Enhanced Gram-Schmidt Process for Improving the Stability in Signal and Image Processing

Mario Mastriani and Marcelo Naiouf

Abstract—The Gram-Schmidt Process (GSP) is used to convert a non-orthogonal basis (a set of linearly independent vectors) into an orthonormal basis (a set of orthogonal, unit-length vectors). The process consists of taking each vector and then subtracting the elements in common with the previous vectors. This paper introduces an Enhanced version of the Gram-Schmidt Process (EGSP) with inverse, which is useful for signal and image processing applications.

Keywords—Digital filters, digital signal and image processing, Gram-Schmidt Process, orthonormalization.

I. INTRODUCTION

ORTHOGONALIZATION processes play a key role in many iterative methods used in Correlation Matrix memory [1], array signal processing [2], the Kalman Filtering problem [3], datamining and bioinformatics [4], among others [5, 6], with different implementation possibilities [3, 7-9], however, the issues involved in the use of Very-Large-Scale Integration (VLSI) technology to implement an adaptive version of the GSP, based on the escalator structure, are discussed in [10], while alternative versions for optimal filtering and control problems without using GSP are discussed in [11-15]. Besides, it has a very important field of applications in communica-tions, see [16-25]. Returning to the situation at hand, the EGSP is useful to perform a stable orthonormalization with inverse process, in opposition to previous versions that achieve it thanks to impractical or unstable algorithmic methods [26, 27], or being stable doesn’t have inverse, such as Modified GSP [28]. Finally, a good orthonormalization algorithm with inver-se is essential for filtering and compression in Digital Signal and Image Processing.

The original Gram-Schmidt Process (GSP) is outlined in Section II. Inverse of GSP is outlined in Section III. Enhanced Gram-Schmidt Process (EGSP) is outlined in Section IV. Performance proof is outline in Section V. In Section VI, we discuss briefly the employed routines. Finally, Section VI provides a conclusion of the paper.
function \[u,r\] = gsp(v)
\[\begin{array}{l}
u(:,1) = v(:,1);
i = 1;
\end{array}\]
\[\begin{array}{l}
[M,N] = size(v);
for n = 2:N
  \begin{array}{l}
  acu = 0;
  for m = 1:n-1
    \begin{array}{l}
    r(i) = ((v(:,n))' * u(:,m)) / ((u(:,m))' * u(:,m));
    acu = acu + r(i) * u(:,m);
  \end{array}
  \end{array}
  i = i+1;
\end{array}\]
\[\begin{array}{l}
  u(:,n) = v(:,n) - acu;
end\]

III. INVERSE OF GSP (IGSP)

An algorithmic version of the IGSP is indispensable for multiple applications [1-31], therefore, an original process is exposed below. However, it is important to mention that this version is unstable under certain conditions [28].

A. An Algebraic Version for the Discrete IGSP

Based on prior considerations \(v_1 = u_1\), and the remaining \(v_n\) are defined by
\[v_n = U_n + \sum_{m=1}^{n-1} r_{nm} u_m, \quad n = 2, 3, \ldots, N\] (4)

considering Eq. (2).

B. Algorithmic Version

The algorithmic version of IGSP is based on Eq. (4) and (2), as shown in the Fig. 2.

The algorithm as a function would be:
\[\begin{array}{l}
function v = igsp(u,r)
\end{array}\]
\[\begin{array}{l}
v(:,1) = u(:,1);
i = 1;
\end{array}\]
\[\begin{array}{l}
[M,N] = size(u);
for n = 2:N
  \begin{array}{l}
  acu = 0;
  for m = 1:n-1
    \begin{array}{l}
    r(i) = ((v(:,n))' * u(:,m)) / ((u(:,m))' * u(:,m));
    acu = acu + r(i) * u(:,m);
  \end{array}
  \end{array}
  i = i+1;
\end{array}\]
\[\begin{array}{l}
  v(:,n) = v(:,n) - acu;
end\]

Finally, a modified version of the traditional GSP exists [28], and it is stable, but it doesn’t have inverse because it is an in-situ method, i.e., it is a destructive algorithm [31]. This is the reason for which becomes indispensable a new algorithm that being stable has inverse.

