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.

Mario Mastriani is with the Laboratorio de Imágenes y Señales (LIS), Secretaría de Investigación y Desarrollo, Universidad Nacional de Tres de Febrero, 910 Florida St., (C1005AAT), 6th Floor, Room B, Buenos Aires, Argentina (email: mmastriani@untref.edu.ar).

Marcelo Naiouf is with the Instituto de Investigación en Informática (III-LIDI), Facultad de Informática, Universidad Nacional de La Plata, 50 and 120 St. (B1900ASJ), 2nd. Floor, La Plata, Argentina (email: mnaiouf@lidi.info.unlp.edu.ar).

II. Gram-Schmidt Process (GSP)

A. Algebraic Version

Given a set of key vectors that are linearly independent but nonorthonormal, it is possible to use a preprocessor to transform them into an orthonormal set; the processor is designed to perform a Gram-Schmidt orthogonalization on the key vectors prior to association [29, 30]. This way to transform is linear, maintaining a one-to-one correspondence between the input (key) vectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, … , $\boldsymbol{v}_N$, and the resulting orthonormal vectors $\boldsymbol{u}_1$, $\boldsymbol{u}_2$, … , $\boldsymbol{u}_N$, as indicated in:

$$\{ \boldsymbol{v}_1 , \boldsymbol{v}_2 , \dots , \boldsymbol{v}_N \} \Leftrightarrow \{ \boldsymbol{u}_1 , \boldsymbol{u}_2 , \dots , \boldsymbol{u}_N \}$$

where $\boldsymbol{u}_1 = \boldsymbol{v}_1$, and the remaining $\boldsymbol{u}_n$ are defined by [30]

$$u_n = v_n - \sum_{m=1}^{n-1} r_{m,n}\, u_m , \quad n = 2, 3, \dots , N \tag{1}$$

with

$$r_{m,n} = \frac{u_m^T v_n}{u_m^T u_m} \tag{2}$$

where $(.)^T$ means transpose of (.). The orthogonality of key vectors may also be approached using statistical considerations. Specifically, if the input space dimension M is large and the key vectors have statistically independent elements, then they will be close to orthogonality with respect to each other [1]. However, the number of coefficients $\boldsymbol{r}_{m,n}$ is

$$\boldsymbol{N_r} = \frac{N(N-1)}{2} \tag{3}$$

which is independent of the key vectors dimension M, being N the number of vectors.

B. Algorithmic Version

The algorithmic version of GSP is based on Eq. (1) and (2), as shown in Fig. 1. A matrix $\mathbf{v}$ (M x N) would be built from the base of the input (key) vectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, … , $\boldsymbol{v}_N$, and its columns would be the same vectors. Similarly, with the resulting orthonormal vectors $\boldsymbol{u}_1$, $\boldsymbol{u}_2$, … , $\boldsymbol{u}_N$, a matrix $\mathbf{u}$ (M x N) is built whose columns are these vectors. The process will also build a vector $\mathbf{r}$ (N_r x 1), in terms of Eq. (1), (2) and (3). The algorithm as a function is

