Stability analysis and improvement of the block Gram-Schmidt algorithm

William Jalby, Bernard Philippe

INRIA

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tel:(1)39 63 55 11

Rapports de Recherche

N° 1162

Programme 2
Structures Nouvelles d'Ordinateurs

STABILITY ANALYSIS AND IMPROVEMENT OF THE BLOCK GRAM-SCHMIDT ALGORITHM

William JALBY
Bernard PHILIPPE

Février 1990

RR_1162

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CÉDEX
FRANCE
Téléphone : 99 36 20 00
Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Publication Interne n°509 - Janvier 1990 - 24 Pages

STABILITY ANALYSIS AND IMPROVEMENT OF THE BLOCK GRAM-SCHMIDT ALGORITHM [1]

ETUDE DE LA STABILITE ET AMELIORATION DE LA VERSION PAR BLOCS DE L'AGORITHME DE GRAM-SCHMIDT

William Jalby and Bernard Philippe
IRISA Campus de Beaulieu 35042 RENNES CEDEX FRANCE

[1]Submitted to SIAM Journal on Scientific and Statistical Computing

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L. A. 227)
UNIVERSITÉ DE RENNES 1 I. N. S. A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

Abstract

The advent of supercomputers with hierarchical memory systems has imposed the use of block algorithms for the linear algebra algorithms. Although block algorithms may result in impressive improvements in performance, their numerical properties are quite different from their scalar counterpart and deserve an in depth study. In this paper, the numerical stability of Block Gram Schmidt orthogonalization is studied and a variant is proposed which has numerical properties similar to the classical Modified-Gram-Schmidt while retaining most of the performance advantages of the block formulation.

Keywords: Gram-Schmidt orthogonalization, stability, block algorithms.

Résumé

L'apparition de mémoires hiérarchisées dans les supercalculateurs a entraîné l'emploi d'algorithmes par blocs en algèbre linéaire. Alors que ces algorithmes permettent souvent d'importants progrès dans les performances, leurs propriétés numériques peuvent nettement différer de celles des algorithmes généraux; ils nécessitent donc un étude précise. Cet article caractérise la situation dans le cas de l'algorithme de l'orthogonalisation de Gram-Schmidt. On en propose alors une variante dont les propriétés numériques sont du même ordre que celles de l'algorithme habituel (Gram-Schmidt modifié) et qui conserve la plus grande partie des avantages de la version par blocs.

Mots clés: Orthogonalisation de Gram-Schmidt, stabilité, algorithmes par blocs.

Contents

1 Introduction

The so-called Gram-Schmidt algorithm is one of the most widely used algorithms to orthogonalize a set of vectors. Let $A \in \Re^{n \times m}$ $(n \geq m)$ be the matrix of which columns (a_i) have to be orthogonalized. The Gram-Schmidt method generates the orthogonal matrix $Q = [q_1; \cdots; q_m]$, which corresponds to the Q factor of the QR decomposition of A, according to the following scheme:

$$
\begin{array}{l}
q_1 := a_1 / \parallel a_1 \parallel \ ; \\
\text{for} \quad k = 1,\ m-1 \\
\qquad\qquad a'_{k+1} := L_k a_{k+1} \ ; \\
\qquad\qquad q_{k+1} := a'_{k+1} / \parallel a_{k+1} \parallel \ ; \\
\text{endfor} \ ;
\end{array}
$$

where L_k stands for the projection onto the orthogonal complement of the subspace spanned by $(q_1, \cdots, q_k)$. Several implementations of the procedure lie under this common presentation (depending upon the representation of L_k).

If V_k denotes the matrix $[q_1; \cdots; q_k]$, L_k may be expressed as the matrix $I - V_k V_k^t$; this corresponds to the so-called classical version (CGS). The matrix L_k may also be expressed as the composition of the projections onto the orthogonal complements of the one dimensional subspaces spanned by $q_1, \cdots, q_k$:

$$L_k = (I - q_k q_k^t) \cdots (I - q_1 q_1^t) \tag{1}$$

The corresponding algorithm is the so-called Modified version (MGS).

Although (CGS) and (MGS) are equivalent in exact arithmetic, they behave quite differently in finite arithmetic: (CGS) is considered unreliable, as we will see in the next section. The advent of vector and parallel computers using hierarchical memory systems has emphasized the use of matrix multiplication as an efficient primitive which allows efficient data management, and has considerably renewed the interest in block algorithms based on matrix multiplications [2, 3]. For example, in the case of the Gram-Schmidt procedure, a natural block version can be obtained by using blocks of consecutive columns to orthogonalize the blocks with respect to each other, (MGS) being used to orthogonalize inside a block. In such a case, L_k can be expressed as:

$$L_k = (I - q_k q_k^t) \cdots (I - q_l q_l^t)(I - Q_\beta Q_\beta^t) \cdots (I - Q_1 Q_1^t) \tag{2}$$

by assuming that q_{k+1} is a column of the block $Q_{\beta+1}$, whose first column is q_l. The corresponding algorithm is:

$$
\begin{array}{l}
Q_1 := MGS(A_1) \ ; \\
\text{for} \quad \beta = 1,\ \nu - 1 \\
\qquad\qquad B_{\beta+1} := A_{\beta+1} \ ; \\
\qquad\qquad \text{for} \quad \alpha = 1, \beta \\
\qquad\qquad\qquad\qquad S := Q_\alpha^t B_{\beta+1} \ ;
\end{array}
$$

$$B_{\beta+1} := B_{\beta+1} - S * Q_\alpha \ ;$$

endfor ;

$$Q_{\beta+1} := MGS(B_{\beta+1}) \ ;$$

endfor ;

This paper addresses the estimation of the numerical reliability of (BGS) which, when quickly examined, exhibits some similarities with (CGS). We prove that this is partially true and that it can be improved to achieve the same quality as (MGS) at the price of an increase in complexity by a factor of $(1+1/\nu)$, where ν is the number of blocks (supposed of equal size). The estimation of the loss of orthogonality is heavily based on results obtained by Bjorck, who first characterized the situation for (MGS) [1].

2 An illustration of the scene

Bjorck [1] proved that if $(\bar{Q}, \bar{R})$ are the factors obtained by the (MGS) procedure, there exist constants $(K_i)_{i=1,3}$, only depending upon n and m, such that [1]:

$$\begin{array}{lcl} \| \bar{Q}\bar{R} - QR \|_F & \leq & K_1 \| A \|_F \, \epsilon \\ \| \bar{Q}^t\bar{Q} - I \|_2 & \leq & K_2 \| A \|_F \, \| R^{-1} \|_2 \, \epsilon \\ \| U\bar{R} \|_F & \leq & K_3 \| A \|_F \, \epsilon \end{array}$$

where (Q, R) are the exact factors of A, U is the upper triangular part of the matrix $\bar{Q}^t\bar{Q} - I$, and ϵ is the precision parameter. These results demonstrate the importance of the condition number of A ($\chi_2(A)$), which satisfies:

$$\chi_2(A) \ \leq \ \| A \|_F \, \| R^{-1} \|_2 \ \leq \sqrt{m} \, \chi_2(A)$$

For instance, let us consider the matrix:

$$A = \begin{pmatrix} 1 & 1 & 1 \\ \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & \sigma \end{pmatrix}$$

Its condition number is $\chi(A) = \sqrt{3+\sigma^2}/\sigma$. The loss of orthogonality when applying (CGS) or (MGS) is reported in Table 1 for several values of σ.

It should be noted with (BGS) that for a block size of 1 or m it corresponds exactly to (MGS); the only case where (BGS) corresponds to (CGS) and not to (MGS) arises when $m = 3$ for a blocksize of 2. Then, the very special example described above can be viewed as an illustration of the potential bad properties of (BGS).

