2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Data Extrapolation Using Genetic Programming to Matrices Singular Values Estimation

Jose Aguilar, *Member, IEEE* and Gilberto González

***Abstract*—In mathematical models where the dimensions of the matrices are very large, the use of classical methods to compute the singular values is very time consuming and requires a lot of computational resources. In this way, it is necessary to find new faster methods to compute the singular values of a very large matrix. We present a method to estimate the singular values of a matrix based on Genetic Programming (GP). GP is an approach based on the evolutionary principles of the species. GP is used to make extrapolations of data out of sample data. The extrapolations of data are achieved by irregularity functions which approximate very well the trend of the sample data. GP produces from just simple's functions, operators and a fitness function, complex mathematical expressions that adjust smoothly to a group of points of the form (x_i,y_i). We obtain amazing mathematical formulas that follow the behaviour of the sample data. We compare our algorithm with two techniques: the linear regression and non linear regression approaches. Our results suggest that we can predict with some percentage of error the largest singular values of a matrix without computing the singular values of the whole matrix and using only some random selected columns of the matrix.**

I. Introduction

The analysis of singular values and singular value decomposition (SVD) is applied in a widely different kind of disciplines. The more classical application is in least square fitting of data [4], [11]. One particular application is also in data mining, where is used for ranking documents in very large databases. For example, latent semantic indexing (LSI) is a variant of the *vector space model* in which a low-rank approximation to the vector space representation of the database is employed. That is, we replace the original matrix by another matrix that is close as possible to the original matrix, but whose column space is only a subspace of the column space of the original matrix [2], [9]. This new matrix is composed by the largest singular values of the original matrix.

We propose an algorithm to estimate the largest singular values of matrices of large dimensions based on Genetic Programming (GP). Our algorithm can be split in two phases, one phase of acquisition of data and the other phase of extrapolation of the data. In the phase of acquisition of data we use randomised algorithms which are based in random sampling. This means that the problem is solved using randomly selected samples of data, because the data is very large to be processed by standard methods [6], [10]. In these types of algorithms the correct answer is not always achieved, but we can say that the answer is correct with some probability. The data obtained in this phase consists of points (x_i, y_i), where x_i is the number of columns sampled from the original matrix and y_i is the particular singular value corresponding to the smaller matrix formed by the sampled columns. In the phase of extrapolation of data we use the GP [1], [7] and the data received from the phase of acquisition of data. This is the phase that we are going to study in this work, because the extrapolation problem can be complex and time consuming if we like to obtain good results. Particularly, we make estimations of the singular values of the matrix of large dimensions. Our algorithm generates a mathematical expression which tries to represent the general trend of the data as accurate as possible. This mathematical expression is used to make an extrapolation to estimate the singular value corresponding to this data. That is, an extrapolation is made for every singular value after the phase of acquisition of data.

Obviously if we use a large number of samples the data will be better, but the time of computation will be higher and eventually bigger than if we use a classical method for computing the singular values. We make comparisons between the effectiveness of our algorithm based on GP and extrapolations using linear regression, and non linear regression. The comparisons are made for large and small dense and sparse matrices.

This paper is organized as follows. In section 2 we present the concept of the singular values of matrices. In section 3, we illustrate our algorithm to estimate the largest singular values of matrices of large dimensions. In section 4, we propose an efficient algorithm for solving the data extrapolation problem based on GP. In section 5, we present the linear regression and non linear regression approaches to compare our algorithm with them, and the experimental studies on different matrices (dense and sparse). Finally, some concluding remarks are given.

II. Singular Values of Matrices

The singular value decomposition (SVD) has enjoyed a long and rich history. Singular value analysis has been applied in a wide variety of problems, it has become an invaluable tool in applied mathematics and mathematical modeling [3]. Recently, it is being used in data mining applications and by search engines to rank documents in

This work was supported in part by the CDCHT-ULA grant I-E-05-25, and CDCHT-ULA grant I-820-05-02-AA.

J. Aguilar is with Universidad De Los Andes, Facultad de Ingeniería, CEMISID, Hechicera, Mérida, 5101 Venezuela (e-mail: aguilar@ula.ve).

G. Gonzalez is with Universidad De Los Andes, Facultad de Ingeniería, Departamento de Cálculo, Hechicera, Mérida, 5101 Venezuela.

0-7803-9487-9/06/$20.00/©2006 IEEE

very large databases, including the Web. The dimensions of matrices, which appear in these applications, are becoming so large that classical algorithms for computing the SVD cannot always be used. In general, the singular value decomposition is the factorization of a matrix A into the product:

$$A = U \Sigma V^T$$

of unitary matrices U and V and a diagonal matrix Σ. A formal statement of the existence theorem for the SVD and associated definitions can be found in standards texts on linear algebra.