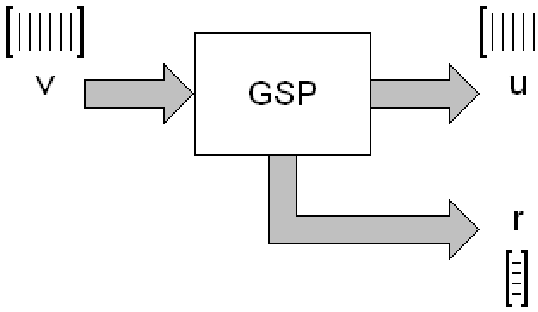

Fig. 1 Gram-Schmidt Process (GSP).

```
function [u,r] = gsp(v)
u(:,1) = v(:,1);
i = 1;
[M,N] = size(v);
for n = 2:N
  acu = 0;
  for m = 1:n-1
    r(i) = ((v(:,n)')*u(:,m))/((u(:,m)')*u(:,m));
    acu = acu + r(i)*u(:,m);
    i = i+1;
  end
  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 $\boldsymbol{v}_1 = \boldsymbol{u}_1$, and the remaining $\boldsymbol{v}_n$ are defined by

$$v_n = u_n + \sum_{m=1}^{n-1} r_{m,n}\, u_m , \quad n = 2, 3, \dots , N \qquad (4)$$

considering Eq. (2).

B. Algorithmic Version

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

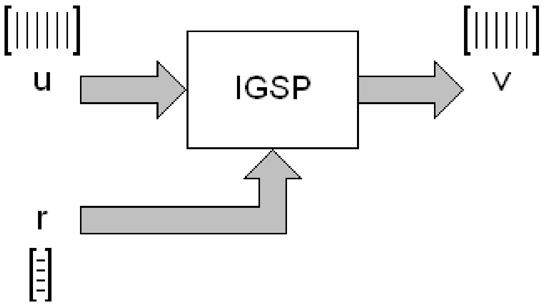

Fig. 2 Inverse Gram-Schmidt Process (IGSP).

The algorithm as a function would be:

```
function v = igsp(u,r)
v(:,1) = u(:,1);
i = 1;
[M,N] = size(u);
for n = 2:N
  acu = 0;
  for m = 1:n-1
    acu = acu + r(i)*u(:,m);
    i = i+1;
  end
  v(:,n) = u(:,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 $\boldsymbol{v}_1 , \boldsymbol{v}_2 , \dots , \boldsymbol{v}_N$:

1. $\boldsymbol{v}_i^{T} \boldsymbol{v}_i = \min (\boldsymbol{v}_1^{T} \boldsymbol{v}_1 , \boldsymbol{v}_2^{T} \boldsymbol{v}_2 , \dots. , \boldsymbol{v}_N^{T} \boldsymbol{v}_N)$

2. $(\boldsymbol{v}_i + \textbf{bias I})^{T} (\boldsymbol{v}_i + \textbf{bias I}) = 1$, where $\mathbf{I} = [\ 1\ 1\ \dots.\ 1\ ]^{T}$ being a (N x 1) vector

3. $(N)\ \textbf{bias}^2 + (2\, \boldsymbol{v}_i^{T} \mathbf{I})\ \textbf{bias} + (-1 + \boldsymbol{v}_i^{T} \boldsymbol{v}_{i)} = 0$

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

Finally,

$$\textbf{bias} = -(\boldsymbol{v}_i^{T} \mathbf{I} / N) + [(\boldsymbol{v}_i^{T} \mathbf{I} / N)^{2} -(\boldsymbol{v}_i^{T} \boldsymbol{v}_i / N) + (1 / N)]^{1/2}$$

B. Algorithmic Version of Improvement of the Stability

The algorithm as a function is

```
function [v,bias] = is(v)
[M,N] = size(v);
for n = 1:N
  w(n) = (v(:,n)')*v(:,n);
end
[minw,n] = min(w);
if minw < 1,
  bias = -(sum(v(:,n))/N)-(((sum(v(:,n))/N)^2)-
           (w(n)/N)+(1/N))^(1/2);
else
  bias = 0;
```

```
end
v = v + bias * ones(M,N); % see Fig.1
```

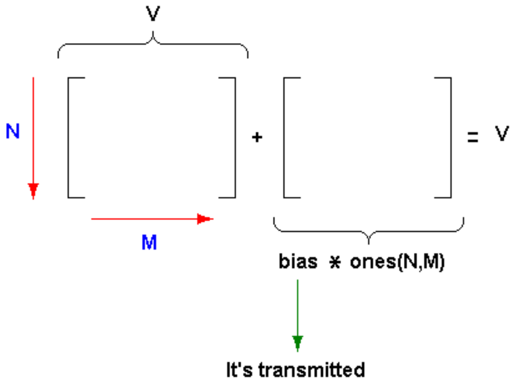

Fig. 3 Bias inclusion for the redefinition of matrix V.