[1] In this paper, bounds are expressed with constants K, which only depend upon n and m, and the blocksize p. We do not provide precise estimation of these constants because we feel that they are very pessimistic and would only muddle the presentation. They are interesting because of their existence. For the same reason, we systematically neglect the $O(\epsilon^2)$ terms.

σ	$\chi(A)$	$\| \bar{Q}^t\bar{Q} - I \|_2$	
		MGS	CGS
10^{-4}	$\sqrt{3}.10^4$	3.10^{-13}	2.10^{-9}
10^{-5}	$\sqrt{3}.10^5$	7.10^{-13}	5.10^{-8}
10^{-6}	$\sqrt{3}.10^6$	7.10^{-11}	5.10^{-5}
10^{-7}	$\sqrt{3}.10^7$	2.10^{-9}	1.10^{-2}

Table 1: Loss of orthogonality in (MGS) and (CGS) (Experiments performed with MATLAB and $\epsilon = 2^{-52} \approx 2.10^{-16}$

3 Error in a projection

The elementary step of (BGS) is the projection of a block A_β^α onto the orthogonal complement of $r(Q_\alpha)$ where $\alpha < \beta$ and $r(Q_\alpha)$ is the vector space span by the columns of Q_α. This step corresponds to the computation involved in the innermost loop of (BGS). The sequence A_β^α is then defined by:

$$\begin{cases} A_\beta^1 = A_\beta \text{ and} \\ A_\beta^{\alpha+1} = P_\alpha A_\beta^\alpha \text{ where } P_\alpha = I - Q_\alpha Q_\alpha^t \text{ for } 1 < \alpha < \beta \leq \nu \end{cases} \tag{3}$$

Due to rounding error effects, the computed sequences are not $\{A_\beta^\alpha\}$ and $\{Q_\alpha\}$, but $\{\bar{A}_\beta^\alpha\}$ and $\{\bar{Q}_\alpha\}$ and the recurrence formula (3) becomes:

$$\begin{cases} \bar{A}_\beta^1 = A_\beta \text{ and} \\ \bar{A}_\beta^{\alpha+1} = \bar{A}_\beta^\alpha - \bar{Q}_\alpha \bar{R}_\beta^\alpha + \Delta_\beta^\alpha \quad 1 \leq \alpha < \beta \leq \nu \end{cases} \tag{4}$$

where $\bar{R}_\beta^\alpha = \bar{Q}_\alpha^t \bar{A}_\beta^\alpha$. Let $\bar{P}_\alpha = I - \bar{Q}_\alpha \bar{Q}_\alpha^t$.

Because our purpose is to look at the loss of orthogonality, we do not assume that $\bar{Q}_\alpha^t \bar{Q}_\alpha = I$. Let $D_\alpha = \bar{Q}_\alpha^t \bar{Q}_\alpha - I$. In such a case, $\bar{P}_\alpha$ is, a priori, no longer a projection. Let us decompose any vector u as the sum of its components over $r(\bar{Q}_\alpha)$ and $r(\bar{Q}_\alpha)^\perp$, i.e.:

$$u = \bar{Q}_\alpha v + w \quad \text{with} \quad \bar{Q}_\alpha^t w = 0$$

Then, we obtain:

$$\bar{P}_\alpha u = -\bar{Q}_\alpha D_\alpha v + w$$

which proves that $\bar{P}_\alpha$ is the identity over $r(\bar{Q}_\alpha)^\perp$, but is not the null operator over $r(\bar{Q}_\alpha)$, as it would be if $\bar{Q}_\alpha$ was orthogonal.

Below are some estimations which will be useful for the following section:

Lemma 3.1 *Let $\bar{Q}_\alpha$ and D_α defined as above, then the following inequalities hold:*

- $\| \bar{Q}_\alpha \|_2 = \sqrt{1 + \| D_\alpha \|_2} \leq 1 + 1/2 \| D_\alpha \|_2$
- *under the assumption that $\| \bar{Q}_\alpha D_\alpha \|_2 \leq 1$:*

$$\| \bar{P}_\alpha \|_2 = 1$$

Proof : Obvious. □

Let η be the largest number such that $\eta(1+\eta/2) \leq 1$ ($\eta \approx 0.78$). We will assume that

$$\text{for all } \alpha, \quad \| D_\alpha \|_2 \leq \eta \tag{5}$$

and therefore $\| \bar{P}_\alpha \|_2 = 1$.

The error Δ_β^α is bounded by the following estimation:

Proposition 3.1 *There exists a sequence of constant numbers K_β^α such that:*

$$\| \Delta_\beta^\alpha \|_F \leq K_\beta^\alpha \| \bar{A}_\beta^\alpha \|_F \epsilon$$

Proof : The computation of $\bar{A}_\beta^{\alpha+1}$ consists of three steps. Dropping indices α and β for sake of clarity, these steps are:

$$\begin{array}{lllll} W^{(1)} & = & fl(\bar{Q}^t \bar{A}) & = & \bar{Q}^t \bar{A} + V^{(1)} \\ W^{(2)} & = & fl(\bar{Q} W^{(1)}) & = & \bar{Q}\bar{Q}^t A + \bar{Q} V^{(1)} + V^{(2)} \\ W^{(3)} & = & fl(\bar{A} - W^{(2)}) & = & \bar{A} - \bar{Q}\bar{Q}^t A - \bar{Q} V^{(1)} - V^{(2)} + V^{(3)} \end{array}$$

Using the classical results on the error bounds for matrix multiplications [4], we derive the following inequalities:

$$\begin{array}{lll} \| V^{(1)} \|_F & \leq & K(n) \| \bar{Q} \|_2 \| \bar{A} \|_F \epsilon \qquad \text{and then} \\ \| Q V^{(1)} \|_F & \leq & K(n) \| \bar{Q} \|_2^2 \| A \|_F \epsilon \qquad \text{and} \\ \| V^{(2)} \|_F & \leq & K(p) \| \bar{Q} \|_2^2 \| A \|_F \epsilon + O(\epsilon^2) \end{array}$$

where $K()$ is a polynomial of low order and $\bar{Q} \in \Re^{n \times p}$.

The error due to the substraction may be bounded by:

$$\begin{array}{lll} \| V^{(3)} \|_F & \leq & \| \bar{A} - W^{(2)} \|_F \epsilon \\ & \leq & \| \bar{I} - \bar{Q}\bar{Q}^t \|_2 \| A \|_F \epsilon + O(\epsilon^2) \\ & \leq & \| A \|_F \epsilon + O(\epsilon^2) \end{array}$$

since we have assumed (5). Putting together all these bounds we obtain:

$$\begin{array}{lll} \| \Delta \|_F & \leq & \| \bar{Q} V^{(1)} \|_F + \| V^{(2)} \|_F + \| V^{(3)} \|_F \\ \| \Delta \|_F & \leq & K \| A \|_F \epsilon \end{array}$$

□

Lemma 3.2 *There exists a constant $\tau = O(\epsilon)$ such that $\| \bar{A}_\beta^\alpha \|_F \leq (1+\tau)^{\alpha-1} \| A_\beta \|_F$ for all $\alpha \leq \beta \leq \nu$.*

Proof : Obvious from Lemma 3.1 and Proposition 3.1 with

$$\tau = max_{\alpha \leq \beta \leq \nu} (K_\beta^\alpha \epsilon)$$

□

Once $\bar{A}_\beta^\beta$ is computed, a Modified Gram-Schmidt process is applied to get $\bar{Q}_\alpha$.

