$[C_k, \Sigma_k] = \text{GS}(\tilde{C}_k),$$

$$\Delta_k = C_k^H R_k,$$

$$R_{k+1} = R_k - C_k \Delta_k$$

**end.**

$X_{m+1} = X_0 + \overline{U}_m F_m^{-1} D_m$ , where

$$D_m = (\Delta_0, \dots, \Delta_m)^H, \quad \overline{U}_m = (\hat{U}_0, \dots, \hat{U}_m), \quad F_m = \begin{pmatrix} \Sigma_0 & \Phi_{0,1} & \cdots & \Phi_{0,m} \\ & \Sigma_1 & \ddots & \vdots \\ & & \ddots & \Phi_{m-1,m} \\ & & & \Sigma_m \end{pmatrix}.$$

Using the same approach, one can generate a more efficient version for block EN.

**Algorithm 8.** Block eEN method

**Initialization:**  $X_0, \mathcal{H}_0$  arbitrary,  $R_0 = B - \mathcal{A}X_0$ ,  $\tilde{X}_0 = X_0$ .

**For**  $k = 0, 1, \dots, m$ :

$$\Psi_{i,k} = C_i^H \mathcal{A} \mathcal{H}_0 R_k, \quad i = 0, \dots, k-1,$$

$$T_k = R_k - \mathcal{A} \mathcal{H}_0 R_k + \sum_{i=0}^{k-1} C_i \Psi_{i,k},$$

$$\hat{U}_k = \mathcal{H}_0 T_k,$$

$$\Phi_{i,k} = C_i^H \mathcal{A} \mathcal{H}_0 T_k, \quad i = 0, \dots, k-1,$$

$$\tilde{C}_k = \mathcal{A} \mathcal{H}_0 T_k - \sum_{i=0}^{k-1} C_i \Phi_{i,k},$$

$$[C_k, \Sigma_k] = \text{GS}(\tilde{C}_k),$$

$$\Delta_k = C_k^H T_k,$$

$$\tilde{X}_{k+1} = \tilde{X}_k + \mathcal{H}_0 R_k,$$

$$R_{k+1} = T_k - C_k \Delta_k$$

**end.**

$X_{m+1} = \tilde{X}_{m+1} + \bar{U}_m F_m^{-1} (D_m - G_m E)$ , where  $F_m$  and  $D_m$  are defined as above, and

$$\bar{U}_m = (\hat{U}_0, \dots, \hat{U}_m), \quad E = (I_s, I_s, \dots, I_s)^H, \quad G_m = \begin{pmatrix} 0 & \Psi_{0,1} & \dots & \Psi_{0,m} \\ & 0 & \ddots & \vdots \\ & & \ddots & \Psi_{m-1,m} \\ & & & 0 \end{pmatrix}.$$

Even though these new versions avoid the multiplication with a dense  $H_k$ , they are still computationally expensive due to an increase in both operation count and memory requirement with each iteration. It is possible to save memory and operations per step by developing restarted or truncated versions.

Tables 1–4 show the type and number of operations, and the amount of memory required for the methods. Only operations and storage of order  $n$  is considered, since we assume that  $m$  and  $s$  are small compared to  $n$ . The methods are denoted by the abbreviations given earlier with an initial “B” appended indicating a block version as opposed to the single right-hand sided version. The truncated version of any method is indicated by the inclusion of a “t”. However, block Orthomin is used for the truncated version of block GCR since it is known by that name in the literature. The notes “opt.” and “alt. appr.” indicate the use of the optimal line search principal and the alternate evaluations of  $T_k$  and  $\tilde{C}_k$  discussed earlier. The primitive labeled “dmxm1” refers to dense matrix multiplications of an  $n \times ks$  with a  $ks \times s$  matrix or a  $ks \times n$  with an  $n \times s$  matrix and “dmxm2” refers to dense matrix multiplications of an  $n \times s$  with an  $s \times s$  matrix or an  $s \times n$  with an  $n \times s$  matrix. The column labeled +, −, \* counts the number of vector operations involving  $s$  vectors of length  $n$ . The total operation count for each of these primitives is included at the top of each column. The primitives “spmv” and “precond” refer to the product of a sparse matrix and a vector of length  $n$ , and a preconditioner operation involving a sparse matrix operating implicitly or explicitly on a vector of length  $n$ . Their operation counts depend on the matrix and are