C.Algorithmic Version of EGSP

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.

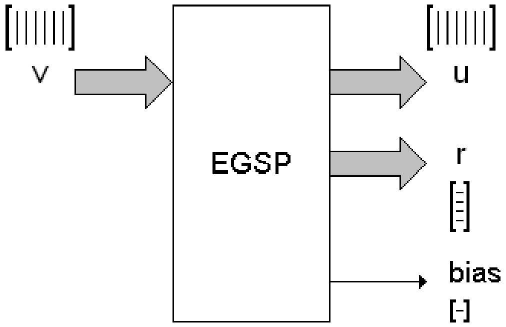

Fig. 4 GSP improved in stability (EGSP).

The following programs represent the complete process

```
% input of v
[v,bias] = is(v);
[u,r] = gsp(v);
% bias, u and r are transmitted

% or

% input of v
[u,r,bias] = egsp(v); % EGSP as a function
% bias, u and r are transmitted

% where

function [u,r,bias] = egsp(v)
[M,N] = size(v);
for n = 1:N
  w(n) = (v(:,n)')*v(:,n);
end
[minw,n] = min(w);
if minw < 1,
  bias = -(sum(v(:,n))/N)-(((sum(v(:,n))/N)^2)
         -(w(n)/N)+(1/N))^(1/2);
else
  bias = 0;
end
v = v + bias * ones(M,N);
u(:,1) = v(:,1);
i = 1;
for n = 2:N
  acu = 0;
  for m = 1:n-1
    r(i) = ((v(:,n)')*u(:,m))/((u(:,m)')*u(:,m));
    acu = acu + r(i)*u(:,m);
    i = i+1;
  end
  u(:,n) = v(:,n) - acu;
end
```

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

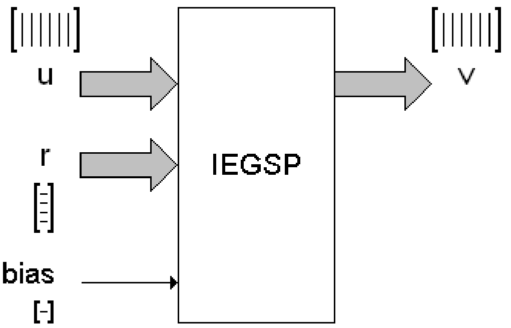

Fig. 5 IEGSP.

and the following programs represents the complete inverse process

```
% bias, u and r are recived
v = igsp(u,r);
v = si(v,bias);

% where

function v = si(v,bias)
[M,N] = size(v) ;
v = v - bias * ones(M,N); % see Fig.3

% or

function v = iegsp(u,r,bias)
v(:,1) = u(:,1);
i = 1;
[M,N] = size(u);
for n = 2:N
  acu = 0;
  for m = 1:n-1
    acu = acu + r(i)*u(:,m);
    i = i+1;
  end
  v(:,n) = u(:,n) + acu;
end
v = v - bias * ones(M,N);
```

V.Performance Proof

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

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

$$GSP-OISR = \frac{size(\mathbf{u})+size(\mathbf{r})}{size(\mathbf{v})} \qquad (5)$$

Replacing in order, the corresponding dimensions, in Eq. (5)

$$GSP-OISR = \frac{NM+N_r}{NM} \qquad (6)$$

considering Eq. (3)

$$GSP-OISR = \frac{NM+\frac{N(N-1)}{2}}{NM} \qquad (7)$$

and simplifying

$$GSP-OISR = \frac{2M+N-1}{2M} \qquad (8)$$

However, for practical considerations $M >> N$ [1, 5, 8-15], then GSP-OISR ≅ 1, being M the input space dimension, and N the number of vectors of such space. Similarly, and for identical considerations, the IGSP-OISR ≅ 1, too.