Accurate estimates of the largest 10%-25% singular values of a matrix are useful for understanding properties of the matrix form a theoretical perspective. In general, singular values can be used, among many things, to determine:

--The 2-norm of a matrix: represents the maximum magnification that can be undergone by any vector when acted on by the matrix.

--The closest distance to any matrix with rank N, whenever the N-*th* singular value can be estimated

--A lower bound for the condition number of a matrix. The condition number of a matrix, is one of the simplest and useful measures of the sensitivity of a linear system associated with the matrix.

A. Singular Values and Information Retrieval

As mentioned earlier, the SVD is being used in some automated search and retrieval systems to rank document in very large databases, and more recently, it has been extended to retrieval, ranking and visualization systems for the web [2], [9]. These systems are based on a pre-processed mathematical model of document query space. The relationship between possible query terms and documents is represented by an m by n matrix A, with ij*th* entry a_{ij}.

The entries a_{ij} consist of information on whether term i occurs in document j, and may also include weighting information to take into account specific properties, such as: the length of the document, the importance of the query term in the document, and the frequency of the query term in the document. $A = [a_{ij}]$ is usually a very large, sparse matrix, because the number of keyword terms in any single document is usually a very small fraction of union of the keyword terms in all of the documents.

After creation and pre-processing of the matrix A, the next step is to computation of the SVD of A. Although the computation does not have to take place in real time, it has to be completed quickly to enable frequent updating of the matrix model

The noise matrix A is reduced by constructing a modified matrix A_k, from the k largest singular values and their corresponding vectors, i.e.

$$A_k = U_k \Sigma_k V_k^T$$

Queries are processed in two steps: a) *Query projection* step: input queries are mapped to pseudo-documents in the reduced query-document space by the matrix U_k, then weighted by the corresponding singular values S_i from the reduced rank, singular matrix Σ_k. b) *Matching step*: similarities between the pseudo-document q' and documents in there reduced term document space V_k^T are ranked by measuring the cosine of the angle between the query and the modified document vector.

III. Description of the Algorithm to Estimate the Largest Singular Values of Matrices of Large Dimensions

Let A be a very large matrix M x N whose singular values cannot be computed due to its size. As we said before, the procedure proposed is composed by two phases,

- *The first phase* is the acquisition of the singular values of the smaller matrices composed by random selection of some columns from the original very large matrix. For example, suppose we are interested in estimate the first (largest) singular value. We can start with a small number n of sampled columns to obtain a smaller matrix and repeat the process with the same number n to obtain different small matrices. For each small matrix we compute the first singular value, these values will be in most cases different. Then, we compute the mean of these firsts singular values (y_i) for $x_i = n$, therefore we get one point (x_i, y_i). For obtain others points (x_i, y_i), we just choose a different value n ($n << N$) and do the same procedure described before.
- *The second phase* is the extrapolation process, in this phase we use the points (x_i, y_i) produced by the phase one and adjust a curve to fit this group of points. Then we use the curve to make a prediction about the first singular value of the original large matrix when $x_i = N$. Obviously this process can be done for others singular values. For this phase, we propose an algorithm based on GP.

IV. Data Extrapolation Using Genetic Programming

Curve fitting is the problem arising when we have different points (data) and we wish to find a function capable to adjust to these points [3]. Then, we can use this function to make extrapolations to predict behave of others points outside the original range (out of sample) where are the input data. Due the random nature of our algorithm to obtain the points (x_i, y_i), we can be sure that the data will always have certain degree of experimental random noise, therefore our strategy is to obtain a curve that represents the general trend of the data for our particular application.

We use GP to achieve the automatic creation of mathematical expressions; this is achieved by feeding genetically a population of mathematical expressions, using the principles of natural selection from the "theory of evolution" of Darwin and genetic operations inspired biologically. The biological operations more common include sexual recombination, mutation and replication

(duplication) [1, 7, 8]. GP use the evolution of populations (of math expressions) with the objective of find the best mathematical expression to adjust to the points (x_i, y_i). In GP we use a fitness function which allows to measure how fit is a math expression (individual) to adjust to the points, therefore each individual have his own fitness. Individuals are then probabilistically selected from the population based on their fitness to participate in different genetic operations. The fittest individuals have better chance of being selected, but unfit individuals are allocated in some trials, so GP is not purely a gradient algorithm.

--The functions used by us were: $\sqrt{x}, \sqrt[3]{x}, \sqrt[5]{x}$, $Lnx, Log_2 x, Log_{10} x, Arctg(x),$.

--The constants used were: 2, 3, 5, 7, 9.

--The operators used were: +, -, /, *.