Table 1

Number and types of operations for various block methods in  $k$ th iteration step

| Method               | Primitive/operations |          |           |      |         |
|----------------------|----------------------|----------|-----------|------|---------|
|                      | dmxm1                | dmxm2    | $+, -, *$ | spmv | precond |
|                      | $2ks^2n$             | $2s^2n$  | $sn$      |      |         |
| BBBM, $\Phi_k = I_s$ | 2                    | 3        | 3         | $s$  | $s$     |
| BBBM, opt.           | 2                    | 5        | 1         | $s$  | $s$     |
| BGBM, $\Phi_k = I_s$ | —                    | $2k + 3$ | 3         | $s$  | $s$     |
| BGBM, opt.           | —                    | $2k + 5$ | 1         | $s$  | $s$     |
| BGCR                 | 3                    | 5        | —         | $s$  | $s$     |
| BGCR, alt. appr.     | 2                    | 5        | —         | $2s$ | $s$     |
| BeGCR                | 2                    | 4        | —         | $s$  | $s$     |
| BGMRES               | 2                    | 4        | —         | $s$  | $s$     |
| BBEN                 | 4                    | 3        | 3         | $2s$ | $2s$    |
| BGEN                 | —                    | $4k + 3$ | 3         | $2s$ | $2s$    |
| BEN                  | 6                    | 5        | 2         | $2s$ | $2s$    |
| BEN, alt. appr.      | 4                    | 5        | 2         | $4s$ | $2s$    |
| BeEN                 | 4                    | 4        | 2         | $2s$ | $2s$    |
| BSEN                 | 6                    | 8        | 1         | $2s$ | $2s$    |
| BSEN, alt. appr.     | 4                    | 8        | 1         | $4s$ | $2s$    |
| BeSEN                | 4                    | 7        | 1         | $2s$ | $2s$    |

therefore not listed. Note that it is possible to decrease the additional work vectors required for block (t)GBM( $m$ ) to  $(m + 6)s$  with an increase of its computational complexity, see [1].

## 5. Convergence theory

Both the block Broyden methods as well as the block EN-like methods possess a finite termination property. A proof for the block Broyden methods can be found in [6].

To simplify the presentation, we use the following definitions:

$$V_k := F_k(F_k^H Q_k)^{-H}, \quad (10)$$

$$\mathcal{P}_k^{(0)} := \mathcal{E}_k \mathcal{E}_{k-1} \cdots \mathcal{E}_0, \quad (11)$$

and

$$\mathcal{P}_k := \mathcal{E}_k \mathcal{E}_{k-1} \cdots \mathcal{E}_1. \quad (12)$$

Table 2  
Computational complexities for truncated block versions

| Method                            | Number of operations per iteration                          |
|-----------------------------------|---|
| BtBBM( $m$ ), $\Phi_k = I_s$      | $[(4m + 6)s^2 + 3s]n + s \text{ spmv} + s \text{ prec}$     |
| BtBBM( $m$ ), opt.                | $[(4m + 10)s^2 + s]n + s \text{ spmv} + s \text{ prec}$     |
| BtGBM( $m$ ), $\Phi_k = I_s$      | $[(4m + 6)s^2 + 3s]n + s \text{ spmv} + s \text{ prec}$     |
| BtGBM( $m$ ), opt.                | $[(4m + 10)s^2 + s]n + s \text{ spmv} + s \text{ prec}$     |
| block Orthomin( $m$ )             | $[(6m + 10)s^2]n + s \text{ spmv} + s \text{ prec}$         |
| block Orthomin( $m$ ), alt. appr. | $[(4m + 10)s^2]n + 2s \text{ spmv} + s \text{ prec}$        |
| BtBEN( $m$ )                      | $[(8m + 6)s^2 + 3s]n + 2s \text{ spmv} + 2s \text{ prec}$   |
| BtGEN( $m$ )                      | $[(8m + 6)s^2 + 3s]n + 2s \text{ spmv} + 2s \text{ prec}$   |
| BtEN( $m$ )                       | $[(12m + 10)s^2 + 2s]n + 2s \text{ spmv} + 2s \text{ prec}$ |
| BtEN( $m$ ), alt. appr.           | $[(8m + 10)s^2 + 2s]n + 4s \text{ spmv} + 2s \text{ prec}$  |
| BtSEN( $m$ )                      | $[(12m + 16)s^2 + s]n + 2s \text{ spmv} + 2s \text{ prec}$  |
| BtSEN( $m$ ), alt. appr.          | $[(8m + 16)s^2 + s]n + 4s \text{ spmv} + 2s \text{ prec}$   |