IV. ENHANCED GRAM-SCHMIDT PROCESS (EGSP)

A. Algebraic Version of Improvement of the Stability

The developed algorithm is the very traditional version of GSP (well known for its bad numerical properties, see [28]), modified versions of the same one exist, called Modified GSP (MGSP), see [31], but unfortunately, they don’t have inverse, because, they are in-situ algorithm, i.e., they are applicable on the same set of key vectors, i.e., they constitute destructive methods. On the other hand, the unstability happens when the denominators of the \(r\) elements are close to zero. Therefore, to assure the stability it is necessary to apply the following procedure on the input (key) vectors \(v_1, v_2, \ldots, v_N\):

1. \(v_1^\top v_1 = \min (v_1^\top v_1, v_2^\top v_2, \ldots, v_N^\top v_N)\)

2. \((v_1 + \text{bias I})^\top (v_1 + \text{bias I}) = 1\), where \(I = [1 \ 1 \ \ldots \ 1]^\top\) being a \((N \times 1)\) vector

3. \((N) \text{bias}^2 + (2 v_1^\top I) \text{bias} + (-1 + v_1^\top v_1) = 0\)

In such a way that the minimum denominator of the elements of \(r\) are equal to one.

Finally,
\[\text{bias} = -(v_1^\top I / N) + [(v_1^\top I / N)^2 - (v_1^\top v_1 / N) + (1 / N)]^{1/2}\]

B. Algorithmic Version of Improvement of the Stability

The algorithm as a function is
\[\begin{array}{l}
function [v,bias] = is(v)
\end{array}\]
\[\begin{array}{l}
[M,N] = size(v);
for n = 1:N
  \begin{array}{l}
  w(n) = (v(:,n))' * v(:,n);
  end
\end{array}\]
\[\begin{array}{l}
[wmin,n] = min(w);
if wmin < 1,
  \begin{array}{l}
  bias = -(sum(v(:,n)) / N - (sum(v(:,n)) / N)^2 - (w(n) / N + (1 / N))^{1/2});
  end
else
  bias = 0;
end\]

Fig. 1 Gram-Schmidt Process (GSP).

Fig. 2 Inverse Gram-Schmidt Process (IGSP).
The improvement of the stability is based on previous development, as shown in Fig. 1. Therefore, the final algorithmic version of GSP improved in stability, i.e., EGSP, can be observed in Fig. 4.

The following programs represent the complete process

\[
\begin{align*}
\text{function } & \quad \text{v} = \text{is(v);} \quad \text{bias, u and r are transmitted} \\
\text{or} & \quad \text{v} = \text{egsp(v);} \quad \text{EGSP as a function} \\
\text{function } & \quad \text{v} = \text{si(v,bias);} \quad \text{bias, u and r are transmitted} \\
\text{or} & \quad \text{v} = \text{iegsp(u,r,bias);} \quad \text{IGSP as a function}
\end{align*}
\]

Considering the improvement of the stability, the IEGSP algorithm will be as shown in Fig. 5

\[
\text{function } v = \text{si(v,bias);} \quad [M,N] = \text{size(v);} \quad v = v - \text{bias * ones(M,N);} \quad \text{see Fig. 3}
\]

\[
\text{or}
\]

\[
\text{function } v = \text{iegsp(u,r,bias);} \quad v(:,1) = u(:,1); \quad i = 1; \\
\text{for } n = 2:N \quad acu = 0; \\
\text{for } m = 1:n-1 \\
r(i) = ((v(:,n)'*u(:,m))/(u(:,m)'*u(:,m)) + acu + r(i)*u(:,m); \\
i = i+1; \\
\text{end} \\
u(:,n) = v(:,n) - acu; \\
\text{end}
\]

V. PERFORMANCE PROOF

A. Output-Input Size Rate (OISR) Without Improvement of the Stability

The OISR for GSP (i.e., GSP-OISR) is:

\[
\text{GSP-OISR} = \frac{\text{size}(u) + \text{size}(r)}{\text{size}(v)}
\]
Replacing in order, the corresponding dimensions, in Eq. (5)
\[
GSP-OISR = \frac{NM + N}{NM} \tag{6}
\]
considering Eq. (3)
\[
GSP-OISR = \frac{N(N-1)}{NM} + \frac{2}{NM} + 1 \tag{7}
\]
and simplifying
\[
GSP-OISR = \frac{2M + N - 1}{2M} \tag{8}
\]
However, for practical considerations \( M \gg N \) \([1, 5, 8-15]\), then \( GSP-OISR \approx 1 \), being \( M \) the input space dimension, and \( N \) the number of vectors of such space. Similarly, and for identical considerations, the IGSP-OISR \( \approx 1 \), too.