Let R'_β and $\bar{\bar{R}}_\beta$ be the exact and the computed R-factor of the QR decomposition of $\bar{A}_\beta^\beta$, respectively. From [1], we have the following estimations:

$$\begin{array}{lcl} \| \bar{A}_\beta^\beta - \bar{Q}_\beta \bar{\bar{R}}_\beta \|_F & \leq & K_1 \| \bar{A}_\beta^\beta \|_F \epsilon \\ \| \bar{Q}_\beta^t \bar{Q}_\beta - I \|_2 & \leq & K_2 \| \bar{A}_\beta^\beta \|_F \| {R'_\beta}^{-1} \|_2 \epsilon \\ \| (\bar{Q}_\beta^t \bar{Q}_\beta - I) \bar{\bar{R}}_\beta \|_F & \leq & K_3 \| \bar{A}_\beta^\beta \|_F \epsilon \end{array}$$

Let $C(\beta) = \| \bar{A}_\beta^\beta \|_F \| {R'_\beta}^{-1} \|_2$. If orthogonalization was exact, we would have $\| A \|_F \| R^{-1} \|_2 \geq c(\beta)$ and generally, it would be far from equal. We assume that it is still the case with finite arithmetic. The quantity $C = max_{\beta \leq \nu} C(\beta)$ is of importance in the sequel.

Let $\bar{R}_\beta^\beta$ be the matrix $\bar{Q}_\beta^t \bar{A}_\beta^\beta$. From the equality:

$$\bar{R}_\beta^\beta - \bar{\bar{R}}_\beta = (\bar{Q}_\beta^t \bar{Q}_\beta - I) \bar{\bar{R}}_\beta + \bar{Q}_\beta^t (\bar{A}_\beta^\beta - \bar{Q}_\beta \bar{\bar{R}}_\beta)$$

we obtain:

$$\| \bar{R}_\beta^\beta - \bar{\bar{R}}_\beta \|_F \leq (K_3 + \| \bar{Q}_\beta \|_2 K_1) \| \bar{A}_\beta^\beta \|_F \epsilon$$

and if

$$\Delta_\beta^\beta = -\bar{A}_\beta^\beta + \bar{Q}_\beta \bar{R}_\beta^\beta \tag{6}$$

we may write:

$$\Delta_\beta^\beta = -\bar{A}_\beta^\beta + \bar{Q}_\beta \bar{\bar{R}}_\beta + \bar{Q}_\beta (-\bar{\bar{R}}_\beta + \bar{R}_\beta^\beta)$$

which implies the following proposition:

Proposition 3.2 *There exists a constant K_β^β such that:*

$$\| \Delta_\beta^\beta \|_F \leq K_\beta^\beta \| \bar{A}_\beta^\beta \|_F \epsilon$$

4 Loss of orthogonality in the whole process

By adapting Bjorck's procedure for blocks, we define the following matrices:

$$U_\beta^\alpha = \begin{cases} \bar{Q}_\alpha^t \bar{Q}_\beta & \text{for } 1 \leq \alpha < \beta \leq \nu \\ \text{and} & \\ 0 & \text{for } 1 \leq \beta \leq \alpha \leq \nu \end{cases}$$

and:

$$D_\alpha = \bar{Q}_\alpha^t \bar{Q}_\alpha - I \text{ for } 1 \leq \alpha \leq \nu$$

Let U and D be the upper block triangular and the block diagonal matrices defined by the blocks $\{U_\beta^\alpha\}$ and $\{D_\alpha\}$ respectively.

From the previous section, we may ensure that:

$$\| D \|_2 \leq K_2 C\epsilon \tag{7}$$

where $C = max_\alpha\ C(\alpha)$.

To estimate a bound on $\| U \|_2$, we first estimate a bound of $\| U\bar{R} \|_2$ where $\bar{R}$ is the upper triangular matrix defined by the blocks $\{\bar{R}_\beta^\alpha\}_{1 \leq \alpha \leq \beta \leq \nu}$.

Proposition 4.1 *There exist two constants K_4 and K_5 such that:*

$$\| U\bar{R} \|_F \leq (K_4 C + K_5) \| A \|_F\ \epsilon$$

Proof : By writing equation (4) for $\alpha = \nu + 1, \cdots \beta - 1$, and from the definition of Δ_β^β, we obtain the following equation by accumulating the equalities:

$$\bar{A}_\beta^{\nu+1} = \sum_{\alpha=\nu+1}^{\beta} \bar{Q}_\alpha \bar{R}_\beta^\alpha - \sum_{\alpha=\nu+1}^{\beta} \Delta_\beta^\alpha$$

The block (μ, β) of $U\bar{R}$ can then be expressed as:

$$\begin{aligned} [U\bar{R}]_{\mu\beta} &= \textstyle\sum_{\alpha=\mu+1}^{\beta} U_{\mu\alpha} \bar{R}_\beta^\alpha \\ &= \bar{Q}_\mu^t (\bar{A}_\beta^{\mu+1} + \textstyle\sum_{\alpha=\mu+1}^{\beta} \Delta_\beta^\alpha) \end{aligned}$$

But from equation (4):

$$\begin{aligned} \bar{Q}_\mu^t \bar{A}_\beta^{\mu+1} &= (\bar{Q}_\mu^t - (I + \bar{D}_\mu)\bar{Q}_\mu^t)\bar{A}_\beta^\mu + \bar{Q}_\mu^t \Delta_\beta^\mu \\ &= -\bar{D}_\mu \bar{Q}_\mu^t \bar{A}_\beta^\mu + \bar{Q}_\mu^t \Delta_\beta^\mu \end{aligned}$$

which implies

$$\| \bar{Q}_\mu^t \bar{A}_\beta^{\mu+1} \|_F \leq \| \bar{D}_\mu \|_2 \| \bar{Q}_\mu \|_2 \| \bar{A}_\beta^\mu \|_F + \| \bar{Q}_\mu \|_2 \| \Delta_\beta^\mu \|_F$$

and

$$\| [U\bar{R}]_{\mu\beta} \|_F \leq \| \bar{D}_\mu \|_2 \| \bar{Q}_\mu \|_2 \| \bar{A}_\beta^\mu \|_F + \| \bar{Q}_\mu \|_2 \sum_{\alpha=\mu}^{\beta} \| \Delta_\beta^\alpha \|_F$$

From

$$\| U\bar{R} \|_F \leq \textstyle\sum_{1 \leq \mu \leq \beta \leq \nu} \| [U\bar{R}]_{\mu\beta} \|_F$$

and from lemma 3.2 and propositions 3.1 and 3.2, the result of the proposition is obtained. □

Let $E_1 = \bar{A} - QR$ where $\bar{A} = \bar{Q}\bar{R}$ and (Q, R) is the result of the QR factorization of A. Then the following result can be proved.

Proposition 4.2 *There exists a constant K_6 such that:*

$$\| E_1 \|_F = \| \bar{Q}\bar{R} - QR \|_F \leq K_6 \| A \|_F \epsilon$$

and the condition $\| E_1 \|_F \| R^{-1} \|_2 < \sqrt{2} - 1$ implies that $\bar{A}$ has full rank.