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

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

$$GSP-OISR = \frac{size(\mathbf{u})+size(\mathbf{r})+1}{size(\mathbf{v})} \qquad (9)$$

Replacing in order, the corresponding dimensions, in Eq. (9)

$$GSP-OISR = \frac{NM+N_r+1}{NM} \qquad (10)$$

considering Eq. (3)

$$GSP-OISR = \frac{NM+\frac{N(N-1)}{2}+1}{NM} \qquad (11)$$

and simplifying

$$GSP-OISR = \frac{2NM+N^2-N-2}{2NM} \qquad (12)$$

For identical considerations to the previous case GSP-OISR ≅ 1 and IGSP-OISR ≅ 1, too.

VI. Employed Routines

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

```
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 **u** and a vector **r** which can be stored or transmitted starting from the matrix **v**, while the IGSP will reconstruct to the original matrix **u** based on the matrix **v** and the vector **r**. Since for practical considerations [1, 5, 8-15] the size of the vector **r** is insignificant compared with the size of the matrices **v** and **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 **v**.

References

[1] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New York: Macmillan, 1994.

[2] S. Haykin, *Modern Filters*. New York: Macmillan, 1990.

[3] S. Haykin, *Adaptive Filter Theory*. New Jersey: Prentice-Hall, 1991.

[4] H Lodhi and Y. Guo, "Gram-Schmidt kernels applied to structure activity analysis for drug design", in Proc. of 2nd. Workshop on Datamining in Bioinformatics, Edmonton, Alberta, Canada, July 23rd. 2002, pp. 37-42.

[5] R. A. Mozingo and T. W. Miller, *Introduction to Adaptive Arrays*. New York: Wiley, 1980.

[6] R. T. Compton, *Adaptive Antennas: Concepts and Performance*. New Jersey: Prentice-Hall, 1988.

[7] G. W. Stewart, *Introduction to Matrix Computations*. Orlando: Academic Press, 1973.

[8] J. Makhoul, "A class of all-zero lattice digital filters: properties and applications," IEEE Trans. Acoust. Speech and Signal Process., vol. ASSP-26, pp.304-314, 1978.

[9] N. Ahmed and D. H. Youn, "On a realization and related algorithm for adaptive prediction," IEEE Trans. Acoust. Speech and Signal Process., vol. ASSP-28, pp.493-4974, 1980.

[10] K. A. Gallivan and C. E. Leiserson, "High-performance architectures for adaptive filtering based on the Gram-Schmidt algorithm," SPIE, Real Time Signal Process. VII, vol.495, pp.30-36, 1984.

[11] M. Mastriani, "Self-Restorable Stochastic Neuro-Estimation using Forward-Propagation Training Algorithm," in *Proc. of INNS*, Portland, OR, July 1993, vol. 1, pp. 404-411.

[12] M. Mastriani, "Self-Restorable Stochastic Neurocontrol using Back-Propagation and Forward-Propagation Training Algorithms," in *Proc. of ICSPAT*, Santa Clara, CA, Sept. 1993, vol. 2, pp.1058-1071.

[13] M. Mastriani, “Pattern Recognition Using a Faster New Algorithm for Training Feed-Forward Neural Networks,” in *Proc. of INNS*, San Diego, CA, June 1994, vol. 3, pp. 601-606.

[14] M. Mastriani, “Predictor of Linear Output,” in *Proc. IEEE Int. Symposium on Industrial Electronics*, Santiago, Chile, May 1994, pp.269-274.

[15] M. Mastriani, “Predictor-Corrector Neural Network for doing Technical Analysis in the Capital Market,” in *Proc. AAAI International Symposium on Artificial Intelligence*, Monterrey, México, October 1994, pp. 49-58.

[16] N. Al-Dhahir and J. M. Cioffi, "Efficiently computed reduced-parameter input-aided MMSE equalizers for ML detection: A unified approach," IEEE Trans. on Info. Theory, vol. 42, pp. 903-915, May 1996.

[17] P. J. W. Melsa, R. C. Younce, and C. E. Rhors, "Impulse response shortening for discrete multitone transceivers," IEEE Trans. on Communications, vol. 44, pp. 1662-1672, Dec. 1996.