Table 3  
Computational complexities for restarted versions

| Method                      | Average number of operations per iteration                |
|-----------------------------|---|
| BBBM( $m$ ), $\Phi_k = I_s$ | $[(2m + 6)s^2 + 3s]n + s \text{ spmv} + s \text{ prec}$   |
| BBBM( $m$ ), opt.           | $[(2m + 10)s^2 + s]n + s \text{ spmv} + s \text{ prec}$   |
| BGBM( $m$ ), $\Phi_k = I_s$ | $[(2m + 6)s^2 + 3s]n + s \text{ spmv} + s \text{ prec}$   |
| BGBM( $m$ ), opt.           | $[(2m + 10)s^2 + s]n + s \text{ spmv} + s \text{ prec}$   |
| BeGCR( $m$ )                | $[(2m + 8)s^2]n + s \text{ spmv} + s \text{ prec}$        |
| BGMRES( $m + 1$ )           | $[(2m + 8)s^2]n + s \text{ spmv} + s \text{ prec}$        |
| BBEN( $m$ )                 | $[(4m + 6)s^2 + 3s]n + 2s \text{ spmv} + 2s \text{ prec}$ |
| BGEN( $m$ )                 | $[(4m + 6)s^2 + 3s]n + 2s \text{ spmv} + 2s \text{ prec}$ |
| BeEN( $m$ )                 | $[(4m + 8)s^2 + 2s]n + 2s \text{ spmv} + 2s \text{ prec}$ |
| BeSEN( $m$ )                | $[(4m + 14)s^2 + s]n + 2s \text{ spmv} + 2s \text{ prec}$ |

**Lemma 2.** Assume  $V_j^H Q_{j-1}$ ,  $j = 1, \dots, k$ , has no zero columns, and  $Q_k$  has full rank. Let  $V_0$  be of full rank and in the range of  $\mathcal{E}_0^H$ , and  $k \leq 2n/s$  be odd.

Define the sequence of  $n \times s$ -matrices

$$Z_1^H \mathcal{E}_1 = 0, \quad Z_i^H \mathcal{E}_i = Z_{i-2}^H, \quad i = 3, 5, \dots, k.$$

Table 4  
Additional work vectors of length  $n$  required

| Method                            | Memory required |
|-----------------------------------|-----------------|
| B(t)BBM( $m$ )                    | $(2m + 5)s$     |
| B(t)GBM( $m$ )                    | $(2m + 5)s$     |
| B(t)GCR( $m$ ), block eGCR( $m$ ) | $(2m + 4)s$     |
| BGMRES( $m + 1$ )                 | $(m + 5)s$      |
| B(t)BEN( $m$ )                    | $(2m + 6)s$     |
| B(t)GEN( $m$ )                    | $(2m + 6)s$     |
| B(t)EN( $m$ ), block eEN( $m$ )   | $(2m + 4)s$     |
| B(t)SEN( $m$ ), block eSEN( $m$ ) | $(2m + 4)s$     |

Then,  $Z_i$ ,  $i = 1, 3, \dots, k$ , exist and their columns are linearly independent. Additionally,

$$Z_i^H \mathcal{P}_j^{(0)} = 0, \quad j = i, \dots, k,$$

$$Z_i^H R_j = 0, \quad j = i + 1, \dots, k.$$

**Proof.** See [6].  $\square$

Using the same approach, one can prove for the block EN-like methods:

**Lemma 3.** Assume  $V_j^H Q_{j-1}$ ,  $j = 1, \dots, k$ , has no zero columns, and  $Q_k$  has full rank. Let  $V_0$  be of full rank and in the range of  $\mathcal{E}_0^H$ , and  $k \leq n/s$ .