Proof : By summing equations (3) from $\alpha = 1$ to $\alpha = \beta - 1$, we obtain:

$$\bar{A}_\beta^\beta = A_\beta - \sum_{\alpha=1}^{\beta-1} \bar{Q}_\alpha \bar{R}_\beta^\alpha + \sum_{\alpha=1}^{\beta-1} \Delta_\beta^\alpha$$

Since $\bar{A}_\beta^\beta = \bar{Q}_\beta \bar{R}_\beta^\beta - \Delta_\beta^\beta$ it follows that:

$$\| E_1 \|_F = \sum_{\beta=1}^{\nu} \sum_{\alpha=1}^{\beta} \| \Delta_\beta^\alpha \|_F$$

Propositions 3.1 and 3.2 provide bounds for $\{\Delta_\beta^\alpha\}$:

$$\| E_1 \|_F \leq (\sum_{\beta=1}^{\nu} \sum_{\alpha=1}^{\beta} K_\beta^\alpha \| \bar{A}_\beta^\alpha \|_F)\epsilon$$

Let $K_6 = max_{\alpha \leq \beta} K_\beta^\alpha (1 + \tau)^{\alpha - 1}$ where τ is the $O(\epsilon)$ quantity which has been introduced in Lemma 3.2. Then

$$\| E_1 \|_F \leq K_6 \textstyle\sum_{\beta=1}^{\nu} \| \bar{A}_\beta \|_F \epsilon \leq K_6 \| \bar{A} \|_F \epsilon$$

which corresponds to the first part of the proposition.

For the second part, Bjorck's proof can be used:

$$\bar{A}^t \bar{A} = R^t (I + F_1) R \tag{8}$$

where $F_1 = (Q^t E_1 R^{-1})^t + Q^t E_1 R^{-1} + (E_1 R^{-1})^t E_1 R^{-1}$.

Since

$$\| F_1 \|_2 \leq 2 \| E_1 \|_F \| R^{-1} \|_2 + \| E_1 \|_F^2 \| R^{-1} \|_2^2$$

condition $\| E_1 \|_F \| R^{-1} \|_2 < \sqrt{2} - 1$ implies $\| F_1 \|_2 < 1$ and therefore the non singularity of $(I + F_1)$ □

Theorem 4.1 *Let $C = max_\alpha C(\alpha)$. For sufficiently small $(C\epsilon)$ and $(\| A \|_F \| R^{-1} \|_2 \epsilon)$, there exists a constant K_7 such that :*

$$\| I - \bar{Q}^t \bar{Q} \|_2 \leq K_7 C \| A \|_F \| R^{-1} \|_2 \epsilon$$

Proof : From the definitions of matrices U and D, we have:

$$\| I - \bar{Q}^t \bar{Q} \|_2 \leq 2\| U \|_2 + \| D \|_2$$

Since $\| D \|_2$ has already been bounded in 4.1, we only have to consider $\| U \|_2$; because $\| U \|_2 \leq \| U\bar{R} \|_F \| \bar{R}^{-1} \|_2$, we only need to focus on $\| \bar{R}^{-1} \|_2$. Here again, we adapt Bjorck's proof by using the following formula:

$$\bar{R}^t \bar{R} = R^t(I + F_2)R \tag{9}$$

where

$$F_2 = F_1 - (R^{-1})^t(U\bar{R})^t(\bar{R}R^{-1}) - (\bar{R}R^{-1})^t(U\bar{R})R^{-1} - (\bar{R}R^{-1})^t D(\bar{R}R^{-1})$$

with F_1 being the matrix introduced at the end of the proof of Proposition 4.2. From 9, it follows that,

$$\| \bar{R}R^{-1} \|_2^2 \leq 1 + \| F_2 \|_2$$

Let $x = \sqrt{1 + \| F_2 \|_2}$ and $a = \| A \|_F \| R^{-1} \|_2 \epsilon$. From the definition of $F_1$and from Proposition 4.2, we obtain:

$$\begin{aligned} \| F_1 \|_2 &\leq 2\| E_1 \|_F \| R^{-1} \|_2 + \| E_1 \|_F^2 \| R^{-1} \|_2^2 \\ &\leq 2K_6 a + K_6^2 a^2 + O(\epsilon^2) \end{aligned}$$

Then from the definition of F_2, from equation (9) and from Proposition 4.2 we may insure that

$$x^2 \leq 1 + 2K_6 a + K_6^2 a^2 + 2a(K_4 C + K_5)x + K_2 C x^2 \epsilon$$

Then x is such that the following relation is true:

$$(1 - K_2 C\epsilon)x^2 - 2a(K_4 C + K_5)x - 1 - 2K_6 a - K_6^2 a^2 \leq 0$$

If $(C\epsilon)$ is small enough to keep $(1 - K_2 C\epsilon)$ positive, then the polynomial has two zeros: one is negative and the other is greater than 1. The second root is denoted $1 + \rho$; $\rho > 0$ and $\rho = O(\epsilon)$. In conclusion, we have $\| F_2 \|_2 \leq \rho$.

Let us assume that the quantities $(\| A \|_F \| R_{-1} \|_2 \epsilon)$ and $(C\epsilon)$ are small enough to ensure that $\rho < 1$. Then, by considering the inverse of both sides of equation (9), we obtain:

$$\| \bar{R}^{-1} \|_2^2 \leq \| R^{-1} \|_2^2 \| (I + F_2)^{-1} \|_2 \leq (1 - \rho)^{-1} \| R^{-1} \|_2^2$$

This last bound ends the proof of the theorem.

□

The worst bound can be obtained when one block has a condition number almost equal to the condition number of the whole sytem; in that situation, the quantity $(C\| A \|_F \| R^{-1} \|_2)$ is of the order of the square of the condition number of A.

5 (B2GS) A block method as stable as (MGS)

Theorem 4.1 shows the importance of the constant C in the error bound. Even when it is small, the error which occurs within the orthogonalization of a block may significantly impact on the precision of the following steps. By reorthonormalizing every block, i.e. by applying twice (MGS) to every block, the constant C would disappear from the estimation, since it would correspond to the QR factorization of the $\{\bar{Q}_\beta\}$ blocks, which have condition numbers close to unity. This remark leads directly to the algorithm (B2GS) which has exactly the same structure as (BGS) except that the reorthonormalization procedure is applied twice on each block.

To illustrate the stability improvement in (B2GS) compared to (BGS), some numerical results are presented below.

For the first serie of experiments, the results obtained by (BGS) and (B2GS) are compared using MATLAB. The matrix under consideration is the so-called Hilbert matrix:

$$A = (a_{ij}) \in \Re^{20\times 10} \quad \text{where} \quad a_{ij} = 1/(i+j-1)$$

The characteristics of the matrix are :

$$\begin{array}{lcl} \| A \|_F & = & 1.9 \\ \| R^{-1} \|_2 & = & 1.4\ 10^{11} \\ \chi(A) & = & 2.6\ 10^{11} \end{array}$$

The value of the residuals, with respect to the block size, are reported in Table 2.

Block size	1	2	3	4	5
(BGS)	$5.2\ 10^{-6}$	$2.8\ 10^{-6}$	$2.3\ 10^{-5}$	$1.1\ 10^{-4}$	$5.2\ 10^{-3}$
(B2GS)	$5.2\ 10^{-6}$	$3.0\ 10^{-6}$	$4.3\ 10^{-6}$	$3.1\ 10^{-6}$	$4.0\ 10^{-6}$

Table 2: Comparison of loss of orthogonality for (BGS) and (BGS) with reorthonogonalization for each block size on a Hilbert matrix (MATLAB)

A second serie of experiments was performed by implementing the codes corresponding to the three algorithms ((MGS), (BGS) and (B2GS)) in FORTRAN on the CRAY2. The matrix used was a matrix A of size 1024×512 obtained by the multiplication of two matrices M and H such that:

$$\begin{array}{ll} A = MH & \text{where} \\ M \in \Re^{1024\times 512} & m_{ij} = 1 \qquad j = 1,512 \\ & m_{ij} = \delta_{i-1,j}\epsilon \quad i = 2,1024 \text{ and } j = 1,512 \\ H \in \Re^{512\times 512} & h_{ij} \text{ randomly selected in [-1,+1]} \end{array}$$

where ϵ is the machine precision parameter. The condition number of the resulting matrix A was estimated by a LINPACK procedure and its value was $\chi(A) = 0.503E+08$.

