Analysis of data streams using self-organizing methods

RICHARD WASNIOWSKI
Computer Science Department
California State University
Carson, CA 90747, USA

Abstract: - With the increase of data streams generation by sensor networks systems it is desirable to develop and improve methods that can automatically classify streams. This paper discusses some clustering mechanisms, and pilot experiments on the data collected from sensor networks. Using self organizing methods and other techniques we obtain maps that establish a new relationship in acquired data structure. Analysis of the obtained clusters are made by Group Method of Data Handling . The results of the preliminary experiments validate the feasibility of this approach, and at the same time, indicate directions of further work. In order to reveal new structures in data streams we apply Self-Organizing Maps and Group Method of Data Handling algorithms.

Key-Words: - self-organizing map, polynomial algorithm, group method of data handling

1 Introduction

Monitoring streams of data generated by sensor nets is currently perceived as one of the important challenges for the data analysis and data processing. Complex systems are usually described by huge amounts of data, parameters and often have chaotic behavior. To reveal interrelations among these parameters new methods and algorithms of computer analysis have to be developed. These problems are of exceptional interest now when computer technologies provide enormous possibilities for collecting, storing and processing of information obtained by tracing system behavior. It is believed that this information is particularly important for the prediction of the system behavior. In this case researchers dealing with such types of systems often try to apply statistical or neural networks methods for the discovery of new knowledge about the system. In framework of this approach self-adjustable method known as self-organization methods are especially interesting because they could be applied in autonomous regime without external setting for every particular case. In this article we use two types of self-organization methods for investigation of financial market behavior by analyzing data series The study of the system behavior is a very important and actual question.

2 Problem Formulation

Many classical approaches and methods are pushed to, or even beyond their limits by the sheer size of the sensor data and by the high frequency of their arrival. Once sensor data have been monitored, pre-processed and temporarily stored using data base technology, their further analysis, however, is at present left to specialized, domain-dependent software packages. Evaluation algorithms for sensor data communicate with the data base hosting the data via languages like SQL for retrieval purposes only, but are themselves entirely coded in traditional imperative and object-oriented languages. This kind of reduction of data base technology to a subordinate role as pure data producer and its exclusion from the much more interesting and rewarding field of data analysis is challenged by our approach.

3 Self-organizing methods

2.1 SOM

The SOM is an unsupervised-learning neural-network method that provides a similarity graph of input data. A typical simplified version of the SOM algorithm consists of two steps iterated for every sample: finding the best matching units and adaptation of the weights.

Initially all neuron weights are initialized by uniform distribution on the interval $[0,1]$. The distance between two neurons of the two-dimensional grid is found as

$$d_j = \|x - w_j\| = \sqrt{\sum_{i=0}^{N-1} (x_i - w_{ij})^2},$$

where j is the index of neuron in the net, i is the dummy index of vector components, w_{ij} is the weight of synapse, which matches i -component of input vector with output neuron j. Thus, we find for each vector x such a neuron c, for which the distance between x and c is the smallest

$$c \equiv \|x - w_c\| = \min_j d_j .$$

Vectors of weights w_j are adapted using the following rule

$$w_j(t+1) = \begin{cases} w_j(t) + \alpha(t) h_{cj}(t) \cdot | x(t) - w_j(t) |, j \in N_c, \\ w_j(t), \quad j \notin N_c \end{cases}$$

where N_c describes the neighborhood of”neuron-winner” (3), $\alpha(t)$ is the learning-rate factor, which decreases monotonously with the regression steps, $h_{cj}(t)$ is a scalar multiplier called the neighborhood function. Thus, training process leads to reduction of the distance between input signal and position of ”neuron-winner” as well as to the reduction of the Euclidean distance between input vector and any vector $w_j, j \in N_c$. The SOM-map is obtained as the result of this mapping of vector x on neurons plane. Similar input vectors are placed closely to each other on the SOM-map. Such procedure makes it possible to single out the input information and locate the input vectors in the vicinity of similar vectors on the map without preliminary training, using only internal properties of input data. Next processing of the input data by chosen rule (4) leads to the neuron-grids training and formation of corresponding clusters.

2.2 GMDH

Dynamics of data streams behavior can be analyzed by methods of multiple regression [8]. But these methods take into account the whole set of input data and overload the final model (equation of regression). Self-organizing algorithms do not have this disadvantage. Group Method of Data Handling (GMDH) creates the model that includes only the most influential variables [7, 11]. The GMDH algorithms are based on a sorting-out procedure of model simulation and provide the best model according to the criterion given by the researcher. This model describes relations between their elements and the state of the whole system. Most of GMDH algorithms use polynomial referenced functions. General connections between input and output variables can be shown by Volterra functional series. A discrete analogue of Volterra series is Kolmogorov-Gabor polynomial

$$y = a_0 + \sum_{i=1}^{N} a_i x_i + \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} x_i x_j + \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} a_{ijk} x_i x_j x_k + ...,$$

(5)

where y - output variable vector, $(x_1, x_2, ..., x_N)$ - input data,

$(a_1, ..., a_N, ..., a_{ij}, ..., a_{ijk}, ...)$ - vector of coefficients or weights. Input data might consist of independent variables, functional expressions or finite residues. The key feature of GMDH algorithms is a partition of input data into two subsets. The first one is used to compute coefficients of the polynomial using the list square technique and to evaluate internal error by some criterion. The second one is used to calculate external error using information, which is not applied for the coefficients computations. Principles of self-organization manifest themselves in rationalization of optimal polynomial search. Internal criterion monotonously decreases when complexity of polynomials increases, simultaneously external criterion passes its minimum. Then it is possible to choose polynomial of optimal complexity, which is unique for this criterion. In other words, we provide sorting-out procedure for partial polynomials to find polynomial of optimal complexity (optimal model). It shows the dependence of the output variable on the most influential variables, which are chosen from all input variables. External criterion reaches its minimum on optimal model. Interpretation of the results is similar to multiple regression logic: the bigger is the coefficient - the more influential is the variable near it.