Define the sequence of  $n \times s$ -matrices

$$Z_1^H \mathcal{E}_1 = 0, \quad Z_i^H \mathcal{E}_i = Z_{i-1}^H, \quad i = 2, 3, \dots, k.$$

Then,  $Z_i$ ,  $i = 1, 2, \dots, k$ , exist and their columns are linearly independent. Additionally,

$$Z_i^H \mathcal{P}_j = 0, \quad j = i, \dots, k + 1,$$

$$Z_i^H R_j = 0, \quad j = i + 1, \dots, k + 1.$$

**Proof.** The lemma is proved by induction. Let us prove the above for  $k = 1$ .

Define  $Z_1$  through  $Z_1^H \mathcal{E}_0 = V_0^H$ . This is possible, since we assumed that  $V_0$  is in the range of  $\mathcal{E}_0$ .  $Z_1$  has linearly independent columns, since  $V_0$  is of full rank. Then

$$Z_1^H \mathcal{E}_j = Z_1^H \mathcal{E}_0 (\mathcal{I} - Q_0 V_0^H) \cdots (\mathcal{I} - Q_{j-1} V_{j-1}^H) = 0, \quad j = 1, \dots, k + 1.$$

Consequently,

$$\begin{aligned} Z_1^H \mathcal{P}_j &= Z_1^H \mathcal{E}_j \mathcal{E}_{j-1} \cdots \mathcal{E}_1 = 0, & j &= 1, \dots, k+1, \\ Z_1^H R_j &= Z_1^H \mathcal{P}_j R_0 = 0, & j &= 1, \dots, k+1, \\ Z_1^H \tilde{Q}_j &= Z_1^H R_j - Z_1^H R_{j+1} = 0, & j &= 1, \dots, k, \\ Z_1^H Q_j &= Z_1^H \tilde{Q}_j - Z_1^H \mathcal{E}_j Q_j V_j^H R_j = 0, & j &= 1, \dots, k. \end{aligned}$$

Now, assume that  $Z_1, \dots, Z_{i-1}$  exist and have linearly independent columns for  $i \leq k$ , and

$$\begin{aligned} Z_{i-1}^H R_j &= Z_{i-1}^H \mathcal{P}_j = 0, & j &= i-1, \dots, k+1, \\ Z_{i-1}^H \tilde{Q}_j &= Z_{i-1}^H Q_j = 0, & j &= i-1, \dots, k. \end{aligned}$$

We define  $Z_i^H \mathcal{E}_i = Z_{i-1}^H$ . Since  $Z_{i-1}^H Q_{i-1}$  has no zero columns and the columns of  $Q_{i-1}$  are the only null vectors of  $\mathcal{E}_i$  due to the assumption that  $V_i^H Q_{i-1}$  has no zero columns,  $Z_i$  exists.

Also, we have the following equation:

$$Z_i^H \mathcal{E}_j = Z_i^H \mathcal{E}_i (\mathcal{I} - Q_i V_i^H) \cdots (\mathcal{I} - Q_{j-1} V_{j-1}^H) = Z_{i-1}^H (\mathcal{I} - Q_i V_i^H) \cdots (\mathcal{I} - Q_{j-1} V_{j-1}^H) = Z_{i-1}^H,$$

since  $Z_{i-1}^H Q_l = 0$ ,  $l = i, \dots, j-1$ , due to the assumption.

Assume the matrices  $Z_1, \dots, Z_i$  have linearly dependent columns,  $z_1, z_2, \dots, z_{is}$ , i.e., there exist  $\gamma_j$ ,  $j = 1, \dots, is$ , with  $\gamma_j \neq 0$  for at least one  $j$  and

$$\sum_{j=1}^{is} \gamma_j z_j = 0. \quad (13)$$

Consequently,

$$0 = \sum_{j=1}^{is} \gamma_j z_j^H \mathcal{E}_i = \sum_{j=s+1}^{is} \gamma_j z_{j-s}^H \neq 0,$$

since  $z_1, z_2, \dots, z_{(i-1)s}$  are linearly independent. This is a contradiction, and the vectors  $z_1, \dots, z_{is}$  must therefore be linearly independent.

We have, for  $j = i, \dots, k+1$ ,

$$Z_i^H \mathcal{P}_j = Z_i^H \mathcal{E}_j \mathcal{E}_{j-1} \cdots \mathcal{E}_1 = Z_{i-1}^H \mathcal{E}_{j-1} \cdots \mathcal{E}_1 = Z_{i-1}^H \mathcal{P}_{j-1} = 0.$$

This leads to

$$\begin{aligned} Z_i^H R_j &= Z_i^H \mathcal{P}_j R_0 = 0, & j &= i, \dots, k+1, \\ Z_i^H \tilde{Q}_j &= Z_i^H R_j - Z_i^H R_{j+1} = 0, & j &= i, \dots, k, \\ Z_i^H Q_j &= Z_i^H \tilde{Q}_j - Z_i^H \mathcal{E}_j Q_j V_j^H R_j = -Z_{i-1}^H Q_j V_j^H R_j = 0, & j &= i, \dots, k+1. \quad \square \end{aligned}$$

We can also show that the rank of  $\mathcal{E}_k$  cannot increase, as  $k$  increases.