(MGS) residual: $0.423\ 10^{-8}$		
Block Size	(BGS) Residuals	(B2GS) Residuals
16	$0.115\ 10^{-4}$	$0.140\ 10^{-8}$
32	$0.255\ 10^{-4}$	$0.353\ 10^{-8}$
48	$0.775\ 10^{-4}$	$0.286\ 10^{-8}$
64	$0.997\ 10^{-4}$	$0.279\ 10^{-8}$
80	$0.104\ 10^{-3}$	$0.298\ 10^{-8}$
96	$0.724\ 10^{-4}$	$0.299\ 10^{-8}$
112	$0.876\ 10^{-4}$	$0.454\ 10^{-8}$
128	$0.941\ 10^{-4}$	$0.549\ 10^{-8}$
144	$0.102\ 10^{-3}$	$0.342\ 10^{-8}$
160	$0.693\ 10^{-4}$	$0.483\ 10^{-8}$
176	$0.752\ 10^{-4}$	$0.620\ 10^{-8}$
192	$0.101\ 10^{-3}$	$0.708\ 10^{-8}$
208	$0.117\ 10^{-3}$	$0.123\ 10^{-7}$

Table 3: Comparison of residuals for (MGS), (BGS) and (B2GS)

Table 3 compares the Frobenius norm of the residuals for various block-sizes and for (BGS) and (B2GS). These results show clearly the interest of (B2GS) which gives results with an improvement in the order of magnitude of 4 compared to the standard (BGS). Furthermore, for all the block-sizes, (B2GS) achieve results very close to the standard (MGS) procedure (results of which are given in the first row of table 3).

6 Performance analysis of (B2GS)

Since one of the major advantage of block algorithms is to provide a good data locality together with a good potential for vectorization and parallelization, our performance analysis will be carried out along two axes: number and characteristics of the arithmetic operations (i.e. number of operations, vectorization and parallelization properties) and data locality. First, the methodology for the performance analysis will be briefly described, then the two axes mentioned above will be studied. The behavior of (B2GS) will be systematically compared with the (BGS) one and the choice of the block size will be analyzed. Finally, some experimental results on an ALLIANT FX80 and a CRAY-2 will be presented.

6.1 Performance model

Throughout all section 6, we will use the same framework as the one used in [2] [3] for studying block algorithms. Let us recall briefly its major characteristics. The target

machine will be assumed to be a shared memory multi-(vector) processor, using a memory systems consisting of a cache (of size CS) with fast access and a large memory with slower access. According to the methodology developed in [2] [3], the total execution time will be split into two components:

1. T_a: Arithmetic time: this represents the total computation time assuming that the cache has an infinite size and that initially all the data reside in cache
2. T_l: Load time: this represents the extra time spent in the loads to be performed from the memory due to the finite cache size (this includes the initial data loads as well as the ones occurring because all the data cannot fit in the cache)

The first component (T_a) takes into account all the parallelization and vectorization properties of the algorithm studied while the second one (T_l) attempts to quantify its data locality characteristics. This second measure is extremely difficult to evaluate accurately because it depends upon many intrinsic details of a particular cache organization; as it was proposed in [2] [3], instead of considering the problem of optimizing T_l we will consider the problem of optimizing N_l which is the total number of loads from memory. For determining N_l, we will assume that the data transfers between the cache and the memory are under software control (i.e. we can specify what are the data which will reside in cache). Such a measure is much simpler to compute and allows to capture most of the trends in performance behavior of the algorithm. Furthermore, we will evaluate the ratio $\mu = N_l/N_a$ where N_a is the total number of floating point operations. This quantity allows to get a better appreciation of the relative cost of the loads from memory.

Our analysis of (B2GS) is greatly simplified due to the fact that the differences between (BGS) and (B2GS) are minor. For example since (B2GS) is using exactly the same computational primitives as (BGS), all the results relative to the parallelization and vectorization properties can be carried out from (BGS) to (B2GS); the only difference is the different weight associated with each primitive.

6.2 Optimization of the arithmetic time

First, let us evaluate the cost in terms of number of floating point operations of the extra computation involved in the reorthonormalization. The regular version of (BGS) has the same complexity as (MGS) or (CGS). For a $n \times m$ system, this number of operations is given by:

$$N_{op}(BGS) = N_{op}(MGS) \simeq 2nm^2$$

Let us assume that $m = k\omega$ where ω is the block size. Then, an extra reorthonormalization of every block leads to a total of number of operations for (B2GS) given by:

$$N_{op}(B2GS) \simeq 2nm^2 + 2kn\omega^2 = N_{op}(BGS) + 2nm\omega \tag{10}$$

This clearly shows the interest of keeping ω relatively small. Additionnaly, it also indicates that the idea of varying block sizes is not worth from the arithmetic point of view. In fact, if we assume non constant block sizes (i.e. a sequence $\{\omega_i\}_{i=1,k}$), the extra

cost associated with the reorthogonalization step is $\sum_{i=1}^{k} n\omega_i^2$. For a fixed number of blocks, this quantity will be minimal if all the block sizes are constant and equal to m/k In terms of relative costs:

$$\frac{N_{op}(B2GS)}{N_{op}(BGS)} = 1 + \frac{\omega}{m} = 1 + \frac{1}{k}$$

Besides the extra number of operations, (B2GS) differs from (BGS) by the work repartition between the different primitives. Although all the primitives used in (BGS) (and therefore in (B2GS)) offer a good degree of vectorization and parallelization, there will be a sensible discrepancy in performance between the block factorization ((MGS) primitive on a block) and the matrix multiply primitives. This is due to the fact that matrix multiply lends itself to a very efficient use of chaining and registers and has a much simpler synchronization graph than the (MGS) primitive.

More formally, if V_{MGS} (resp. V_{MAT}) denotes the speed in megaflops of the (MGS) primitive (resp. the matrix multiply primitive), we end up with:

$$T_a(BGS) = \frac{2mn\omega}{V_{MGS}} + \frac{2mn(n-\omega)}{V_{MAT}}$$

$$T_a(B2GS) = \frac{4mn\omega}{V_{MGS}} + \frac{2mn(n-\omega)}{V_{MAT}}$$

The difference between (BGS) and (B2GS) is therefore:

$$\frac{T_a(B2GS) - T_a(BGS)}{T_a(BGS)} = \frac{\frac{V_{MAT}}{V_{MGS}}}{\frac{V_{MAT}}{V_{MGS}} + k - 1}$$

If we assume that $\frac{V_{MAT}}{V_{BGS}} \leq 6$ which is reasonable for many practical machines, this gives:

$$\frac{T_a(B2GS) - T_a(BGS)}{T_a(BGS)} \leq \frac{6}{k-5}$$

This last estimation insures that as long as the number of blocks (k) is large enough, the discrepancy in performance (due to the extra arithmetic operations) between BGS and $B2GS$ will be low. In practice, for large problems, it is relatively easy to obtain a large number of blocks: for instance, for 512 vectors and a block size of 16, the discrepancy is less than 22%.

6.3 Optimization of data locality

Again due to the fact that (B2GS) uses the same primitives as (BGS), similar conclusions can be derived: increasing the block size increases data locality. For simplifying the analysis, we can assume without loss of generality that $\omega < CS/n$; this corresponds to the case where the orthogonalization of a block $m \times \omega$ fits in cache. In the case of (B2GS),

such a restriction is really not too constraining due to the fact that the use of block sizes larger than CS/n would generally result in a prohibitive penalty for the number of floating operations.