4 Conclusion

The results obtained by SOM algorithm are sufficiently demonstrative and make it possible to understand relationships inherent to such a complex system like stock market. The GMDH algorithm, in its turn, makes it possible to establish analytical dependence of the stock prices of the companies, which have correlated behavior. The introduced self-organizing methods complement each other. The obtained results are self-consistent and allow finding an optimal non-overloaded model, which is easy for economic interpretation. Using the combination of such methods promises to be very perspective.

References:

[1] John Y. Campbell, Andrew W. Lo, Archie Craig MacKinlay, John W. Campbell, Andrew Y. Lo: The Econometrics of Financial Markets, Princeton University Press, Princeton (1997)

[2] Cox D. R., Hinkley D. V. and Barndorff-Nielsen : Time series models in econometrics, finance and other fields, Chapman & Hall (1996)

[3] Matlab, The MathWorks, Inc., http://www.mathworks.com/products/matlab/

[4] Message Passing Interface (MPI), http://www.mpi-forum.org/

[5] A.-P. N. Refenes, A. N. Burgess, Y. Bentz: Neural Networks in Financial Engineering: A Study in Methodology (1997) IEEE Transactions on Neural Networks vol 8, n 6, November 1997

[6] G. J. Deboeck and T. Kohonen (eds): Visual Explorations in Finance with Self Organizing Maps, Springer Finance, New York (1998)

[7] T. Kohonen, Self-Organizing Map, 2nd ed., Springer-Verlag, Berlin, 1995.

[8] J.A. Muller, A.G. Ivakhnenko and F. Lemke, GMDH algorithms for complex system modelling, Mathematical and Computer Modelling of Dynamical Systems, vol.4. no.4. pp. 275-316, 1998.

[9] L. Anastasakis and N. Mort, The development of self-organization techniques in modelling: a review of the group method of data handling (GMDH), ACSE Research Report No 813, University of Sheffield, UK, 2001 or www.shef.ac.uk/acse/research/students/l.anastasakis/813.pdf

[10] E. Ferster, B. Rents, Methods of correlation and regressionanalysis:Transl. from german - Moskow: Finance and statistics, 1983. (In russian)

[11] T. Kohonen, S. Kasaki, K. Lagus, et.al. Self-Organization of a massive document collection. IEEE transaction of Neural Networks 2000, V.11, ^{1}3, pp.574-585.

[12] F. Lillo and R. Mantegna, Variety and volatility in financial markets, Physical Review E, vol.62. no.5. pp.6126-6134, 2000.

[13] H.R. Madala and A.G. Ivakhnenko, Inductive Learning Algorithms for Complex Systems Modeling, Boca Raton: CRC Inc., 1994.

[14] R. Mantegna, Hierarchical Structure in Financial Markets, e-print cond-mat/9802256 v.1, 1998.

[15] MATLAB*P, A. Edelman, MIT, http://www-math.mit.edu/ ~ edelman/

[16] A Parallel Linear Algebra Server for Matlab-like Environments, G. Morrow and Robert van de Geijn, 1998, Supercomputing 98 http://www.supercomp.org/sc98/TechPapers/sc98_FullAbstracts/Morrow779/index.htm

[17] MATLAB Parallel Example, Kadin Tseng, http://scv.bu.edu/SCV/Origin2000/matlab/MATLABexample.shtml

[18] MultiMATLAB: MATLAB on Multiple Processors A. Trefethen et al, http://www.cs.cornell.edu/Info/People/lnt/multimatlab.html

[19] ParaMat, http://www.alphadata.co.uk/dsheet/paramat.html

[20] Investigation of the Parallelization of AEW Simulations Written in MATLAB, Don Fabozzi 1999, HPEC99

[21] Matpar: Parallel Extensions to MATLAB, http://hpc.jpl.nasa.gov/PS/MATPAR/

[22] MPI Toolbox for Matlab (MPITB), http://atc.ugr.es/javier-bin/mpitb_eng

[23] Brown, M. and C. Harris (1994). *Neurofuzzy Adaptive Modelling and Control*, Prentice Hall, New York.

[24] Ewing, G. O., and R. Wolfe (1977). Surface Feature Interpolation on Two-dimensional Time-space Maps, *Environment and Planning A*, Vol.9, pp.429-437.

[25] Hayashi, C. (1952). On the Prediction of Phenomena from Qualitative Data and the Quantification of Qualitative Data from the Mathematical Statistical Point of View, *Annals of the Institute of Statistical Mathematics*, Vol. 3, pp 69-98.

[26] Shimizu, E. (1992). Time-space Mapping and Its Application in Regional Analysis, *Doboku Keikakugaku Kenkyu*, No.10, pp.15-29, in Japanese.

[27] Tolman, E. (1948). On Cognitive Maps in Rats and Men, *Psychological Review*, Vol.55, pp.189-208.

[28] Torgerson, W. S. (1952). Multidimensional Scaling, I. Theory and Method, *Psychometrika*, Vol.17, pp.401-419.

[29] Rumelhart, D. E., J. L. McClelland and the PDP Research Group (1987). *Parallel Distributed Processing*, Cambridge, MA : MIT Press.

[30] Automatic Array Alignment in Parallel Matlab Scripts, I. Milosavljevic and M. Jabri, 1998

[31] Parallel MATLAB Development for High Performance Computing with RTExpress, http://www.rtexpress.com/