All this functions, constants and operators, are used by GP to generate mathematical expressions using genetic operations. Hence, with GP we can generate complex functions with any regularity form that follow very accurate some trend of points like, for example:

$$p(x) := \sqrt{\operatorname{atan}(1) + \operatorname{atan}(\log(x)) + \sqrt[5]{(x)} \cdot \frac{\ln\left(7 + \sqrt[5]{x}\right)}{0.69} + 7 - \sqrt[5]{x}}$$

V. Numerical Experiments

We implemented our algorithm using two different kinds of matrices: Dense and Sparse.

To adjust the points of the form (x_i, y_i) we compare our approach with linear regression and non linear regression approaches.

A. Linear Regression

The general model of linear regression can be presented by:

$$y = a_0 z_0 + a_1 z_1 + a_2 z_2 + \ldots + a_m z_m + e$$

Where $z_0, z_1, z_2, \ldots, z_m$ are the (m+1) different functions. This model includes a lot of different regressions. One particular case is when the z_i are simple monomials and the regression is called polynomial. Is remarkable that the terminology linear only refers to the dependence of the model in the parameters (a_i), but the function z_i can be highly non linear. In our study we use logarithms functions and roots functions, but the most used functions are $R(t)= a\sqrt{t}+b$, $R(t)= a\sqrt{t}+bLn(t)+c$.

B. Non Linear Regression

There are different cases where the linear model is not the best way to adjust the data, in such case is better to use the non linear model. The non linear model has non linear dependence in their parameters a_i. For example one common function used by us is $R(t)= at^b+c$, which includes a lot of roots functions. We will not enter in more details about all the different aspects of the non-linear regression, because is beyond the scope of this paper.

C. Dense matrix 2000 x 3000

In the figure 1, we show the evolution of the singular values for different numbers of columns sampled. We can see that the singular values grow with the numbers of columns sampled. The most relevant information in this graph is the irregularity of the curve that link the points, this show the difficulty associated to make predictions with some groups of points of the singular values for the original large matrix. S1 is the largest singular value and S2 is the second largest singular value.

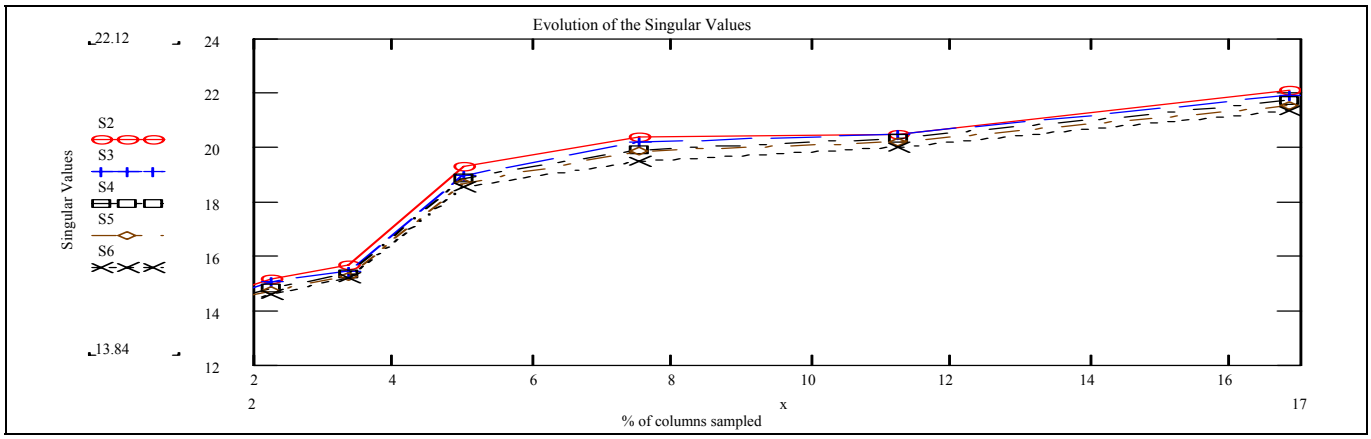

Fig. 1. Singular values of small matrices for different % of sampled columns.

In table 1 we show some results from different experiments and the percentage error in the predictions of the first (S1) and second (S2) largest singular values. We see that the error of our approach is very good with an execution time smaller than linear regression approach. That is, our approach can follow the trend of the sample data better.

TABLE 1
RESULTS FOR DENSE MATRICES

Methods	S_1=	Error	S_2=	Error

	1224.97		28.71	
GP, Sampled Col 20%	1224	0.079%	28.72	0.034%
GP, Sampled Col 2.5%	1232	0.57%	28.407	0.01%
Linear reg, Sampled Col 15%, R(t)= a√t+b, 9 points	1225	0.001%	35.894	25%
Non-linear reg, Sampled Col 5%, R(t)= at^b+c, 13 points	1217	0.40%	31.861	10.9%
Non-linear reg, Sampled Col 2%, R(t)= at^b+c, 5 points	1208	1.38%	32.361	12%

E. Sparse matrix 2500 x 3500

In table 2 we show some results from different experiments and the percentage error in the predictions of the singular values for this case. Similar to the previous one, here we obtain very good results with our algorithm with the smallest execution time. Even in some cases we obtain the best results.