The good point of (B2GS) is that the two orthogonalizations primitives are applied one after each other for each block, this means that no additional loads will be involved due to reorthonormalization:

$$N_l(B2GS(\omega)) = N_{load}(BGS(\omega))$$

Using the results presented in [3] on (BGS) behavior, we derive that:

$$\mu(B2GS) = \frac{1}{m}(1 + k - \frac{2}{k}) + \frac{1}{n}(1 - \frac{1}{k})$$

As expected, μ is an increasing function with respect to k. This trend is exactly contradictory with the one observed for the optimization of the arithmetic time. A precise determination of the best choice of the number of blocks requires a precise knowledge of the cost of the memory fetch versus a floating point operation in order to find the right trade off between arithmetic time minimization and memory loads minimization. However, it should be noted that a value of $k = 10$, which gives a relatively good arithmetic optimization, would result in a value for $\mu(B2GS)$ given by:

$$\mu(B2GS) \simeq \frac{10.8}{m} + \frac{0.9}{n}$$

which gives reasonably good value for μ under the conditions that m and n are large enough. In fact, if these conditions are not met, as we will see in the next section on experimental results, (B2GS) performance as well (BGS) are sensibly affected.

6.4 Experimental results

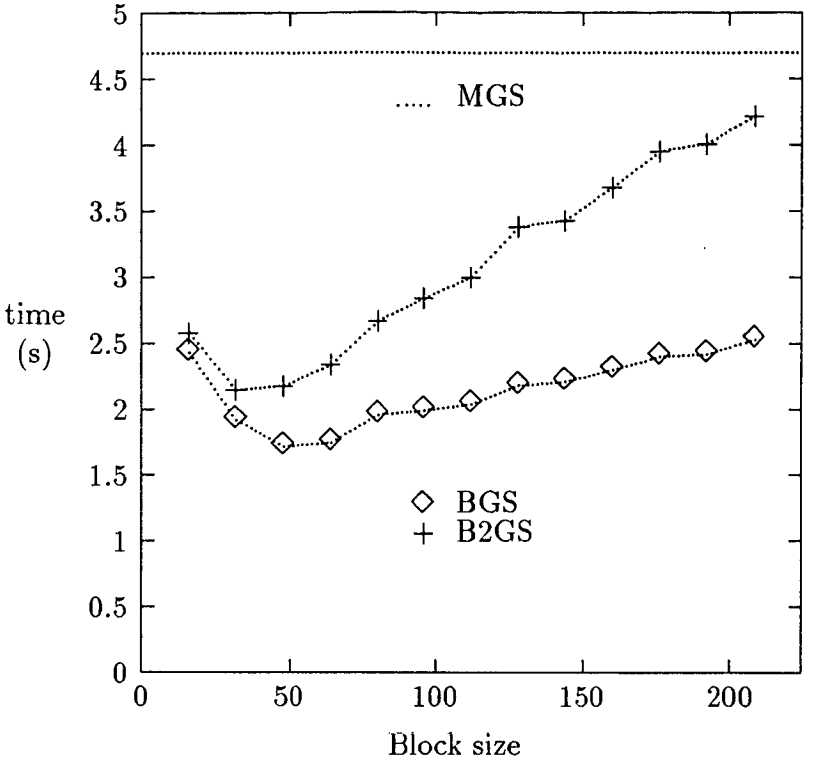
In this subsection, experimental results obtained on a CRAY-2 (1CPU) and an ALLIANT FX80 (8 processors) are presented; they support the performance analysis results by exhibiting trends as predicted and showing that B2GS offers comparable performance with BGS. The choice of these machines was motivated that they had both a hierarchical memory systems with three levels: vector registers (8 vector registers of 64 elements for the CRAY-2, 8 vector registers of 32 elements for the ALLIANT), an intermediate memory level (a private local memory of 16Kwords for the CRAY-2, a shared cache of 64 Kwords for the ALLIANT), and finally, the main memory.

6.4.1 Various block sizes

In this experiment, (BGS) and (B2GS) were run on a 1024×512 matrix with various block sizes (machine used: CRAY-2, one CPU). The timings obtained are presented in Figure 1, where the result for the classical MGS is plotted as a reference point. As predicted by the performance analysis, we clearly see two conflicting trends: at the beginning, increasing

the block size decreases the total execution time (the benefit from minimizing the load is bigger than the extra arithmetic work), then keeping increasing the block size reverses the situation: the penalty due to the extra operations is overwhelming, resulting in an increase of the execution time.

Figure 1: Running times for (MGS) (BGS) and (B2GS), rectangular matrix (1024x512), (CRAY-2 1CPU)



In the case of BGS, the minimal execution $1.72s$ is obtained for a block size of 48 while for B2GS, the minimal execution time $2.15s$ corresponds to a block size of 32. The extra cost of B2GS is 25%. A part of this penalty is due to the fact that our implementation of B2GS on the CRAY2 did not keep the block of columns in the local memory between two successive orthonormalizations (this requires a slight modification of the performance analysis); this implies that this implementation B2GS does not only increase the number of operations but also the number of loads. Such a characteristic explains also why B2GS times are ramping up much faster for large block sizes than BGS. However B2GS still achieves a speedup of over 2 when compared to the standard MGS.

6.4.2 Various matrix sizes

In these experiments, (MGS), (BGS) and (B2GS) were run for different matrix shapes and sizes on an ALLIANT FX80. The results are presented in Figure 2 (square matrix $n \times n$), Figure 3 (rectangular matrix $1024 \times n$) and Figure 4 (rectangular matrix $2048 \times n$). In all these figures, the y-axis correspond to the normalized Megaflops which is obtained by dividing for the three codes the same number of floating point operations (corresponding

to MGS) by the timing. This allows a fair comparison between the three algorithms: essentially, if the normalized Megaflop rate is 3 times higher for BGS than for MGS, this means that BGS execution time is 3 times smaller.

The experiments were performed by surrounding the code to be measured by a repetition loop in order to reduce the impact of the clock accuracy. This has an adverse effect for the small matrix sizes which entirely fit in cache; in such cases, the first iteration will load the matrix in the cache then all the subsequent operations will be performed with the data entirely resident in cache. This explains the strange shape of the curves for MGS, where the performance first increases then decreases when the data do not fit any more in cache. It should be noted that BGS and B2GS are not affected by such a phenomenom due to the fact that their structure in blocks allows them to keep most of their references to cache.

The block sizes were chosen according to the following rules:

- Square matrix, $n \times n$, (Figure 2):

$$\begin{array}{ll} \text{BGS}: & \text{Block size} = 16 \text{ for } 32 \leq n \leq 96 \\ & \text{Block size} = 32 \text{ for } 96 < n \leq 1024 \\ \text{B2GS}: & \text{Block size} = 16 \text{ for } 32 \leq n \leq 160 \\ & \text{Block size} = 32 \text{ for } 160 < n \leq 1024 \end{array}$$

- Rectangular matrix, $1024 \times n$, (Figure 3):

$$\begin{array}{ll} \text{BGS}: & \text{Block size} = 16 \text{ for } 32 \leq n \leq 64 \\ & \text{Block size} = 32 \text{ for } 64 < n \leq 1024 \\ \text{B2GS}: & \text{Block size} = 16 \text{ for } 32 \leq n \leq 128 \\ & \text{Block size} = 32 \text{ for } 128 < n \leq 1024 \end{array}$$

- Rectangular matrix, $2048 \times n$, (Figure 4):

$$\begin{array}{ll} \text{BGS}: & \text{Block size} = 16 \text{ for } 32 \leq n \leq 64 \\ & \text{Block size} = 32 \text{ for } 64 < n \leq 512 \\ \text{B2GS}: & \text{Block size} = 16 \text{ for } 32 \leq n \leq 160 \\ & \text{Block size} = 32 \text{ for } 128 < n \leq 512 \end{array}$$

This choice of the block sizes were done accordingly to experimental results.