**Lemma 4.** Assume,  $V_k$  is of full rank.

$$\text{rank}(\mathcal{E}_k) - s \leq \text{rank}(\mathcal{E}_{k+1}) \leq \text{rank}(\mathcal{E}_k), \quad (14)$$

where

$$\text{rank}(\mathcal{E}_{k+1}) = \text{rank}(\mathcal{E}_k) - s \quad (15)$$

if and only if there exists a  $n \times s$ -matrix  $Y$  of full rank with  $Y^H \mathcal{E}_k = V_k^H$ .

**Proof.**

$$\text{rank}(\mathcal{E}_k) = \text{rank}(\mathcal{E}_k^H) = m. \quad (16)$$

Consequently, there exist  $z_i, i = 1, \dots, n - m$ , with  $z_i^H \mathcal{E}_k = 0$ . Now,

$$z_i^H \mathcal{E}_{k+1} = z_i^H \mathcal{E}_k (\mathcal{I} - Q_k V_k^H) = 0, \quad i = 1, \dots, n - m. \quad (17)$$

In order for  $\text{rank}(\mathcal{E}_{k+1})$  to be smaller than  $\text{rank}(\mathcal{E}_k)$ , we need  $y_l$  with  $y_l^H \mathcal{E}_k = (v_l^{(k)})^H$ , where  $v_l^{(k)}$  is the  $l$ th column of  $V_k$ .  $\square$

**Theorem 2.** For nonsingular  $F_k^H Q_k$ , the block EN-like method converges within at most  $n/s$  steps.

**Proof.** Using the results of Lemma 4, we can redefine a sequence of  $Z_i$  without the restriction of Lemma 3 that  $V_j^H Q_{j-1}$  has no zero columns:

For  $V_i^H Q_{i-1}$  without zero columns, we choose  $Z_i^H \mathcal{E}_i = Z_{i-1}$  as in the proof for Lemma 3. If the  $l$ th column  $\hat{v}_l$  of  $V_i^H Q_{i-1}$  is a zero column, and there exists a vector  $y_l$  with  $y_l^H \mathcal{E}_i = \hat{v}_l^H$ , we set  $z_{(i-1)s+l} = y_l$ . If no such  $y_l$  exists, we choose a  $z_{(i-1)s+l}$  with  $z_{(i-1)s+l}^H \mathcal{E}_i = 0$  and  $z_{(i-1)s+l}$  linearly independent of  $z_1, \dots, z_{(i-1)s+l-1}$ . Such a vector exists, since, if  $V_i^H Q_{i-1}$  has zero columns,  $\text{rank}(\mathcal{E}_i)$  is smaller than  $\text{rank}(\mathcal{E}_{i-1})$ .

Now, we can show, as in Lemma 3, that

$$Z_i^H R_j = 0, \quad j = i, \dots, k + 1,$$

for  $k \leq n/s$ , from which we conclude finite termination within at most  $n/s$  steps.  $\square$

Successive residual matrices of the block Broyden methods and the block EN-like methods have simple relationships via the error matrix  $\mathcal{E}_k$ . These are summarized in Lemmas 5 and 6, respectively.

**Lemma 5.** For the block Broyden method,

$$R_{k+1} = R_k (I_s - \Phi_k) + \mathcal{E}_k R_k \Phi_k, \quad (18)$$

which for  $\Phi_k = I_s$  reduces to

$$R_{k+1} = \mathcal{E}_k R_k. \quad (19)$$

**Proof.** The proof is according to [1].  $\square$

**Lemma 6.** For the EN-like method,

$$R_{k+1} = \mathcal{E}_{k+1} \mathcal{E}_k R_k. \quad (20)$$

**Proof.** Using  $\mathcal{E}_{k+1} Q_k = 0$ ,

$$R_{k+1} = \mathcal{E}_{k+1} R_k = \mathcal{E}_{k+1} (R_k - Q_k) = \mathcal{E}_{k+1} \mathcal{E}_k R_k. \quad \square$$

Using this lemma, the following convergence theorem follows.