TABLE 2
RESULTS FOR SPARSE MATRICES

Methods	S_1=22.72	Error	S_2=7.8	Error
GP Sampled Col 16%, 5 points	22.67	0.22%	7.03	9.87%
Linear reg, Sampled Col 15%, R(t)= a√t+bLn(t)+c, 11 points	23.018	1.31%	11.506	47%
Linear reg, Sampled Col 15%, R(t)= a√t+b, 11 points	19.235	15.3%	7.579	2.83%
Non-linear reg, Sampled Col 2%, R(t)= at^b+c, 18 points	23.398	2.98%	6.389	18%
Non-linear reg, Sampled Col. 15%, R(t)= at^b+c, 5 points	23.56	3.69%	7.63	2.17%

F. Text Mining Matrix

In this set of experiments we consider a 50000 by 10000 matrix from a text-mining problem. The matrix represents data to be input into an automatic retrieval system. It is sufficiently small that we can use a software package to compute all of the singular values and vectors and compare result with our method.

We took random samples of the rows of the matrix and computed the largest 3 singular values of the smaller matrix constructed form the randomly sampled rows. We repeat the process 100 times and computed the mean and the standard deviation. Results form our experiments are given in table 1.

TABLE 3
ESTIMATES OF SINGULAR VALUES OF A DOCUMENT-QUERY MATRIX

No. Documents sampled	% of Documents	Est. for S_1	Est. for S_2	Est. for S_3
5000	10%	127.8	94.2	72.3
10000	20%	144.6	105.4	86.1
20000	40%	223.4	145.2	106.2
30000	60%	301.1	176.6	124.3
40000	80%	410.2	197.1	138.1
50000	100%	432.2	206.2	167.2

When the number of documents is 50000 they are the actual values.

VI. CONCLUSION

We show here that we can predict with some percentage of error the largest singular values of a matrix without computing the singular values of the whole matrix and using only some random selected columns of the matrix. This matrix whose elements belongs to the interval [0,1], may be dense or sparse. In the case that the matrix is dense the predictions are very good, and although for a sparse matrix the predictions are not so good, but still fairly acceptable. We think that the reason of this difference is that the random selection for sparse matrices may some times not include relevant columns whose elements contain an important contribution to the value of the singular values.

In other order of ideas we also show that GP is a versatile tool to generate functions for extrapolation and curve fitting, in particular we show that its results are comparable with the classical techniques of linear regression and non linear regression and in some cases better. However the functions obtained with GP are very different in form than the classical functions used for extrapolation. It is very interesting how GP produces from just simples functions, operators and fitness function, complex mathematical expressions that adjust smoothly to a group of points. One interesting work than can be developed in a future with GP is to try to generate functions in higher dimensions capable of make extrapolations and interpolations. Additionally is possible to apply a variant of our method to estimate the singular vectors of large matrices.

ACKNOWLEDGMENT

J. A. Author thanks Eng. L. Matheus from Universidad de los Andes, Venezuela, with the support to the last part of the experiments.

REFERENCES

[1] J. Aguilar, F. Rivas, *Introducción a las Técnicas de Computación Inteligente*. Universidad de los Andes, MERITEC, 2001.

[2] M. Berry, Z. Drmac and E. Jessup. "Matrices, Vector Spaces, and Information Retrieval". *SIAM Review*, vol. 41, pp. 335-362, 1999.

[3] W. Cheney and D. Kincaid D., *Análisis numérico las matemáticas del cálculo científico*, Addison-Wesley, 1994.

[4] G. Golub and C. Van Loan, *Matrix Computations*, The Jhon Hopkins University Press, Baltimore, MD, 1996.

[5] N. Higham , D. Higham, *Matlab Guide*, SIAM, 2000.

[6] M. Kobayashi, G. Dupret, O. King and H. Samukawa, Estimation of Singular Value of very large matrices using random sampling, *Computers and Mathematics with applications*, Vol. 42, 2001.

[7] J. Koza, *Genetic Programming II*, MIT Press, 1994.

[8] W. Langdon, R. Poli, *Foundations of Genetic Programming*, Springer, 2002.

[9] A. Langville and C. Meyer, "A Survey of Eigenvector Methods for Web Information Retrieval", *SIAM Review*, October 12, 2004.

[10] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge Univ. Press, Cambridge, UK, (1995).

[11] W. H. Press, B. Flannery, S. Teukolsky, W. Vetterling, *Numerical Recipes: The art of scientific computing*, Cambridge University Press, 1986.