B. Output-Input Size Rate (OISR) with Improvement of the Stability

The OISR for GSP (i.e., GSP-OISR) is:
\[
GSP-OISR = \frac{\text{size}(\text{u}) + \text{size}(\text{r}) + 1}{\text{size}(\text{v})} \tag{9}
\]
Replacing in order, the corresponding dimensions, in Eq. (9)
\[
GSP-OISR = \frac{NM + N + 1}{NM} \tag{10}
\]
considering Eq. (3)
\[
GSP-OISR = \frac{N(N-1)}{NM} + \frac{2}{NM} + 1 \tag{11}
\]
and simplifying
\[
GSP-OISR = \frac{2NM + N^2 - N - 2}{2NM} \tag{12}
\]
For identical considerations to the previous case GSP-OISR \( \approx 1 \) and IGSP-OISR \( \approx 1 \), too.

VI. EMPLOYED ROUTINES

The used routines are the functions: egsp() and iegsp(), besides the following ones

```matlab
M = input('M = '); N = input('N = '); v = rand(M,N); [u,r,bias] = egsp(v); aux = po(u)'; % proof of orthogonality v = iegsp(u,r,bias); % and
function aux = po(u)
i = 1;
[M,N] = size(u);
for a = 1:N-1
    for b = a+1:N
        aux(i) = u(:,a)'*u(:,b);
i = i+1;
    end
end
```

The second routine proves the orthogonalization of the vectors caused by the EGSP, while the first use to all the other routines is to verify the operation of the IEGSP.

VII. CONCLUSIONS

An algorithm for a discrete version of the IGSP is performed. The GSP generates a matrix \( \text{u} \) and a vector \( \text{r} \) which can be stored or transmitted starting from the matrix \( \text{v} \), while the IGSP will reconstruct to the original matrix \( \text{u} \) based on the matrix \( \text{v} \) and the vector \( \text{r} \). Since for practical considerations \([1, 5, 8-15]\) the size of the vector \( \text{r} \) is insignificant compared with the size of the matrices \( \text{v} \) and \( \text{u} \), then GSP-OISR and IGSP-OISR are approximately equal to one, see Eq. (8). The computer simulations show null error when applying GSP followed by IGSP regarding to the matrix \( \text{v} \).

REFERENCES


Marcelo Naiouf received his PhD degree from the Universidad Nacional de La Plata (UNLP) in 2004. He is currently a researcher and chair professor in the Instituto de Investigación en Informática LIDI (IIBERATIC) of the Computer Science School in UNLP. His main research interests include parallel and distributed systems, algorithms, performance analysis and load planning. Dr. Naiouf is project director 11/F011 “Parallel and distributed processing. Fundamentals and applications in intelligent systems and image processing and video” within the Incentive Program, and publications on topics related records in journals and national and international congresses. Among his most recent research background may be mentioned: - Professor categorized II in the incentive program of the Ministry of Education. -Director of the Doctorate in Computer Science from the Faculty of Informatics, UNLP. - Academic Coordinator of the Iberoamerican Network Engineering and Information Technology (IBERO TIC), consisting of universities in Spain, Argentina, Mexico, Colombia, Cuba, Chile, Paraguay, Peru and Uruguay to the Organization of Iberoamerican States (OEI) for mobility of doctoral students and faculty (2009-2013). - Project Participant "CYTED Grid: Grid technology as an engine of regional development", as part of a network of 16 universities in Europe and America. - Project Participant PAV 076 "Intelligent Systems in support of production processes" accredited by National Agency for Scientific and Technological Promotion.