**Theorem 3.** Assume that  $\|\mathcal{E}_0\| \leq \delta < 1$ , and  $\|\mathcal{E}_{k+1}\| \leq \|\mathcal{E}_k\|$  for  $k \geq 0$  for the block EN-like methods. The following inequality then holds:

$$\|R_{k+1}\| \leq \delta^2 \|R_k\| \leq \delta^{2k+2} \|R_0\|, \quad (21)$$

and the methods converge.

The first condition requires a good estimate  $\mathcal{H}_0$  of  $A^{-1}$ . In practice this could result from a good preconditioner. The second condition is always satisfied for the block EN method ( $F_k = \mathcal{E}_k^H \mathcal{E}_k Q_k$ ), the block BEN method ( $F_k = Q_k$ ) and the block PEN method ( $F_k = Q_k - \sum_{i=0}^{k-1} \hat{Q}_i \hat{Q}_i^H Q_k$ , where the columns of  $\hat{Q}_i$ ,  $i = 0, \dots, k-1$ , are an orthonormal basis of the space spanned by the columns of  $Q_i$ ,  $i = 0, \dots, k-1$ ).

## 6. Numerical results

We implemented the new block methods and others on an SGI Origin 200 with 4 processors using a consistent set of dense and sparse computational primitives. The methods were tested on several matrices from the Harwell–Boeing collection with multiple random right-hand side vectors. The results for two of those matrices are discussed below.

Table 5 shows the results we obtained using the nonsymmetric matrix SHERMAN5. We ran the following block schemes choosing a Krylov subspace of  $k = 5$ : block GMRES (BG( $k$ )), the hybrid block GMRES (BG\_H( $k, m$ ) where  $m$  denotes the degree of the polynomial chosen), the block EN-like methods BeEN, BBEN, BGEN and the truncated scheme BtEN and the block Broyden methods BGBM, BBBM with optimal line search principle and BGBM1 and BBBM1 where  $\Phi_k = I_s$ .

The labels ‘mgs’ and ‘cgs’ denotes whether modified or classical Gram–Schmidt was used for orthogonalization. While the classical Gram–Schmidt performs significantly better for larger  $s$ , modified Gram–Schmidt is more stable and improves convergence for the block GMRES schemes in some cases. We only report the numerical results for ‘mgs’ for those methods for which there was a significant change in the number of sparse matrix vector multiplications. The use of MGS did not affect the convergence of BeEN and BtEN.

For more than one right-hand side most of the block Broyden methods fail, only the ‘good’ block Broyden method with optimal line search succeeds and is faster than block GMRES for less than 20 right-hand sides. As expected, the hybrid schemes are faster than block GMRES. Overall, the block EN-like methods are the fastest for this problem, especially BeEN(4) which for  $s = 20$  is almost twice as fast as the hybrid scheme, almost five times as fast as block GMRES and almost six times as fast as block GBM. For BBEN(4), the use of  $R_k = B - AX_k$  was necessary for  $s = 12$ , since the other version diverges for this case.

Since block GMRES requires less memory than the block Broyden and block EN-like methods, we also ran block GMRES and the hybrid schemes with a Krylov subspace of dimension 8. In this case block GMRES uses the same amount of memory as BeEN. As expected, the performance of block GMRES is significantly increased, however BeEN is still faster by a factor of 1.5 to 2.

If the times achieved for one right-hand side are compared to those for multiple right-hand sides, it is seen that the use of the block methods is, in general, significantly better than using the single right-hand side methods  $s$  times. Mostly, this is due to two effects, an increase in data locality as well as a decrease



Table 5

Times in seconds (number of spmv's) for SHERMAN5 with ILU(0)