The main conclusion of the experimental results is that B2GS costs between 12 and 25 % in terms of performance, compared with BGS; however, the speedups over classical MGS are still impressive, over 3 for large matrices. The only negative point is small size problems, where B2GS is not competitive but it should be noted that BGS performance is also severly affected in such cases. However, for such small problems, the best choice seems to be classical MGS, the reason being that in such cases, the whole data set is close to fit into the cache and therefore does not require specific block algorithms.

Figure 2: Normalized Megaflops for (MGS), (BGS) and (B2GS), square matrix (nxn), (ALLIANT FX80)

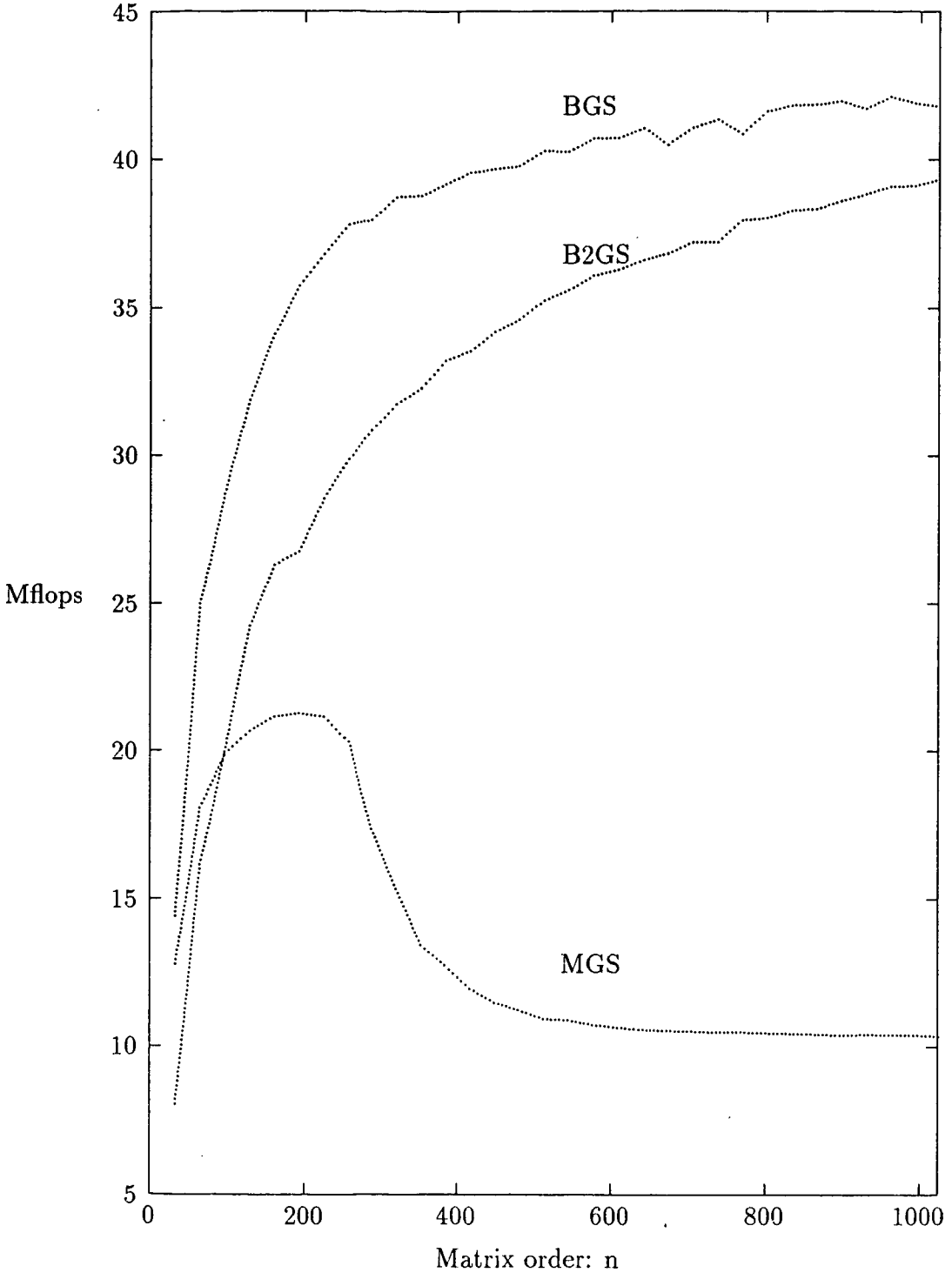


Figure 3: Normalized Megaflops for (MGS), (BGS) and (B2GS), rectangular matrix (1024xn), (ALLIANT FX80)