[18] N. Al-Dhahir and J. M. Cioffi, "Optimum finite-length equalization for multicarrier transceivers," IEEE Trans. on Communications, vol. 44, pp. 56-63, Jan. 1996.

[19] B. Lu , L. D. Clark, G. Arslan, and B. L. Evans, "Divide-and-conquer and matrix pencil methods for discrete multitone equalization," IEEE Trans. on Signal Processing, in revision.

[20] G. Arslan , B. L. Evans, and S. Kiaei, "Equalization for Discrete Multitone Receivers To Maximize Bit Rate," IEEE Trans. on Signal Processing, vol. 49, no. 12, pp. 3123-3135, Dec. 2001.

[21] B. Farhang-Boroujeny and Ming Ding, "Design Methods for Time Domain Equalizer in DMT Transceivers", IEEE Trans. on Communications, vol. 49, pp. 554-562, March 2001.

[22] J. M. Cioffi, A Multicarrier Primer. Amati Communication Corporation and Stanford University, T1E1.4/97-157, Nov. 1991.

[23] M. Ding, A. J. Redfern, and B. L. Evans, "A Dual-path TEQ Structure For DMT-ADSL Systems", Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Proc., May 13-17, 2002, Orlando, FL, accepted for publication.

[24] K. V. Acker, G. Leus, M. Moonen, O. van de Wiel, and T. Pollet, "Per tone equalization for DMT-based systems," IEEE Trans. on Communications, vol. 49, no. 1, pp. 109-119, Jan 2001.

[25] S. Trautmann, N.J. Fliege, “A new equalizer for multitone systems without guard time” IEEE Communications Letters, Vol.6, No. 1 pp. 34 -36, Jan 2002.

[26] M. Tuma and M. Benzi, “A robust preconditioning technique for large sparse least squares problems”, in Proc. of XV. Householder Symposium, Peebles, Scotland, June 17-21, 2002. pp.8

[27] H. J. Pedersen, “Random Quantum Billiards”, Master dissertation, K0benhavns Universitet, Copenhagen, Denmark, july 1997, pp.63

[28] G. H. Golub and C. F. van Loan, *Matrix Computations*, (3rd Edition) Johns Hopkins University Press, Baltimore, 1996.

[29] G. Hadley, *Linear Algebra*. Mass.: Wesley, 1961.

[30] G. Strang, *Linear Algebra and Its Applications.* New York: Academic Press, 1980.

[31] S. J. Leon, “Linear Algebra with Applications” Maxwell MacMillan, New York, 1990.

Mario Mastriani received the B.Eng. degree in 1989 and the Ph.D. degree in 2006, both in electrical engineering from Buenos Aires University. He is Professor of Digital Signal and Image Processing in Bioengineering of Engineering College, at Buenos Aires University (UBA). Pro-fessor Mastriani is the Director of the Laboratory of Research and Development in New Technologies (LIDeNTec) of ANSES, and the Computer Engineering Career of the National University of Tres de Febrero, at Buenos Aires, Argentina. He published 42 papers. He is a currently reviewer of IEEE Transactions on Neural Networks, Signal Processing Letters, Transactions on Image Processing, Transactions on Signal Processing, Communications Letters, Transactions on Geoscience and Remote Sensing, Transactions on Medical Imaging, Transactions on Biome-dical Engineering, Transactions on Fuzzy Systems, Transactions on Multi-media; Springer-Verlag Journal of Digital Imaging, SPIE Optical Engineering Journal; and Taylor & Francis International Journal of Remote Sensing. He (M’05) became a member (M) of WASET in 2004. His areas of interest include Digital Signal Processing, Digital Image Processing, wavelets and Neural Networks.

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 (III-LIDI) of the Computer Science School in UNLP. His main research interests include parallel and distributed systems, algorithms, performance analysis and load planification. 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 (IberoTIC), 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.