| Method $s$     | 1        | 4        | 8          | 12         | 16         | 20         |
|----------------|----------|----------|------------|------------|------------|------------|
| BG(5)(cgs)     | 1.2(205) | 4.4(816) | 8.1(1328)  | 10.1(1380) | 12.5(1536) | 9.7(1100)  |
| BG(5)(mgs)     | 1.1(205) | 5.2(864) | 11.0(1304) | 18.7(1524) | 22.4(1456) | 16.9(1020) |
| BG_H(5,1)(cgs) | 0.9(182) | 2.0(428) | 1.5(280)   | 1.7(300)   | 2.8(464)   | 3.8(660)   |
| BG_H(5,1)(mgs) | 0.9(182) | 2.2(428) | 1.8(280)   | 2.2(300)   | 3.7(448)   | 4.6(620)   |
| BG_H(5,2)(cgs) | 1.2(255) | 0.8(196) | 1.0(232)   | 3.2(732)   | 6.5(1328)  | 9.6(2000)  |
| BG_H(5,2)(mgs) | 1.2(255) | 1.0(196) | 1.3(232)   | 2.3(420)   | 4.1(672)   | 6.1(1020)  |
| BeEN(4)(cgs)   | 0.4(85)  | 0.9(172) | 1.0(168)   | 1.6(252)   | 1.8(272)   | 2.1(300)   |
| BBEN(4)        | 0.5(91)  | fail     | 1.7(280)   | 2.1(348)   | 2.0(304)   | 2.7(380)   |
| BGEN(4)        | 0.5(81)  | 1.0(172) | 1.1(184)   | 1.5(252)   | 2.3(368)   | 2.2(340)   |
| BGBM(4)        | 1.1(117) | 3.4(428) | 6.2(736)   | 6.8(684)   | 9.8(912)   | 11.9(1120) |
| BtEN(4)(cgs)   | 0.5(109) | 1.9(500) | 1.9(456)   | 1.8(396)   | 1.9(400)   | 2.5(500)   |
| BGBM1(4)       | 2.1(260) | fail     | fail       | fail       | fail       | fail       |
| BBBM(4)        | fail     | fail     | fail       | fail       | fail       | fail       |
| BBBM1(4)       | 1.4(161) | fail     | fail       | fail       | fail       | fail       |
| BG(8)(cgs)     | 0.6(100) | 1.5(256) | 1.7(224)   | 2.2(264)   | 3.0(304)   | 3.9(360)   |
| BG_H(8,1)(cgs) | 0.6(100) | 1.0(196) | 1.5(216)   | 2.3(336)   | 3.2(480)   | 4.0(660)   |
| BG_H(8,2)(cgs) | 0.7(131) | 0.9(176) | 1.4(258)   | 2.5(456)   | 3.9(720)   | 5.5(1060)  |

in the number of iterations. For example, BeEN(4) takes only about 5 times as long for  $s = 20$  than for  $s = 1$ , i.e., it is about 4 times as fast to use BeEN(4) versus eEN(4) for 20 different linear systems.

The results for the symmetric, but very ill-conditioned, matrix SAYLR4 from the Harwell–Boeing collection are given in Table 6. We used a Krylov subspace size of 8. Here, the use of ‘mgs’ improved the times by about 20 percent for one right-hand side, but increases the times significantly for larger  $s$  with a comparable number of sparse matrix vector multiplications. For this example, the truncated methods perform overall better than the restarted methods. The alternative implementations for BtEN and block Orthomin were used, since they were somewhat faster on this particular machine. This results in the fairly large number of spmv's listed. For BtBEN(7), the use of  $R_k = B - \mathcal{A}X_k$  improved the convergence for  $s = 4$  by a factor of 1.4 and for  $s = 8$  by a factor of 4. Except for  $s = 20$ , the fastest method overall is BtEN(7). For  $s = 20$ , BeEN(7) is the fastest method.

The times for BG(14) and the corresponding hybrid schemes are included, since the memory usage of BG(14) is comparable to that of BeEN(7) and BtEN(7). Since the times for block GMRES using a larger Krylov subspace are significantly improved for  $s > 1$ , the results for the hybrid schemes are mixed.

Table 6

Times in seconds (number of spmv's) for SAYLR4 with ILU(0)

| Method $s$      | 1        | 4          | 8           | 12          | 16          | 20          |
|-----------------|----------|------------|-------------|-------------|-------------|-------------|
| BG(8)(cgs)      | 5.5(822) | 46.2(6204) | 67.3(8072)  | 98.9(8760)  | 72.4(6480)  | 71.8(4920)  |
| BG(8)(mgs)      | 4.7(919) | 40.4(5396) | 100.2(8200) | 135.7(7428) | 160.6(7008) | 125.4(4860) |
| BG_H(8,1)(cgs)  | 3.2(540) | 22.0(3056) | 23.8(3208)  | 4.9(624)    | 5.7(816)    | 7.6(780)    |
| BG_H(8,1)(mgs)  | 2.8(540) | 22.4(3004) | 22.9(2528)  | 7.0(624)    | 9.1(816)    | 12.8(780)   |
| BG_H(8,2)(cgs)  | 4.0(705) | 10.4(1944) | 4.4(680)    | 3.5(528)    | 6.2(816)    | 11.9(1980)  |
| BG_H(8,2)(mgs)  | 3.8(727) | 11.9(1808) | 5.1(680)    | 4.9(528)    | 8.5(816)    | 14.6(1180)  |
| BeEN(7)(cgs)    | 5.1(665) | 4.2(604)   | 3.8(456)    | 3.8(396)    | 4.8(496)    | 4.1(420)    |
| BBEN(7)         | 4.6(625) | fail       | fail        | 29.4(3060)  | 19.7(1808)  | 8.2(700)    |
| BGEN(7)         | 8.3(923) | fail       | fail        | fail        | fail        | 8.8(700)    |
| BtEN(7)         | 1.1(177) | 2.2(372)   | 2.4(424)    | 3.7(540)    | 4.5(656)    | 5.5(740)    |
| BOrthomin(7)    | 2.4(357) | 4.0(572)   | 5.7(712)    | 10.6(1284)  | 12.9(1264)  | fail        |
| BtBEN(7)        | 1.2(125) | 2.8(260)   | 5.4(440)    | 3.8(348)    | 4.6(400)    | 5.6(420)    |
| BG(14)(cgs)     | 7.0(781) | 7.3(744)   | 7.1(560)    | 7.3(528)    | 7.4(464)    | 7.6(440)    |
| BG_H(14,1)(cgs) | 3.1(369) | 6.2(672)   | 3.9(376)    | 6.4(516)    | 8.2(704)    | 7.8(720)    |
| BG_H(14,2)(cgs) | 2.7(339) | 3.5(432)   | 3.7(376)    | 6.8(660)    | 8.9(944)    | 8.8(1120)   |

## 7. Conclusions and future research

Overall, the results demonstrate the general superiority of the hybrid block GMRES and block EN-like methods over block GMRES and block Broyden methods. The block EN-like methods are therefore a very promising form of the block quasi-Newton approach to the problem. The relative performance of the block EN-like methods and the block GMRES hybrid depends on the problem and both are effective generally. Further work will be done on the parallel and memory efficiency of these methods, adapting some of the hybrid techniques to the quasi-Newton block EN-like method setting and on the examination of nonlinear block EN-like methods.

## Acknowledgements

The authors thank Stratis Gallopoulos and Valeria Simoncini for providing their hybrid block code for use in the experiments and for their helpful comments on the paper.

## References

- [1] P. Deufhard, R. Freund and A. Walter, Fast secant methods for the iterative solution of large nonsymmetric linear systems, *Impact Comput. Sci. Engrg.* 2 (1990) 244–276.

- [2] T. Eirola and O. Nevanlinna, Accelerating with rank-one updates, *Linear Algebra Appl.* 121 (1989) 511–520.
- [3] U. Meier Yang, A family of preconditioned iterative solvers for sparse linear systems, Ph.D. Thesis, Report UIUCDCS-R-95-1904, Department of Computer Science, University of Illinois (1995); also available as CSRD-Report 1408 through anonymous ftp to in directory pub/CSRD\_Reports/reports.
- [4] U. Meier Yang and K. Gallivan, A new family of preconditioned iterative solvers for nonsymmetric linear systems, *Appl. Numer. Math.* 19 (1995) 287–317.
- [5] D. O’Leary, The block conjugate gradient algorithm and related methods, *Linear Algebra Appl.* 29 (1980) 293–322.
- [6] D. O’Leary and A. Yeremin, The linear algebra of block quasi-Newton algorithms, *Linear Algebra Appl.* 212/213 (1994) 153–168.
- [7] V. Simoncini and E. Gallopoulos, An iterative method for nonsymmetric systems with multiple right-hand sides, *SIAM J. Sci. Comput.* 16 (1995) 917–933.
- [8] V. Simoncini and E. Gallopoulos, A hybrid block GMRES method for nonsymmetric systems with multiple right-hand sides, *J. Comput. Appl. Math.* 66 (1996) 457–469.
- [9] G.W. Stewart, *Introduction to Matrix Computations* (Academic Press, New York, 1973).
- [10] E. de Sturler, Nested Krylov methods based on GCR, *J. Comput. Appl. Math.* 67 (1996) 15–41.
- [11] B. Vital, Etude de quelques méthodes de résolution de problèmes linéaires de grande taille sur multiprocesseur, Ph.D. Thesis, Université de Rennes I, Rennes (1990).
- [12] C. Vuik and H. van der Vorst, A comparison of some GMRES-like methods, *Linear Algebra Appl.* 160 (1992) 131–162.