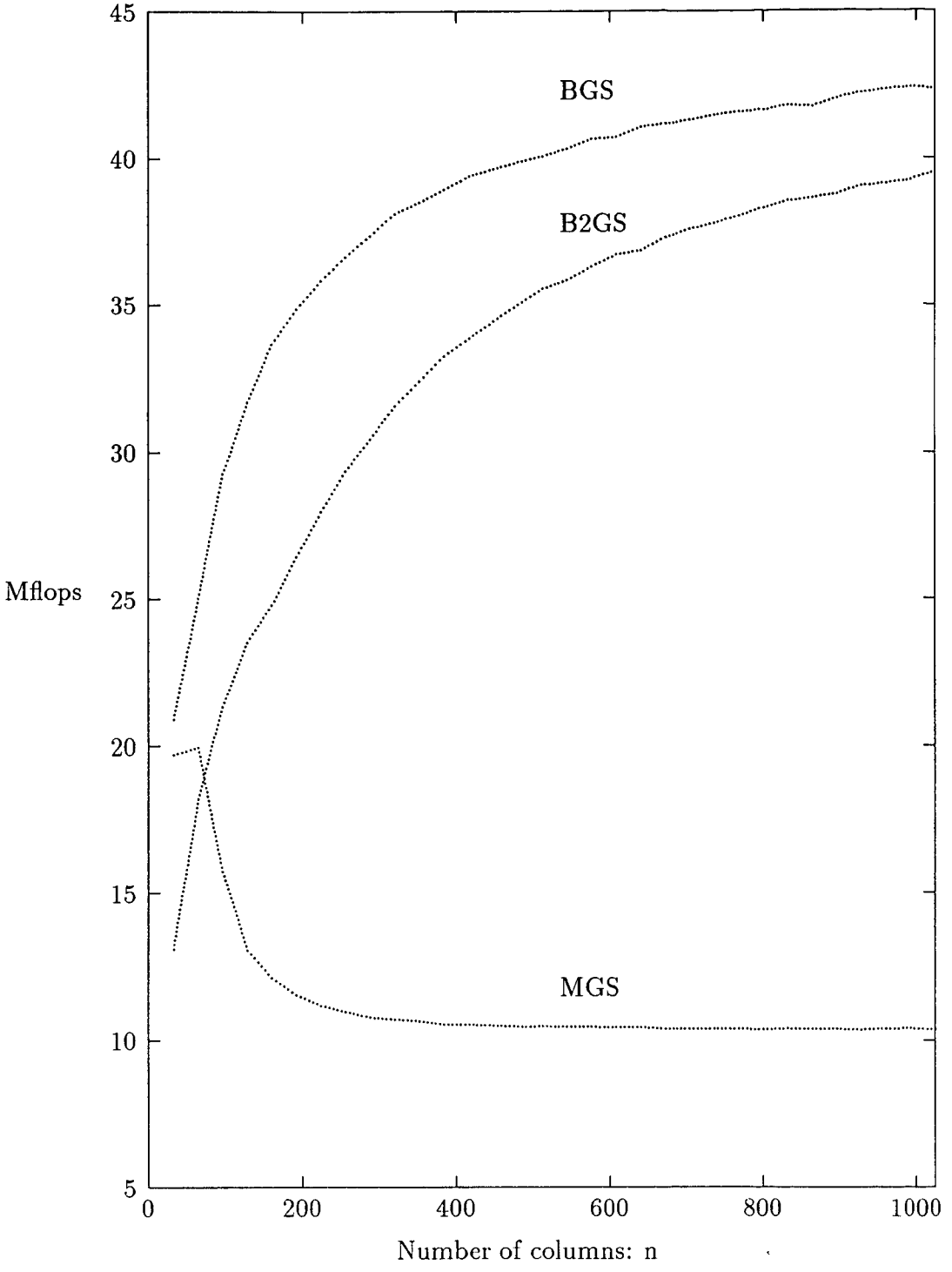
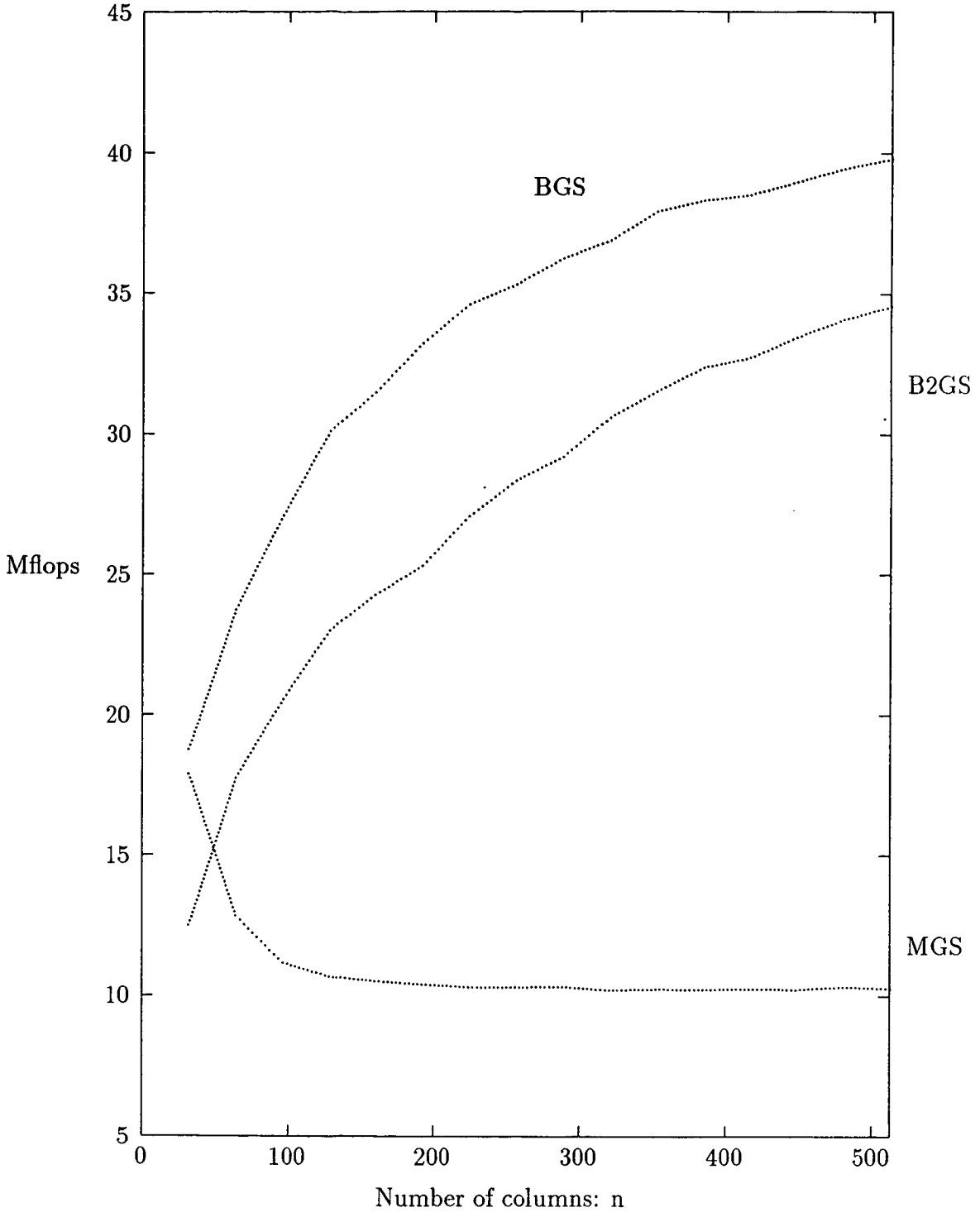


Figure 4: Normalized Megaflops for (MGS), (BGS) and (B2GS), rectangular matrix (2048xn), (ALLIANT FX80)



7 Conclusion

In this paper, it has been proven that by using a reorthogonalization procedure on blocks at an additional cost of low order, the algorithm (BGS) numerically behaves very closely to the (MGS) algorithm. Another alternative consists in replacing for the reorthonormalization process (MGS) by an other orthogonalization procedure such as the one based on polar decomposition. Since a block $\bar{Q}_\beta$ is nearly orthonormal, its orthogonalization may be performed in the following way [5]:

$$\begin{array}{lcll} \bar{D} & = & \bar{Q}_\beta^t \bar{Q}_\beta - I & np^2 \text{ flops (not considering symmetry)} \\ T & = & (I + \bar{D})^{-1/2} & O(p^3) \text{ flops} \\ \bar{Q}_\beta^c & = & \bar{Q}_\beta T & np^2 \text{flops} \end{array}$$

With such a choice, using reorthonormalization costs twice as much as using (MGS), but only involves matrix multiplication. The use of such primitives (resulting in speedup over 2 compared to Blas1 or Blas2 primitives) may in fact offset the additional price in number of operations, and constitute an interesting alternative.

References

[1] Bjorck, A., *Solving linear least squares problems by Gram-Schmidt orthogonalization* BIT, 1967, pp. 1-21.

[2] Gallivan, K., Jalby, W., and Meier, U., *The use of BLAS3 in linear algebra on a parallel processor with a hierarchical memory*, SIAM J. Sci. Stat. Comput., Vol 8, No 6, 1987, pp. 1079-1084.

[3] Gallivan, K., Jalby, W., Meier, U., and Sameh, A., *Impact of hierarchical memory systems on linear algebra algorithm design*, Intl. J. Supercomputer Appl., Vol 2, No 1, 1988, pp. 12-48.

[4] Golub, G., and Van Loan, C., *Matrix computations* The Johns Hopkins University Press, 1983, Baltimore.

[5] Philippe, B., *An algorithm to improve nearly orthonormal sets of vectors on a vector processor* SIAM Jour. Alg. Disc. Meth., Vol 8, No 3, 1987

Liste des publications internes 1990

PI 508 **RAY TRACING ON DISTRIBUTED MEMORY PARALLEL COMPUTERS: STRATEGIES FOR DISTRIBUTING COMPUTATIONS AND DATA.**
Didier BADOUEL, Kadi BOUATOUCH, Thierry PRIOL
Janvier 1990, 16 Pages.

PI 509 **STABILITY ANALYSIS AND IMPROVEMENT OF THE BLOCK GRAM-SCHMIDT ALGORITHM.**
William JALBY, Bernard PHILIPPE
Janvier 1990, 24 Pages.

PI 510 **TESTING FOR THE UNBOUNDEDNESS OF FIFO CHANNELS IN PROGRAMS.**
Thierry JERON
Janvier 1990, 30 Pages.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique